

Proceedings of the Seminar in Computer Science (CS-E4000), Spring 2019

Mario Di Francesco, Antti Ylä-Jääski, and Verónica Toro-Betancur

Tutors for seminar topics

Alex Jung, Ari Keränen, Abbas Mehrabidavoodabadi, Aleksi Peltonen, Amit Tambe, Antti Ylä-Jääski, Buse Gul Atli, Behrouz Jedari, Chao Zhu, Gazi Illahi, Gopika Premsankar, Hans Liljestränd, Hannu Flinck, Hong-Linh Truong, Jukka Nurminen, Jordan Whitefield, Jiayue Xu, Keijo Heljanko, Lachlan Gunn, Mario Di Francesco, Mika Juuti, Maria Montoya-Freire, Mohit Sethi, Pranvera Kortoci, Risto Wichman, Samuel Marchal, Siddharth Rao, Sanna Suoranta, Tuomas Aura, Thanh Bui, Tommi Gröndahl, Vinay Gautam, Vesa Hirvisalo, and Verónica Toro-Betancur

Preface

The *Seminar on Network Security*, *Seminar on Internetworking* and *Seminar on Software Technology and Systems Research* were previously separate Master's level courses in computer science at Aalto University. These seminar courses have now merged into one seminar course. These seminar series have been running continuously since 1995. From the beginning, the principle has been that the students take one semester to perform individual research on an advanced technical or scientific topic, write an article on it, and present it on the seminar day at the end of the semester. The articles are printed as a technical report. The topics are provided by researchers, doctoral students, and experienced IT professionals, usually alumni of the university. The tutors take the main responsibility of guiding each student individually through the research and writing process.

The seminar course gives the students an opportunity to learn deeply about one specific topic. Most of the articles are overviews of the latest research or technology. The students can make their own contributions in the form of a synthesis, analysis, experiments, implementation, or even novel research results. The course gives the participants personal contacts in the research groups at the university. Another goal is that the students will form a habit of looking up the latest literature in any area of technology that they may be working on. Every year, some of the seminar articles lead to Master's thesis projects or joint research publications with the tutors.

Starting from the Fall 2015 semester, we have merged the three courses into one seminar that runs on both semesters. Therefore, the theme of the seminar is broader than before. All the articles address timely issues in security and privacy, networking technologies and software technology.

These seminar courses have been a key part of the Master's studies in several computer-science major subjects at Aalto, and a formative experience for many students. We will try to do our best for this to continue. Above all, we hope that you enjoy this semester's seminar and find the proceedings interesting.

Seminar papers

Bruno Duarte Coscia , <i>V2X for Autonomous Vehicles</i>	9
<i>Tutor: Chao Zhu.</i>	
Antero Vaarnamo , <i>User engagement with public displays</i>	21
<i>Tutor: Maria Montoya-Freire.</i>	
Shamim Biswas , <i>Mobile Edge Computing Assisted Internet of Things</i>	33
<i>Tutor: Abbas Mehrabidavoodabadi.</i>	
Hongkuan Wang , <i>Machine Learning for Video Encoding</i>	47
<i>Tutor: Gazi Illahi.</i>	
Ervin Oro , <i>Android App Collusion</i>	59
<i>Tutor: Jordan Whitefield.</i>	
Kidus Mammo , <i>State of the Art of IoT Data Model Translation</i>	73
<i>Tutor: Ari Keränen.</i>	
Timur Kartaev , <i>Electronic identification for citizens</i>	87
<i>Tutor: Sanna Suoranta.</i>	
Sriram Vobilisetty , <i>Lightweight modern convolutional object detectors</i>	99
<i>Tutor: Xu Jiayue.</i>	
Satenik Hovsepyan , <i>Analysis of fallback authentication mechanisms</i>	109
<i>Tutor: Siddharth Rao.</i>	
Tolgahan Akgun , <i>Is Narrowband-IoT an efficient alternative for large-scale connected device systems?</i>	123
<i>Tutor: Pranvera Kortoçi.</i>	
Matti Mäki-Kihniä , <i>Survey on touch dynamics based continuous authentication</i>	135
<i>Tutor: Sanna Suoranta.</i>	
Miska Kankkonen , <i>Adversarial Attacks in Deep Reinforcement Learning</i>	149
<i>Tutor: Buse Gul Atli.</i>	
Ekaterina Shmeleva , <i>Data exchange between smartphones using visible light</i>	163
<i>Tutor: Maria Montoya Freire.</i>	
Sataponn Phutrakul , <i>Video Crowdsourcing in Vehicular Networks: Use Cases and Challenges</i>	175
<i>Tutor: Chao Zhu.</i>	
Jussi Hietanen , <i>GPS jamming and spoofing</i>	189
<i>Tutor: Tuomas Aura.</i>	
Mikko Pohjalainen , <i>IPv6 Honeypots for IoT Devices</i>	201
<i>Tutor: Amit Tambe.</i>	
Adika Sulaeman , <i>Digital Twins for Industrial Edge 4.0: Concepts and</i>	

<i>Tools</i>	215
<i>Tutor: Hong-Linh Truong.</i>	
Kaspar Papli , <i>Defence Against Authentication Server Breach Attacks: A Review of Multi-Server Identity Providers</i>	229
<i>Tutor: Siddharth Rao.</i>	
Giacomo Mariani , <i>Formal Verification of EAP-TLS with mCRL2</i>	243
<i>Tutor: Aleks Peltonen.</i>	
Yuxi Xia , <i>Privacy-Preserving Machine Learning using trusted hardware</i>	257
<i>Tutor: Samuel Marchal.</i>	
Lukas Klein , <i>Beyond the Google's BeyondCorp</i>	271
<i>Tutor: Mohit Sethi.</i>	
Oscar Stigzelius , <i>Service mesh concept and its realizations</i>	285
<i>Tutor: Hannu Flinck.</i>	
Mats Mulder , <i>Automatic rumour Detection on Social Media: A Survey</i>	301
<i>Tutor: Tommi Gröndahl.</i>	
Sonika Ujjwal , <i>Microservices architecture in I-IoT applications: Motivation and security challenges</i>	315
<i>Tutor: Hirvisalo Vesa.</i>	
Alexander Gödeke , <i>Usability problems of email authentication standards with mailing lists</i>	329
<i>Tutor: Thanh Bui.</i>	
Ruiyang Ding , <i>A Survey on Fallback Authentication Mechanism</i> .	343
<i>Tutor: Siddharth Rao.</i>	
Aleksandra Zhuravleva , <i>A Comparison of Intelligent Edge Computing Platforms</i>	355
<i>Tutor: Behrouz Jedari.</i>	
Javier Benitez Fernández , <i>Survey on Software Defined Radios in IoT</i>	371
<i>Tutor: Verónica Toro-Betancur.</i>	
Tuomas Ebeling , <i>QKD in Small IoT Devices</i>	383
<i>Tutor: Jukka Nurminen.</i>	
Mohammad Elhariry , <i>Machine Learning for image and video encoding</i>	397
<i>Tutor: Gazi Illahi.</i>	
Daniel Saxén , <i>Automated testing of Deep Neural Networks</i>	411
<i>Tutor: Jukka K. Nurminen.</i>	
Héctor Bállega Fernández , <i>Mobile Edge Computing Assisted Internet of Things</i>	425
<i>Tutor: Abbas Mehrabidavoodabadi.</i>	
Arnab Chowdhury , <i>Mobile Crowdsensing in Internet of Things</i> ..	439
<i>Tutor: Abbas Mehrabidavoodabadi.</i>	

Eric Cornelissen , <i>Formal Verification using Tamarin Prover</i>	453
<i>Tutor: Aleksi Peltonen.</i>	
Amaanat Ali , <i>How deep is the learning used in wireless communications? Can we go deeper?</i>	469
<i>Tutor: Risto Wichmann.</i>	
Ville Viinikka , <i>Survey on machine learning for signal detection</i> ..	483
<i>Tutor: Verónica Toro-Betancur.</i>	
Roope Lähetkangas , <i>Buffer overflow: Attacks and Defences</i>	495
<i>Tutor: Thanh Bui.</i>	
Tanvi Jain , <i>Barcodes for mobile computing applications</i>	509
<i>Tutor: Mario Di Francesco.</i>	
Noah Nettey , <i>Machine learning based rate adaptation HTTP video streaming</i>	523
<i>Tutor: Gazi Illahi.</i>	
Miika Kanerva , <i>Function composition in serverless computing</i> ...	535
<i>Tutor: Mario Di Francesco.</i>	
Auli Mustonen , <i>A Survey on Intrusion Detection Datasets</i>	553
<i>Tutor: Buse Gul Atli.</i>	
John Cheng , <i>Comparison of deep convolutional neural network for mobile applications</i>	567
<i>Tutor: Jiayue Xu.</i>	
Natalia Gavrilenko , <i>A survey of verification tools for weak memory models</i>	577
<i>Tutor: Keijo Heljanko.</i>	
Robert Seligmann , <i>Virtual and Augmented Reality in the Industrial Sector</i>	591
<i>Tutor: Antti Ylä-Jääski.</i>	
Akzharkyn Duisembiyeva , <i>Fraudulent user identification methods on social media</i>	605
<i>Tutor: Mika Juuti.</i>	
Luong Tran , <i>CoAP Congestion control and DoS attacks</i>	617
<i>Tutor: Mohit Sethi.</i>	
Shiting Long , <i>A Review of 2-Dimensional DNA Origami</i>	629
<i>Tutor: Vinay Gautam.</i>	
Xuan Nguyen , <i>Tor: Attack and Defense</i>	643
<i>Tutor: Lachlan Gunn.</i>	
Andreas Hitz , <i>Designing solutions for pervasive displays: A survey</i>	655
<i>Tutor: Maria Montoya-Freire.</i>	
Lasse Kärkkäinen , <i>Neural network training with Adam, Nesterov and Standard Gradient Descent</i>	669
<i>Tutor: Alex Jung.</i>	
Gilang Hamidy , <i>Pointer Analysis: Revisiting Prevalent Techniques with Current Developments, Challenges, and Uses</i>	685

Tutor: Hans Liljestrand.

Christian Yudhistira, *Access Control for The Internet of Things* . 699

Tutor: Mohit Sethi.

Tommi Askola, *Distributed data analytics at the edge* 713

Tutor: Gopika Preamsankar.

Kamrul Islam, *Data Analytics using Azure IoT Edge* 725

Tutor: Behrouz Jedari.

Ngadhnjim Plaku, *Bluetooth LE Locator Security and Privacy* ... 739

Tutor: Tuomas Aura.

V2X for Autonomous Vehicles

Bruno Duarte

bruno.duarte@aalto.fi

Tutor: Chao Zhu

Abstract

Vehicle-to-everything (V2X) is a concept that refers to communications between (autonomous) vehicles and their interchange of information. These communications aim not only to guarantee more safety in autonomous vehicles but also to bring new services to emerging markets. Beyond the applications, this article intends to explore the underlying technologies that make V2X a reality. We identify the technology in use cases, and we address their limitations and capabilities.

KEYWORDS: v2x, autonomous vehicles, wireless communication, cloud computing, internet of things.

1 Introduction

The automobile industry has evolved since the origin, but in recent years, with the coalition of Information and Communication Technology (ICT), new improvements have emerged, such as the possibility of autonomous vehicles or so-called self-driving vehicles. The improvement of the transportation experience as well as the safety of pedestrians is the main motivation driving this field.

To enable a vehicle to be autonomous, it necessarily implies that the vehicle is connected either to other cars by the surround or to infrastructures as the cloud. The road with communication capabilities can also have awareness of the environment that many times cannot be perceived by a human driver. These misperceptions of the human driver due to fog, rain or snow can be solved with the assistance of sensors that can provide guidance to avoid accidents. Information from congested routes or temporary blocks also aids the driver to circumvent traffic. All these new capabilities are encouraging enough to enable the eventual possibility of having complete autonomous vehicles in our society.

The communication capabilities of vehicles has obtained designated names from the Third Generation Partnership Project (3GPP) group. The names depend on the other party involved in the communication, such as V2V (vehicle-to-vehicle), V2P (vehicle-to-pedestrian), V2I (vehicle-to-infrastructure) also referenced as V2N (vehicle-to-network) [2] and more generally V2X (vehicle-to-everything) [1] to all of the aforementioned cases [3].

Currently, there are two types of communication technologies that enable V2X: WLAN-based (DSRC, Dedicated Short Range Communications) uses the underlying radio communication provided by 802.11p [4] and Cellular-based [2] that operates over a cellular network. The former is independent of the network in remote areas in which there is a lack of cellular coverage, the latter also supports communication between vehicles (V2V) and allows access to the cloud and rich services of that ecosystem.

This paper address the challenge of identifying the appropriate technology for different V2X scenarios. We survey research on hybrid architectures that employ DSRC and cellular networks, as well as novel approaches, such as "mmWave" [9].

This paper is an overview of the Vehicle-to-everything (V2X) ecosystem, stakeholders and the enabling technologies. In Section 2, we present basic concepts associated with the field as well as the motivation behind V2X. Section 3 presents the underlying technology, with their limitation and capabilities. Section 4 describe some of the stakeholders and organizations behind the standardization. Section 5 concludes the paper with some remarks.

2 Background and motivation

Guarantee the safety of passengers it is an ongoing problem as the CARE Database (Community database on Accidents on the Roads in Europe) shows an average of almost 1.2 million injury accidents between 2006 and 2015 and an average of 33030 fatalities in crash accidents between the same period [8]. In the United States (U.S.), the scenario is even more worrisome with 3.9 million injuries and 32.999 fatalities in the year 2010 [7]. Autonomous vehicles brings controversies due to decisions that must be taken by machines to guarantee human safety. Despite the controversy, the automobile industry increase important investments and research on the field to overcome the ongoing traffic accidents.

Intercommunication is part of the challenge since to acknowledge the surrounding, vehicles need to rely on perception systems e.g. cameras and LIDARs (a portmanteau of light and radar) that uses light reflection to identify objects among other sensors [6]. Frequently, these sensors are sensitive to weather conditions and are prone to have reflectivity issues, leaving the vehicle in a temporary blind state. In these situations, V2X communications can provide the needed awareness from a different source to counter the blindness state.

Recently, DSRC has been disputed by publications whether is capable of providing reliable and efficient V2X communication, as a result of poor performance in high vehicle density scenarios [3]. Instead of immediately rejecting the DSRC technology as an implausible solution, we are confident that DSRC is useful under particular situations and use cases, mainly by the accessible cost [9] of the associated equipment.

2.1 Basic concepts

To grasp the V2X ecosystem, first we need to acknowledge some concepts:

Wireless ad hoc network

A wireless ad hoc network is a decentralised type of a wireless network. This network does not require a specific intermediate infrastructure to establish a connection between peers or nodes. In conventional networks, is common to rely on routers and access points to provide the connectivity whereas in a ad hoc network, each node can interact and transmit data with other nodes directly. Wireless ad hoc network takes the WANET acronym whereas Mobile ad hoc networks are designated as MANET [10].

Vehicle-to-Vehicle (V2V) Communications

Vehicle ad hoc networks are called VANET [16], and V2V refers to the exchange of data between vehicles on an application layer. VANETs is based on a Wireless ad hoc network. Equipment that supports V2V applications transmits information about the location, dynamics and other attributes as part of the V2V service. The V2V payload must be flexible in order to accommodate different information contents. V2V is predominantly broadcast-based; V2V includes the exchange of V2V-related application information between distinct vehicles directly [1].

Vehicle-to-Infrastructure (V2I) Communications

Is a communication model where vehicles communicates with Road Side Units (RSUs) to interchange information. The RSU are fixed stations located along the highway to share information about environment conditions. The RSU sends application layer information to a group of vehicles or a single vehicle supporting V2I applications. Even though V2N (vehicle-to-network) services may need to be further discussed, they are considered to some extent to be V2I depending of the case [1].

Vehicle-to-Pedestrian (V2P) Communications

As we can infer, in this model the communication is done between the Pedestrian and the vehicle. However, in this case is very likely that the Pedestrian is connected to a mobile network. The Pedestrian can receive information from the Road Side Unit (RSU) or also from vehicles. We can think of common scenarios as an attempt of the Pedestrian to pass by a signalized pedestrian crossing, or a person with physical impairment as a wheelchair to signalized his attempt to go over a crosswalk or a intersection [1].

2.2 Motivation for V2X

The main motivation behind V2X technology is the safety aspects of human in transit with vehicles. Bad vehicle status, weather, high speed, road conditions, violation of intersections, small distance or minimal margin of safety are the common causes of accidents on the road made by the driver. V2X aims to reduce the number of accidents by keeping the speed, distance, and maneuvering under control in an automated fashion [15]. In the following sub-section we show some relevant use cases.

2.3 Use cases for V2X

Forward Collision Warning

The Forward Collision Warning is one of the V2V applications in which intends to warn the driver of a potential collision with another vehicle in the same lane and direction of transit. Mainly this application is due to a change of speed of any of the vehicles. The vehicles with V2V services, constantly exchange messages, with location, speed, acceleration and optional estimated trajectory to determine if a notification should be raise to the driver [1].

V2V Use case for emergency vehicle warning

This service focus on emergency vehicles, i.e., an ambulance, to notify near vehicles in the road about the need of freeing the path of the ambulance. The vehicles with V2V services, in a range of 300 to 500 meters from the ambulance would receive the notification, including those without line-of-sight path. In addition, for obvious reasons the latency of this message is critical and is expected to be less than 100 ms [1].

Cooperative Adaptive Cruise Control

Cooperative Adaptive Cruise Control also referred as "CACC", corresponds to a set of vehicles with V2V capabilities that forms a group in which they share common benefits. The CACC provides convenience and safety resembling a flock of birds due to the travelling proximity of the vehicles. The vehicles that participates in the CACC move at the same speed mitigating road congestion and improving the use of fuel efficiently [1].

Road safety services

This service focus on raise awareness about danger to vehicles, or in general to User Equipment (UE) with V2I capabilities. Vehicles approaching a street intersection at high speed might be exposed to delicate scenarios. This service can be extended for VIP individuals, e.g., the President of a country that for safety reasons need to have a clear path at high speed, but at the same time notify the surroundings about the trajectory, this also applies for parades in urban areas. The messages are transmitted between the UEs to the RSU under V2I communication or vice versa to notify a specific area as shown in Fig. 1. The drivers receive the messages and act accordingly the context [1].

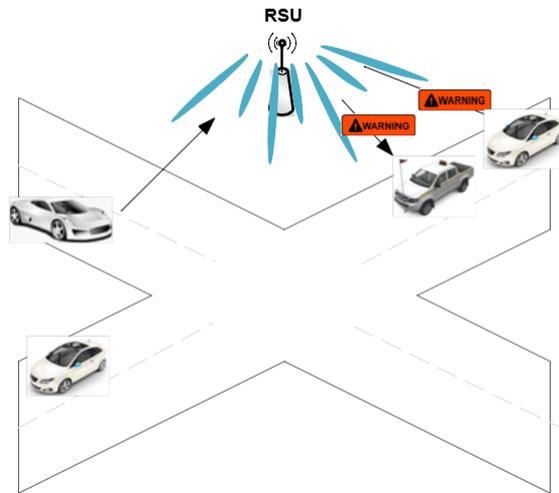


Figure 1. Road safety services via an Road Side Unit [1]

Automated Parking System

Automated Parking System (APS) is a compound service, involves communication between vehicles, the infrastructure (building itself) and services on the cloud. Trough smartphones, the driver can locate free slots in a garage, reserve more hours in the parking lot in addition to the payment fee of the parking service. The occupancy of the slots in the parking lot are updated by the connected vehicles in the garage. The identification of the driver and the availability of slots is being updated by a database on the cloud to provide real-time information to other drivers that aims to use the service [1].

V2X in areas outside network coverage

This use case serve as a fallback when vehicles lack of coverage by the cellular network. Vehicles with V2X capabilities, can exchange messages and still provide some services when experiencing the absence of coverage, for example in rural areas as shown in Fig. 2 [1].

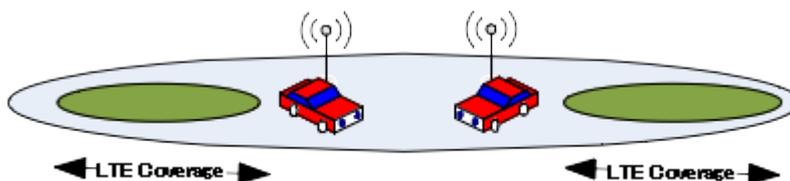


Figure 2. Vehicles outside network coverage [1]

Streaming video in vehicles

Besides the streaming of video from cameras for safety purposes, entertainment videos from public services on the cloud is a reality from the consumer perspective. The overload of the bandwidth with either of these stream of data is a challenge to overcome. V2X capabilities also needs to guarantee the quality of transmission and the efficient use of the network to deliver entertainment content for the passengers. Receiving the content from services in the cloud and distribute among other vehicles that access the same resource in a given VANET is the proper way to decongest the network from heavy transmission of data [12].

3 Underlying technology

Dedicated Short Range Communications (DSRC)

Dedicated Short Range Communications (DSRC) is a communication service that operates from short to medium range, developed to enable V2X communication, i.e., vehicle-to-vehicle and vehicle-to-infrastructure. It was designed with the purpose to provide high data transfer and low latency in a specific area of communication [16]. DSRC operates over radio spectrum bands that depends on the region, these bands can be used up to 7 channels depending of the V2X use case [3] [16]. Since DSRC are mainly apply to VANETs, routing of ad hoc networks is a whole topic with proactive, reactive routing protocols, and position-based routing. Broadcasting, forwarding and beaconing and location service are among the features that enable this technology [16].

Cellular networks for V2X

Cellular technologies can provide unique traits that DSRC communication lacks behind, such as: a) Higher network capacity with higher bandwidth; b) Wider cellular coverage range; c) Consist in a mature technology that is already deployed. However, cellular networks has a centralized control, meaning that all transmission needs to reach first the Base Station (BS) to forward the communication to the correspondent vehicle. This property affects the requirement of short delay by safety applications over V2X [3].

DSRC-cellular hybrid

Combining the strengths of DSRC technology with the unique features of cellular technologies produces hybrid solutions that attempt to fulfill all the expected requirements. Cellular networks can act as a fallback when vehicle-to-vehicle communications are very distant or there are multiple hops in the connection. In addition, access to the Internet is a major factor to take into account as well as the property of acting as a backbone of the entire ecosystem. In regard to the routing protocols, as stated in the DSRC section, these majoritarily rely on their position and other information of the vehicles in the range. When vehicles are out of range for routing purposes, cellular networks can fill the gap to connect these segment and mitigate the fragmentation of the network. Moreover, the possibility of providing Internet access enable a whole new set of applications over V2X. Finally, when cellular congestion peaks due to high load of data transmission, a part of the DSRC spectrum can be utilized to support the communication [3].

"mmWave" Technology

mmWave is next-generation wireless technology, with a multi-gigabit capability and beamforming technique. One of the main promising features of mmWave is the bandwidth. In addition to the bandwidth, the second feature is the short wavelength. The millimeter level of the wavelength enable the possibility to provide a great amount of antennas in a small area. The beamforming technique consist of a signal that once processed generates directional signal transmission by an antenna array. Beamforming is very useful since reduces the interference and assist the localization between other vehicles in the environment. Another property is the manageable Doppler effect that depends of the frequency and mobility of the vehicle, with the beam of the mmWave the Doppler effect has no effect in the communication between vehicles [9]. Fig. 3 shows a comparison between the amount of wireless links in a VANET using mmWave and DSRC, mmWave perfoms better with a clear advantage over DSRC.

3.1 Challenges for V2X

In this section, we would mention some of the challenges that the V2X ecosystem brings to the research community.

On the first place, autonomous vehicles needs to interpret and understand the environment. The nature of MANET and VANET networks

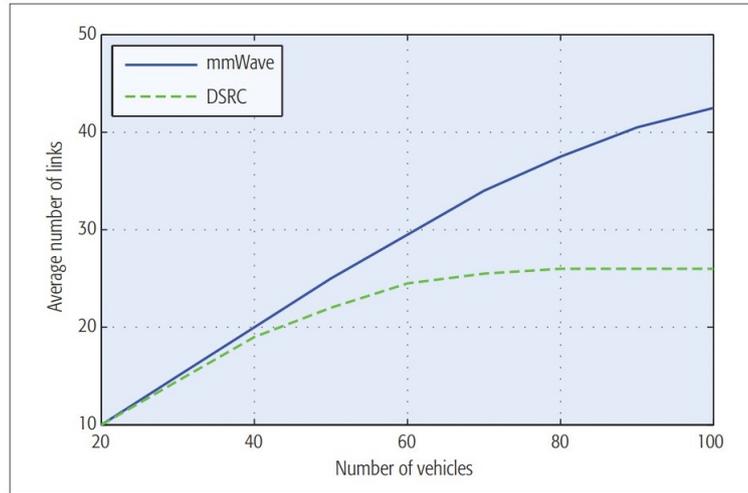


Figure 3. Average number of wireless links in simulation [9]

that suffers ongoing changes provides a complex and very dynamic environment that requires continuous adaptation in order to guarantee a reliable and smooth experience by the driver [6]. Security is not a minor issue, hardening the User Equipment in order to provide integrity as well as protecting the identity of the driver are concerns that are tangled to a myriad of policy implications and law enforcement. Light and fast authentication frameworks that protects from inside or outside attackers to the vehicular network is an important matter that can not be overlooked [16]. Policymakers and regulators, face ethical and cultural challenges due to the sensitive matter of safety and life exposition in urban areas. Drivers, manufactures, corporations and stakeholders in general needs to understand their rights and obligations in this new ecosystem [6]. For engineers, the broadcasting of messages in the VANET constantly represents a threat of flooding the network with packets, requiring the came up with new optimal and efficient way of transmission for ad hoc networks [16]. On hybrid architectures, with DSRC–cellular capabilities the constant effort to adapt and fulfill the fragmentation of the network leads to network overhead, mainly in urban areas where seamless communication is not always possible by the high density of nodes, and the techniques to move from DSRC to Cellular connectivity by the vehicles is not as smooth as expected [3].

4 Standardization and organizations

DSRC Standards

As discussed in Section 3, DSRC is one of the main technologies that enable V2X communications. There are more than one DSRC developed standard depending on the network architecture and the standardization organization, i.e. IEEE in North America, the European Telecommunications Standards Institute (ETSI) in Europe, and the Association of Radio Industries and Businesses (ARIB) in Japan [3].

Cellular Standards

Cellular V2X, also referred as C-V2X was developed within the 3rd Generation Partnership Project (3GPP). 3GPP unites seven telecommunications standard development organizations (ARIB, ATIS, CCSA, ETSI, TSDSI, TTA, TTC), known as "Organizational Partners" and the work is organized in three groups: Radio Access Networks (RAN), Services Systems Aspects (SA), Core Network Terminals (CT) [11]. The standard number 14 released in 2016 introduced four modes. Mode-1 and mode-2, are also known as device-to-device (D2D) communications or proximity services (ProSe), mode-3 and mode-4 aims to guarantee vehicular safety requirements in regard of communication reliability and latency [13]. In addition, there is an enhancement of 3GPP for V2X services, called "eV2X" that improves the technology in four areas: vehicles platooning, advanced driving, extended sensors and remote driving [14].

5G Automotive Association (5GAA)

5GAA is a multi-industry association that represents car manufacturers with the aim of developing, test and promote communications solutions, initiate their standardization and accelerate their commercial availability and global market penetration to address societal need [5].

5 Conclusion

The increasing market share on V2X technologies, is validated by the consolidation of the vehicles manufactures as an association (5GAA), the well-established standards and the constant research from institutions that aims to patent algorithms, protocols and develop Nobel approaches to make the autonomous vehicles a reality in the society. More than the

pure economical factor, the possibility of mitigating transit accidents and elevate the quality of life of citizens is also a clear motivation for governments. V2X brings his own revolution affecting many sectors, but at the same time, many technical challenges need to be solved to start experiencing the benefits of autonomous vehicles in our lives. Therefore, this paper provided an overview of the V2X technology, with their use cases and promising new experiences.

References

- [1] 3GPP. Study on LTE support for Vehicle-to-Everything (V2X) services. Technical Report (TR) 22.885, 3rd Generation Partnership Project (3GPP), 12 2015. Version 14.0.0.
- [2] 3GPP. Study on LTE-based V2X services. Technical Report (TR) 36.885, 3rd Generation Partnership Project (3GPP), 07 2016. Version 14.0.0.
- [3] K. Abboud, H. A. Omar, and W. Zhuang. Interworking of dsrc and cellular network technologies for v2x communications: A survey. *IEEE Transactions on Vehicular Technology*, 65(12):9457–9470, Dec 2016.
- [4] AM Abdelgader and Wu Lenan. The physical layer of the ieee 802.11 p wave communication standard: the specifications and challenges. In *Proceedings of the world congress on engineering and computer science*, volume 2, page 71, 2014.
- [5] 5G Automotive Association et al. The case for cellular v2x for safety and cooperative driving. *white paper (undated, apparently 23 November 2016)(available from: <http://5gaa.org/pdfs/5GAA-whitepaper-23-Nov-2016.pdf>, 2016.*
- [6] Saeed Asadi Bagloee, Madjid Tavana, Mohsen Asadi, and Tracey Oliver. Autonomous vehicles: challenges, opportunities, and future implications for transportation policies. *Journal of Modern Transportation*, 24(4):284–303, Dec 2016.
- [7] LJ Blincoe, TR Miller, Eduard Zaloshnja, and Bruce A Lawrence. The economic and societal impact of motor vehicle crashes, 2010.(revised)(report no. dot hs 812 013). *Washington, DC: National Highway Traffic Safety Administration*, 2015.
- [8] European Commission. *Annual Accident Report*. European Commission, Directorate General for Transport, June 2017.
- [9] L. Kong, M. K. Khan, F. Wu, G. Chen, and P. Zeng. Millimeter-wave wireless communications for iot-cloud supported autonomous vehicles: Overview, design, and challenges. *IEEE Communications Magazine*, 55(1):62–68, January 2017.
- [10] Kamaljit I Lakhtaria. *Technological Advancements and Applications in Mobile Ad-Hoc Networks: Research Trends: Research Trends*. IGI Global, 2012.

- [11] 3GPP The Mobile Broadband Standard. About 3gpp organization. [Online]. Available: <https://www.3gpp.org/about-3gpp/about-3gpp>, 2019. [Accessed: 26-Mar-2019].
- [12] C. R. Storck and F. Duarte-Figueiredo. 5g v2x ecosystem providing entertainment on board using mm wave communications. In *2018 IEEE 10th Latin-American Conference on Communications (LATINCOM)*, pages 1–6, Nov 2018.
- [13] Behrad Toghi, Md Saifuddin, Hossein Nourkhiz Mahjoub, MO Mughal, Yaser P Fallah, Jayanthi Rao, and Sushanta Das. Multiple access in cellular v2x: Performance analysis in highly congested vehicular networks. In *2018 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE, 2018.
- [14] Xuyu Wang, Shiwen Mao, and Michelle X Gong. An overview of 3gpp cellular vehicle-to-everything standards. *GetMobile: Mobile Computing and Communications*, 21(3):19–25, 2017.
- [15] Christian Weiß. V2x communication in europe – from research projects towards standardization and field testing of vehicle communication technology. *Computer Networks*, 55(14):3103 – 3119, 2011. Deploying vehicle-2-x communication.
- [16] Sherali Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan. Vehicular ad hoc networks (vanets): status, results, and challenges. *Telecommunication Systems*, 50(4):217–241, Aug 2012.

User engagement with public displays

Antero Vaarnamo

antero.vaarnamo@aalto.fi

Tutor: Maria Montoya-Freire

Abstract

Public displays offer an efficient and increasingly common instrument to confer information to the general population in many areas of the society. Despite their broad deployment, there are many problems associated with public displays regarding to user engagement and interaction. Widely recognized problems such as display blindness, interaction blindness and display avoidance hinder the potential of displays to engage users, and further research into increasing user interaction with displays is required to mitigate these problems. This paper examines several approaches for combatting these problems using external devices. Both the advantages and disadvantages of the reviewed approaches are investigated and analyzed.

***KEYWORDS:** public displays, user engagement, display blindness, interaction blindness, display avoidance*

1 Introduction

Public displays are widely deployed in several places including offices, shopping centers and airports. They are utilized to show different information, including news, events and advertisements. However, despite

their widespread adoption, displays often fail to properly engage the audience [1]. There are several reasons behind this problem. One major issue is *display blindness*, which refers to people ignoring displays as a result of low expectations of the presented content and displays fading into the background of more interesting information. Other issues include *display avoidance* — deliberately looking away from displays to avoid information overload [2] — and *interaction blindness* — the fact that people often do not realise that displays can be interacted with [3]. Even if these obstacles to interact are overcome, a potential user may decide to avoid the display if it appears to provide no discernible value to the user [2].

Various methods have been developed to combat the user engagement difficulties affecting public displays. For example, colorful screens have been shown to increase interaction with displays [2]. Furthermore, enhanced emotional attachment generated via displaying personal photographs indicated an increase in user engagement [4]. Many of the suggested approaches only focus on the display and its content. As such they do not consider user influence or presence as a possible source for increasing public interest. This paper focuses on reviewing methods that rely on user activity and external devices to entice interaction and increase user engagement. The methods include devices that users can use to control the display and those that monitor user activity and change the content accordingly. The different methods are compared by analyzing their advantages and disadvantages.

This paper is structured as follows. Section 2 describes problems associated with public displays in interacting with users. Section 3 examines various approaches that employ external devices for tackling these problems and to increase user engagement with public displays. Section 4 discusses the advantages and disadvantages of the examined approaches. Finally, we make some concluding remarks in Section 5.

2 Problems with public displays

Numerous problems affect public displays and hinder their ability to effectively convey information to the public. Observational studies conducted by Huang et al. [5] over a decade ago suggest that displays in public are rarely looked at, and when they are, it is usually only for a short period of time. This problem was first described by Huang et al. [5] in 2008 and coined in 2009 by Müller et al. [1] as *display blindness*. Dalton et al. [6]

installed mobile eye trackers in a shopping center to observe people doing shopping to see whether display blindness still affects public displays showing more interactive content in more complex environments. They discovered that displays are looked at more frequently than predicted by earlier studies by Huang et al. [5], but that the time spent looking at them was brief, lasting only one third of a second. Of the people that glanced at the display, only a quarter gave it a second look.

Public displays equipped with touch technology are becoming increasingly more common. Despite their ubiquity, it can be difficult for the public to distinguish between interactive and non-interactive displays [7]. Most public screens are still used passively for broadcasting information or commercials and, as a result, users might overlook the interaction possibilities of displays with interactive capabilities, a problem called *interaction blindness* [2].

Another problem affecting public displays is display avoidance [2]. Kukka et al. [2] describe it as actively not looking at a display when noticing it. In a study conducted by Kukka et al. [2] on a university campus, students walking in a hallway that had their head turned toward the bulletin boards would quickly turn away their gaze once they noticed the display and would turn it back towards the board once they had passed the display. One reason for the behaviour could be to avoid unnecessary information and conserve mental energy. In addition, the authors hypothesize another theory to explain the phenomenon that relates to social interaction. Unacquainted people usually tend to avoid interaction and eye contact in public places. This is considered polite behaviour, at least, in Finland where the study was conducted. People respect each other's privacy and do not want to appear rude.

3 Approaches

Interactive displays can convey interactivity in several ways. These include external smart devices and curiosity objects, responsive elements on the display that react to user interaction using computer vision and sound stimuli. In the following, we discuss and provide examples of the different approaches utilizing external hardware designed for increasing user engagement with public displays.

3.1 User monitoring devices

3.1.1 Kinect and proxemics

Microsoft Kinect is a motion sensing device that enables users to interact with an application without the need for a physical controller. When used together with public displays, passers-by can interact with the display by entering within the field of view (FOV) of the Kinect sensor and performing hand and body gestures [8].

Müller et al. [9] have shown that user interaction with public displays can be increased by showing user silhouettes on the displays. In the study, the displays were noticed more frequently and more quickly than when using other types of user representation such as avatars. The interactivity increase can be attributed to our keen ability to recognize motion and our own image.

This finding was utilized in the study by Grace et al. [8] who used Kinect to test different interactivity cues for increasing user interaction with displays. When a user entered the FOV of the Kinect sensor, a dynamic skeletal representation of the user was displayed. This was followed by a different cue in each experiment. The highest interactivity increase was measured when a spotlight was added on top of the skeletal figure.

The users' orientation and distance from the display should be considered when trying to arouse and maintain user interest [10]. The provided content should vary depending on how engaged the user is and how the user responds to it to maximize user engagement. Proxemic interactions, which are based on proxemics zones proposed by Hall [11], refer to fine-grained reactive models that can be used to model user attention and react accordingly. In proxemic models, the space in front of the display is divided into distinct zones (Figure 1). By monitoring in which zone the user is and the orientation of the user, continuous, subtle feedback can be provided to guide the user towards the display. In addition, dimensions such as movement, identity and location of the user can be considered to further refine the reactive model [12].

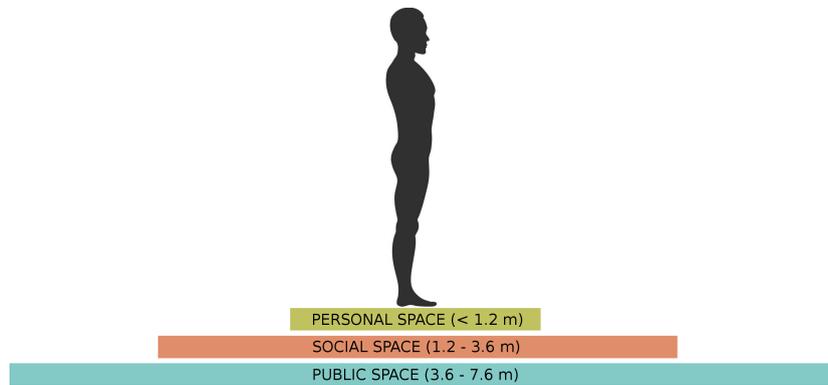


Figure 1. Interpersonal distances in Hall's proxemics zones model [11].

Proxemics has already been incorporated in commercial soft drink vending machines in Japan [12]. An infrared proximity sensor detects a nearby, potential user, after which the advertisement mode changes from remote to proximity mode. When a person stands in front of the vending machine, it recommends a product based on the approximated age and gender of the person with the help of face recognition technology.

Wang et al. [10] have designed Proxemic Peddler that uses proxemic interactions as well as user interaction history in engaging the user with an online marketplace application. The name and functions of the Peddler framework derive from street peddlers that monitor passers-by and tune their sales pitch according to the audience. Their goal is to capture the interest and keep the attention of the crowd over the duration of the whole pitch. When interacting with Proxemic Peddler, the user goes through several phases of interaction from looking at the display, approaching it, clicking on an item and finally making a purchase. If these steps to preferred outcome are not followed, the display provides other phases to try to recapture the user's attention.

3.1.2 Sound

Sorce et al. [13] propose a system that uses sound to counteract display blindness. Unlike displays, sound is not limited by field of view. Likewise, human auditory perception can distinguish the origin of sound down to a few degrees. Visual perception is even finer, but sound could be used to perceive paths toward further interaction or a display. In the proposed system by Sorce et al., sound would be activated only when a user is close to an interaction hotspot. This would limit noise and possibly increase user interaction through a targeted response. Sound would grab the attention of the passers-by and guide them toward the display. Kinect could be used to monitor user activity and user reaction to the sound.

3.2 User controlled devices

3.2.1 Curiosity objects

Generally, people cannot distinguish an interactive display from a non-interactive one. One way to decrease the possibility of this phenomenon called interaction blindness from manifesting itself is with the help of curiosity-provoking objects [7] (Figure 2). The goal of these devices is to passively attract people with their natural properties. The objects reveal the interactive possibilities of the display after people start to play with them, thus weakening the effect of both display blindness and interaction blindness.

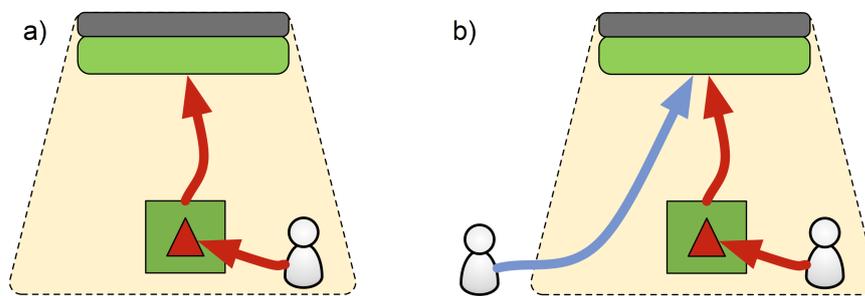


Figure 2. Interaction with display induced by a) curiosity object (primary) and b) social effects (secondary). Picture modified from [7].

Curiosity drives people to explore their surroundings and make sense of the world. Stimuli such as novelty, complexity and surprise together influence how people interact with physical objects [7]. Curiosity can be designed with these properties in mind.

Houben and Weichel conducted an experiment with and without a curiosity invoking device [7]. They found that the device attracted many participants which, after the initial interaction, moved up to the screen when its interactive properties were highlighted. Most of the people who interacted with the object also interacted with the display. The curiosity object increased the interactivity of the display significantly. The authors also noticed that as people were interacting with the display, passers-by would notice the interaction and be attracted to the display, ignoring the curiosity object (Figure 2 b). This would indicate that the device worked as intended: attracting attention to the display and not primarily on itself.

3.2.2 *Smart devices*

One compelling reason for utilizing smart devices such as mobile phones and tablets for interacting with displays is that they are ubiquitous; almost everyone possesses one and is familiar with it. No additional hardware is needed for adding interactivity to the display. Smart devices are equipped with remote, wireless communication and a broad range of sensors and actuators that provide possibilities for many use cases [14]. Smart devices constitute an easy and obvious way for users to communicate with displays.

Smart devices can be used in conjunction with public displays in numerous ways [15]. A smart device can couple with a display and control it with the built-in sensors of the device such as magnetometer and accelerometer [14]. Smartphones can act as remote controllers taking various forms depending on the device hardware. The touch screen or keyboard of the phone could be used as the input mechanism [15]. In addition, the flashlight of the phone can work as a pointing device [16].

Another use case for the smart devices is acting as a second screen [14]. For example, some museums have smartphone applications that grant users access to more information about the artifacts displayed on screen. Similarly, phones can be used as input devices for publishing or streaming content on a display or they could work both ways providing feedback to each other. The phone screen size is limited, but it can be increased by connecting to a large, external co-display, providing enhanced viewing experience.

Smart devices are not limited to smartphones and tablets (Figure 3). Smart device screen sizes have shrunk as wearable devices such as smartwatches and smart glasses have emerged. The difference in screen sizes limits the degree of freedom in interaction design since some of the applications designed for portable smartphones and tablets would not work as intended if used with smaller screen devices [14]. Portable and wearable smart devices have different strengths and weaknesses. Portable smartphones are more accessible than the more intuitive, wearable smart devices. Different use cases should target different smart devices or the smart devices could work in co-operation. For example, in Watchconnect [17], the smartwatch is responsible for gesture motion and the smartphone for storing data.

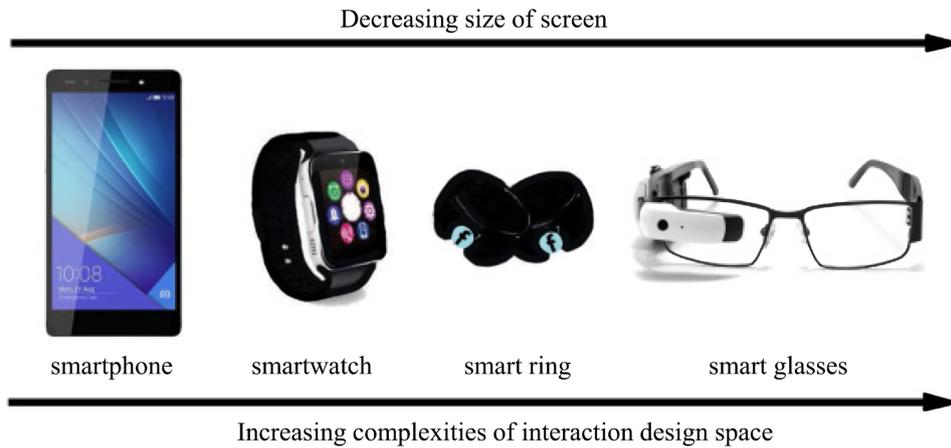


Figure 3. Different smart device screen sizes limits compatibility and design space [14].

4 Discussion

In this paper, we have presented several approaches relying on external hardware for reducing display blindness, interaction blindness and display avoidance. All of these approaches have their own advantages and disadvantages, and no single approach alone is enough to remove the user engagement issues inherent to public displays. Table 1 presents a comparison of the different approaches in terms of how well they reduce the effects of the user engagement issues with public displays. Looking at the table, it might appear that the presented approaches offer no help in combatting display avoidance. However, this is not necessarily true. The articles cited in this paper, many of them influential and highly cited, have focused on studying the effects that their research approach has on display blindness and interaction blindness. Display avoidance is a more recent and less known issue than display blindness and interaction blindness and has consequently received far less attention and focus in research. Conducting research to study the effects of display avoidance is challenging since the reasons for it are not as clear and well understood as they are for the other two issues. More research is needed to better understand display avoidance and how its effects could be diminished.

	Display blindness	Interaction blindness	Display avoidance
Kinect & Proxemics	*	**	—
Sound	*	—	—
Smart devices	—	*	—
Curiosity objects	**	**	—

Table 1. External devices effect on user engagement issues with public displays. (— No effect or effect unknown, * Slight decrease, ** Moderate decrease).

User monitoring devices like Kinect can sense people’s body gestures within its FOV. Interaction with Kinect requires that the user stays within a few meters away from the sensor in a narrow sector, which considerably limits the operation potential of this approach. However, many user interactions can be captured in this limited space such as location, movement, distance, orientation and identity. User engagement with the display can be increased through varied, personalized and targeted response using sophisticated proxemics models. Mirroring user motion is a powerful way to entice user interaction. Nevertheless, this enhanced interaction is only possible within close proximity to the display, and display blindness remains an issue.

Sound does not have the limitation of Kinect nor does it require the user to even to notice the display before it can start to attract the interest of the user. Humans can perceive the source of sound relatively accurately. Sound could be used to guide the user towards the display, thus decreasing display blindness. If multiple speakers are utilized to form a surround system together with motion capture, sound could be used for enticing desired interaction. On its own, though, sound is only helpful for attracting users from afar and even that potential is questionable. In crowded places, different sound sources will likely interfere, reducing the guiding capability of this technique. Furthermore, the volume has to be kept relatively low to minimise noise, further diminishing the effectiveness.

Compared to human-screen interaction (HSI), where the interaction with the screen is enabled by touch-sensitive screens or cameras with motion-sensing capabilities, using smart devices requires no additional hardware [14]. Instead, users are required to use their own device to interact with the display. This might be unacceptable for the elderly who do not always own high-tech devices, and for this reason alone, the use case of the display should be carefully considered. HSI is spontaneous since there is no

need for users to own a device to interact with the screen. With smart devices, interaction happens remotely from long distances with wireless communication, whereas HSI limits the interaction range to in front of the screen. However, smartphones do little to decrease display blindness or display avoidance and mainly enable easier interaction with the display once the user is sufficiently engaged with the content.

Curiosity objects attract people to them and afterwards highlight options for interacting with a nearby display. People are drawn by their own curiosity to interact with these strange devices to understand their purpose and function. These objects significantly diminish the effects of display blindness and interaction blindness as Houben et al. [7] have shown. However, they still require the user to first discover them, find them interesting and approach them to serve their purpose.

An effective approach would likely incorporate several of these aforementioned techniques into a single one. Sound could be used to attract people from afar to within visible range of the display. Curiosity object or Kinect would then further entice interaction and guide the users toward the display. The users could then use their own mobile phone or body gestures to interact with the display.

5 Conclusion

Public displays are widely utilized in the society to confer information to the masses. Nevertheless, several problems plague these displays and user engagement with the displays is not optimal. This paper examined and analyzed the effectiveness of different approaches based on external devices for mitigating these problems. All of the approaches have their advantages and disadvantages, and further research is needed to find a solution that resolves all the issues.

References

- [1] J. Müller, D. Wilmsmann, J. Exeler, M. Buzeck, A. Schmidt, T. Jay, and A. Krüger, "Display blindness: The effect of expectations on attention towards digital signage," *International Conference on Pervasive Computing* pp. 1-8, May 2009.
- [2] H. Kukka, H. Oja, V. Kostakos, J. Goncalves and T. Ojala, "What makes you click: Exploring visual signals to entice interaction on public displays," *Conference on Human Factors in Computing Systems - Proceedings*, pp. 1699-

1708, May 2013.

- [3] T. Ojala, V. Kostakos, H. Kukka, T. Heikkinen, T. Lindén, M. Jurmu, S. Hosio, F. Kruger and D. Zanni, "Multipurpose Interactive Public Displays in the Wild: Three Years Later," *textitComputer*, vol. 45, pp. 42-49, May 2012.
- [4] K. Lee, S. Clinch, C. Winstanley and N. Davies, "I love my display: Combatting display blindness with emotional attachment," *Proceedings of The International Symposium on Pervasive Displays*, pp. 154-159, Jun. 2014.
- [5] E. M. Huang, A. Koster and J. Borchers, "Overcoming Assumptions and Uncovering Practices: When Does the Public Really Look at Public Displays?," *Proceedings of the 6th International Conference on Pervasive Computing*, pp. 228-243, May 2008.
- [6] N. S. Dalton, E. Collins, P. Marshall, "Display Blindness? Looking Again at the Visibility of Situated Displays using Eye Tracking," *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 3889-3898, Apr. 2015.
- [7] S. Houben and C. Weichel, "Overcoming Interaction Blindness through Curiosity Objects," *Conference on Human Factors in Computing Systems*, pp.1539-1544, Apr. 2013.
- [8] K. Grace, R. Wasinger, C. Ackad, A. Collins, O. Dawson, R. Gluga, J. Kay and M. Tomitsch, "Conveying Interactivity at an Interactive Public Information," *Proceedings of the 2nd ACM International Symposium on Pervasive Displays*, pp. 19-24, Jun. 2013.
- [9] J. Müller, R. Walter, G. Bailly, M. Nischt and F. Alt, "Looking Glass: A Field Study on Noticing Interactivity of a Shop Window," *Conference on Human Factors in Computing Systems - Proceedings*, pp. 297-306, May 2012.
- [10] M. Wang, S. Boring and S. Greenberg, "Proxemic peddler: a public advertising display that captures and preserves the attention of a passerby," *Proceedings of the 2012 international symposium on pervasive displays*, p. 3-6, Jun. 2012.
- [11] E.T. Hall, "The Hidden Dimension," Anchor. 1966.
- [12] Y. Taniguchi, "Content scheduling and adaptation for networked and context-Aware digital signage: A literature survey," *ITE Transactions on Media Technology and Applications*, vol. 6, pp. 18-29, 2018.
- [13] S. Sorce, V. Gentile and D. Rocchesso, "Ecological Invitation to Engage with Public Displays," *Proceedings of the 7th ACM International Symposium on Pervasive Displays*, Jun. 2018.
- [14] P. C. Ng, K. E. Jeon and M. Baldauf, "When Smart Devices Interact With Pervasive Screens: A Survey," *Communications and Applications*, vol.13, Oct. 2017
- [15] J. Patterson and S. Clinch, "SlideTalk: Encouraging User Engagement with Slideshow Displays," *Proceedings of the 7th ACM International Symposium on Pervasive Displays*, Jun. 2018

- [16] S. Clinch, "Smartphones and Pervasive Public Displays," *IEEE Pervasive Computing*, vol. 12, pp. 92-95, 2013
- [17] S. Houben and N. Marquardt. "Watchconnect: A toolkit for prototyping smartwatch-centric cross-device applications," *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 1247-1256, 2015.

Mobile Edge Computing Assisted Internet of Things

Shamim Biswas

shamim.biswas@aalto.fi

Tutor: Abbas Mehrabidavoodabadi

Abstract

Mobile Cloud Computing (MCC) is the primary architecture for modern applications, especially for consumer services. MCC relies on a central infrastructure for almost all crucial functions, such as intensive data processing, data collection and analytics. This obviously limits the capabilities of the system. It introduces a single point of failure in the system and, most importantly, increases latency for the connected devices. Mobile Edge Computing (MEC) is proposed as an ideal candidate to overcome these limitations of traditional computation models in recent years. MEC has many capabilities which are ideal for modern networks, including the Internet of Things (IoT). Research in IoT has increased significantly in recent years. Many new applications for IoT are also discovered. They have been shown to be suitable for consumer products, industrial sensing, smart cities, remote monitoring, healthcare and vehicle to everything (V2X) networks. In this survey, we examine the current development state of MEC technology tailored towards IoT applications. We explore the state of the art IoT network applications. We discuss the technologies enabling MEC and some challenges towards achieving it. Finally, we attempt to forecast new developments and the future direction of MEC and IoT research.

KEYWORDS: *Mobile edge computing (MEC), Internet of Things (IoT), software defined networking (SDN), virtualization, latency.*

1 Introduction

Mobile cloud computing paradigm has been the dominating architecture in recent years. It uses a simplistic approach of backhauling most tasks to a single point in the network. Various methods, such as containerization and replication, have been developed to alleviate the problems associated with having a single service point. However, these methods do not solve the problems of high latency and network congestion [10]. Hence, they underperform compared to modern distributed methods, such as Mobile Edge Computing (MEC).

Mobile edge computing is an emerging technology that enables the distributed implementation of the computing and communication by moving resources to the edges of the radio access network [11]. Thus, it reduces the distance between end devices and the servers. However, this implies that network portals, such as wifi routers and mobile base stations, need to be transformed into components with significant computational and storage capabilities.

Despite the need to modify the network devices, MEC brings tremendous benefits, for example, faster response times. Thus, MEC is used where time critical application are important. MEC reduces the over-all network traffic and thus reduces congestion and increases energy efficiency [1]. The mobility of edge resources improves dynamic allocation of resources. If a region has a spike in computation usage, more resources can simply be moved closer to the site [18].

In recent years, IoT have become very popular and are deployed in large numbers. The European Commission has estimated the number of IoT devices to reach 100 billion by 2020 [12]. These devices generate a huge amount of data at a fast velocity. Processing this data is a challenge using traditional MCC technology because of their single service point architecture.

When MEC is used to serve IoT networks numerous applications and optimizations become feasible for both development and research. IoTs backed by MEC are found to be suitable for AR/VR, healthcare, smart grids, V2X, video QoS and other time-sensitive and bandwidth-intensive services [9].

However, this setup is more complex and challenging to implement. It involves deploying computation resources close to network edge. Additionally, new algorithms and network intelligence are required to efficiently route traffic. Network virtualization technologies like Software Defined Networking (SDN) and Network Function Virtualization (NFV) ease the creation and configuration of such networks using only general purpose servers [8].

The rest of the paper is organized as follows: Section 2 introduces the Internet of Things (IoT) and analyses its current trends and challenges. Section 3 explores the emerging ICT technologies, i.e., MEC and compares it with similar technologies. Additionally, it discusses critical technological advancements which enable MEC for IoT development. Section 4 presents the emerging IoT applications enabled by MEC. Section 5 analyses the benefits, challenges and the research direction for MEC assisted IoT systems. Finally, section 6 summarizes the current state of MEC assisted IoT development and concludes the paper.

2 Internet of Things (IoT)

The Internet of Things is regarded as the next phase of internet technology development. In the 1990's, with the dawn of internet, technology was used for data, for instance, for accounting and banking and storing personal data, such as blogs and pictures [4]. In the 2000's, internet concerned people, social media, and personal branding. Mobile devices became important identities of people. In 2020's, internet is expected to concern all physical things such as doors, appliances, vehicles and even cities. They all communicate with each other either directly or through a human interface. Therefore, IoT is the technology which connects everyday things to the internet providing novel opportunities for sensing and intelligence.

The realization of IoT depends on the development of system on chip devices which are small enough to be embedded in everyday objects. They must be equipped with a suitable radio interface and must be energy efficient to be able to run on batteries for long periods. Thus, modern IoT is characterised by low memory, low powered and single chip devices quipped with a radio interface designed for IoTs such as WiFi, NB-IoT or zigbee. Although technologies with these capabilities are far from perfect, the popularity of IoT implementation in industrial settings have exponen-

tially grown in recent years.

Devices

IoTs are used in consumer space as well, for example, smart lights, smart watches, kid cams, and other devices. Consumer IoTs are characterised by easier off the shelf deployability. They require shorter range of wireless communication, usually wifi. One example would be healthcare, where smart watches monitor the heart rate, blood-pressure, calorie consumption, sleep cycle and signs of depression. Smart watches have been shown effective in fall detection of the wearer and thus, can notify emergency medical services if necessary [3]. Other devices include, developer friendly Raspberry Pi and industry specific specialized IoTs.

Connectivity

IoT use cases put an unpredictable and fluctuating demand on the network. They require scalability, QoS and robustness in connectivity. Additionally, IoT performance requirements include large coverage, low battery consumption, low CPU cycles and scalability. Thus, specialized radio protocols have been developed which better satisfy these requirements, such as Narrowband-IoT (NB-IoT). NB-IoT uses unused radio spectrum or guard bands in an efficient manner to dynamically allocate network resources [15]. Other communication protocols suitable for IoT include Zigbee, CoAP, XMPP and MQTT [2].

Challenges

The IoT ecosystem is changing rapidly, making it chaotic for any attempt at protocol standardization. Standards for device manufacturers do not exist, resulting in a heterogeneous system of devices. IoT also face other challenges regarding privacy and security of the data they process. Security design towards IoT has not developed enough to reliably transmit data. [6]

3 Emerging ICT Technologies

3.1 Mobile Edge Computing (MEC)

The European Telecommunications Standards Institute (ETSI) MEC Industry Specification Group (ISG) is the standardization organization re-

sponsible for creating a open and collaborative environment for MEC development. In 2015, they published the foundation specifications on technical requirements and reference architecture for MEC [16]. In the specification, the definition of Edge encompassed both the base stations themselves and data centers close to the radio network [11].

According to ETSI, MEC is defined as [16] : “Mobile edge computing provides an IT service environment and cloud computing capabilities at the edge of the mobile network, within the radio access network (RAN) and in close proximity to mobile subscribers.”

Traditionally, Mobile Cloud Computing (MCC) is used with IoTs for a great variety of applications. Although this approach is easy to implement, i.e., the IoT device just needs to be connected to the internet, some applications have no alternative but to use mobile edge computing techniques. These applications are those which require low latency communication and fast processing.

Mobile Edge Computing (MEC) resolves many of the limitations of MCC assisted IoT network. MEC was introduced to serve time sensitive and low latency applications of IoT. Studies show that average round trip time for a request by a IoT using MCC ranges from 100ms to few seconds [10]. Such latencies are unfeasible for many use cases. In a Vehicle-to-Vehicle network (V2V), cars are moving at high speeds, and therefore, much faster decision making capabilities are required. An average MEC implementation has a response time of less than 20ms, making it a viable solution for V2V [10].

3.2 MEC Compared To Other Computing Techniques

MCC is limited in providing fast responses because the physical location of the processing service is far from the data source. Thus, new computational models are required for offloading computationally intensive tasks. MEC provides two advantages in this use case. Firstly, it distributes the load among local RAN instead of a single cloud server. Secondly, it returns the results faster to the end device due its close proximity. Table 1 shows a comparison between MEC and MCC.

This ability allocate dynamic resources closer to the source of data generation is what differentiates MEC from Content Distribution Networks (CDN). CDN merely cache static data closer to the end devices and have no computation abilities to analyze the data it serves.

There are several other buzzwords surrounding MEC, for example, Fog

Criteria	MCC	MEC
Initial promotion	Aepona (2010)	ETSI (2014)
Node location	Anywhere in the Cloud	RAN edge or Base Station
Service accessibility	Via internet	Closest edge device
Latency	High	Low
Point of failure	Single	Robust
Computational complexity	High	Low
Communication complexity	High	Low
Storage capacity	High	Limited
Relevance to IoT	Low	High

Table 1. Comparison of MEC and MCC. Based on table in [9]

Computing. In the past few years, both MEC and Fog Computing have been used interchangeably even though there is a slight difference. Both of them are similar in principle, i.e., they bring the computation resources closer to the end device. However, they differ in where the computation capability is placed. MEC puts computational intelligence directly into edge gateway which are devices such as programmable automation controllers (PACs). On the other hand, Fog Computing puts computing resource at the local area network level often in IoT gateway or fog node [9].

3.3 Network Function Virtualization (NFV) and Software Defined Network (SDN)

Currently, several enabling technologies for MEC are actively researched, such as Network Function Virtualization (NFV) and Software Defined Networks (SDN). Both are in active development at ETSI [8] and Open Networking Foundation (ONF) respectively. The Network Function Virtualization (NFV) is a standard to exploit agility enhancement and maintenance automation of virtual network services. Earlier, networks required numerous specialized hardware including switches, bridges and gateways to effectively manage traffic. NFV enables to virtualize these services dynamically on standard servers. They provide dynamic and flexible infrastructure services which accelerate innovation.

Software Defined Network (SDN) complements NFV in implementing MEC. It allows interconnecting virtual IT resources with virtual or physical network resources [8]. SDN broadly has two modules, namely, control plane and data plane. Control plane provides a unified view of the private

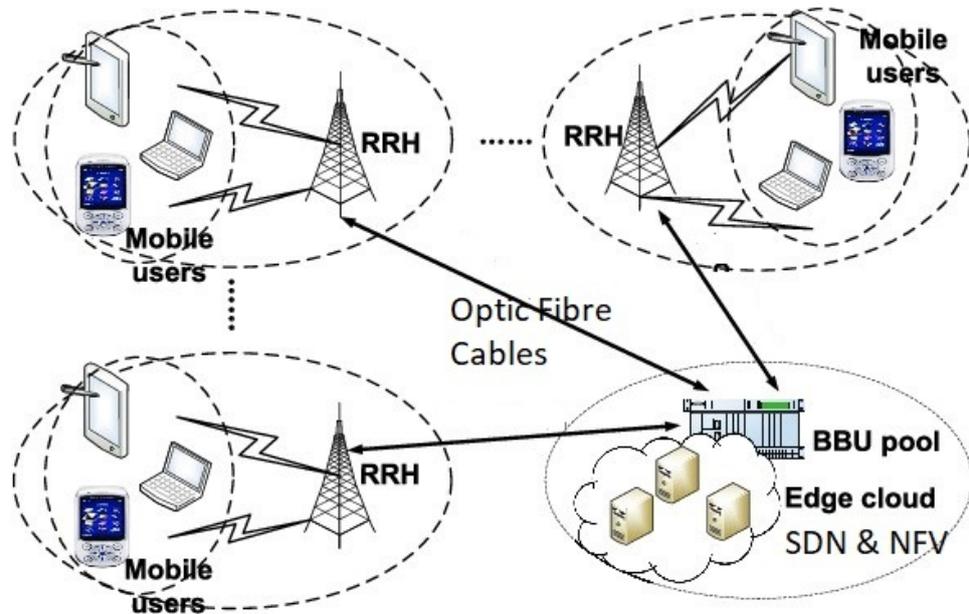


Figure 1. MEC implementation at C-RAN with SDN and NFV. Reused from [14]

network which may be spread across the internet. This allows for intelligent decision making for congestion control and easy management. The data plane handles the transfer of data to immediate peer devices both virtual and physical. The prime advantage of SDN is the separation of the control plane and data plane which simplifies network management.

Network Function Virtualization and Software Defined Networking are two technologies which enable scalable implementation of MEC. These technologies work in tandem to virtualize the infrastructure required for MEC. These technologies reduce both installation and operation costs of the network. Centralized Radio Access Network (C-RAN) is a perfect example which demonstrates the advantages of NFV and SDN. In a C-RAN, instead of deploying individual MEC servers at the edge, a group of radio terminals can be served by a single server running virtual servers and connected via SDN. This server can connect to the radio edge from a distance up to 20km with fibre optic cables [13]. Figure 1 shows MEC implementation at C-RAN with SDN and NFV technology.

4 IoT use cases enabled by MEC

AR/VR

Augmented Reality and Virtual Reality require UltraHD 360-degree video which is simply not possible with the fastest mobile broadband connec-

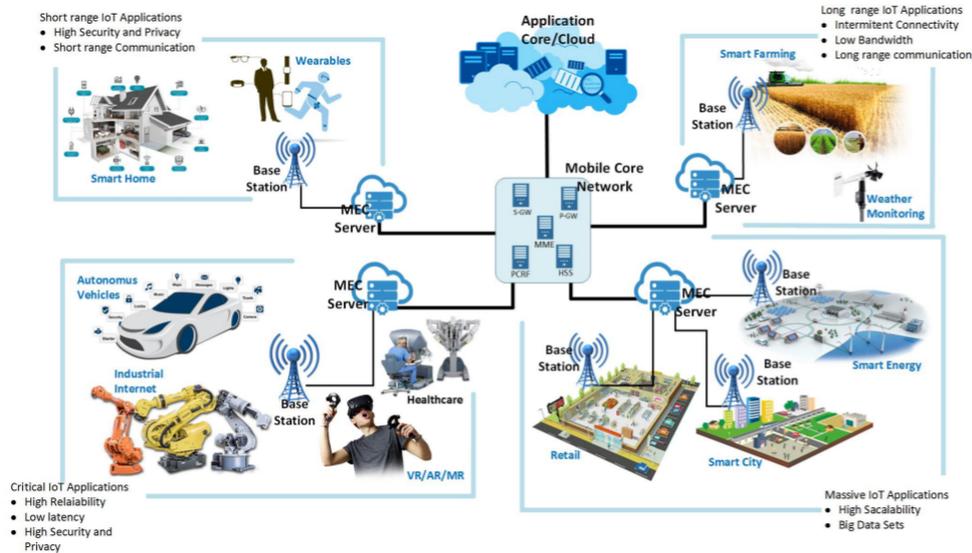


Figure 2. IoT applications enabled by MEC. Reused from [9]

tions. They require a bandwidth of over 1 Gbps and ultra-low latency. MEC implemented by 5G seems the only solution for implementing these systems. AR/VR systems are especially important in retail, gaming and defence industries. Figure 2 shows the various IoT applications enabled by MEC including AR/VR.

Video Analytics and Surveillance

According to Cisco, 70% of internet traffic in the 2020s will be videos. MECs provide adequate bandwidth and realtime-latency for use in video streaming applications. With a MEC backed IoT network, an intelligent surveillance system can detect anomalies at the edge itself and only send the parts of the video stream which are of interest to the central cloud or human inspection centre [11]. A video capture device on a traffic signal can detect accidents, congestion and even weather conditions and aid decision making for a city traffic controller. This approach saves storage space by storing only relevant video streams and saves human effort in manually investigating all the videos.

V2X

IoT with MEC is a strong candidate for enabling future self-driving cars. This is due to its applicability in a low latency V2X communication network. Connected vehicles are crucial for autonomous driving systems [11]. Suppose, a car changes lane to avoid an obstacle, then the car behind it has no way of knowing the obstacle in front. A V2X system can detect such instances and alert the driver in the last car to slow down. In some

cases, car sensors alone cannot detect danger and only collective intelligence can infer an impending collision. Consider another example, a car signals to change lanes and another car is coming fast on the lane but is too far away for the sensors to detect the oncoming traffic. A proximity tower can combine these two signal and alert both the drivers to take preventive measures.

Farming

IoT and connected drones hold huge potential for the farming industry. Sensors in the soil can send real-time data about humidity and nitrate concentrations for ensuring crop health. Drones with imaging devices have been used to count and estimate the final production output [9].

Healthcare

MEC assisted IoT devices may transform healthcare due to their ubiquity. Wearable IoT devices which collect biological information about the wearer is already in use. Indicators like pulse rate, blood pressure and insulin level can be continuously monitored [9]. Furthermore, intelligence at the edge can automatically detect anomalies in these indicators and alert medical professionals. IoTs may also be used for fall detection of elderly [1], remote surgery and remote consultation.

Smart Grid

Smart Grid is a network of smart electrical meters deployed at different geographical locations in an urban setting. These devices communicate with each other and gather electricity usage patterns from the electric grid by houses, services and vehicles. This creates an intelligent system which efficiently schedules power distribution. N. Kumar et. al. have proposed delay tolerant vehicular networks [5] for smart grid management.

Smart Homes

Smart home appliances are the most recognisable form of IoTs for consumers. Major tech companies market various devices for smart home management, such as Google Nest and Amazon Echo. These devices produce huge amounts of data related to the house, such as temperature, humidity and air quality. Similarly, doors and surveillance cameras in a smart office environment can be monitored by MEC assisted IoT devices [9].

Industrial IoT

Specialized IoTs used in the industrial setting are called Industrial IoT (IIoT). For instance, IIoT is used in mining. IoT spread throughout the tunnels could detect harmful gas buildup in the chambers and alert the staff before any mishap. IIoT can be used in assembly line robotics. Both the scenarios are critical for business and MEC assistance for fast and robust connectivity is essential.

5 Discussion

Benefits of MEC assisted IoT

MEC assisted IoT network has several advantages over a MCC assisted network. One major benefit is the ability of computational offloading. IoTs can decide to offload task to the edge server for conserving its battery power. Alternatively, it could do the computation locally when a faster response is required. Low network latency is the prime reason for implementing MEC with IoT. Current implementations of MEC cut the round trip time to about one-tenth the time of traditional systems. MEC assisted IoT, henceforth denoted as MEC-IoT, is energy efficient at both the User Equipment (UE) and the server. This is due to the simpler computation and network architecture. Furthermore, MEC-IoT reduces storage requirements at the central cloud because it can preprocess data before sending it the central cloud services. MEC-IoT is geographically distributed making it robust to single site failure and power outages. It is much more dynamic due to device mobility of both UE and servers [10]. Virtual servers can be easily migrated due to containerization and virtual networking.

Challenges to MEC-IoT

However, there are some challenges to implementing large scale MEC-IoT as well. The 2016 Dyn DNS denial of service attack underlines the importance of having robust security features in IoT devices. Millions of IoT devices were compromised and exploited to attack a single service on the internet. As the number of IoT grows exponentially such attacks could bring down the whole internet.

MEC-IoT implementation increases stress on the developer to modify their applications to a distributed architecture to fully utilize the features of the system. A lack of research in this field only degrades MEC-IOT

deployment possibilities in real life.

There is limited knowledge available to securely deploy and connect IoT devices which have very limited or no input and output interfaces for configuration. Small IoT lack user interfaces and remote configuration of bulk of devices is difficult. IoT device deployment and management is challenging because of large quantities and small size. Companies after deploying IoTs at a site could easily lose track of some the devices.

Future Trends and Research Direction

Use of MEC for consumer IoTs has been limited. Although we already have consumer IoT devices, such as Google Next and Amazon Echo, they communicate through a centralised cloud infrastructure. In coming years, we can expect to see MEC for consumer IoT in the form of wifi routers with computational capacity. Mobile network operators may put dynamic application servers at the base station or public vehicles may become a smart grid of computing resource. Edge computing servers are recently made available in the market by Lenovo and programmable IoT devices are launched by Intel. The success of MEC-IoT depends on industry interest, investment by private companies and governments, and the public acceptability to such pervasive technology in their lives.

Furthermore, several novel kinds of research are directed towards implementing feasible and secure MEC. Special algorithms for collaborative edge caching have been developed which optimize cache hits [17]. A novel method to keep data secret between IoT and cloud server is found in [7]. In [18], the authors have suggested the use of UAVs as IoT gateways for dynamic resource allocation and mobility of the supporting infrastructure. This idea can easily be extended to terrestrial vehicles [5]. For instance, public bus transport can act as a network of mobile gateway points for nearby IoT devices. This eliminates the need to choose strategic places to install gateways which are close to the roads.

One perspective not discussed in this paper is the 5G, which is a major enabler of MEC and IoTs. 5G is the next generation of mobile communication which uses very short waves for faster communication. The short wave radio communication of 5G makes it a line of sight communication and thus requires many small antennas or base stations which is ideal for placing MEC nodes.

6 Conclusion

MEC is a promising technology which may transform how the internet is used. MEC overcomes all the major shortcomings of the traditional computing systems when applied towards IoTs. It is especially suitable for time-sensitive applications. However, despite all the research effort and funding towards MEC and IoT infrastructure, it is still unclear whether actual implementation of these would be viable for companies, convenient for the users and acceptable by the people. Additionally, novel software abstraction and algorithms are required to run traditional software on MEE-IoTs. The vast number of research papers published in this field indicates that industry and academia outlook is positive towards MEC-IoT but no quantitative studies are found in the literature. The success of MEC-IoT would depend on the technological advancements of its supporting technologies, industry investment and its reception by the consumers.

References

- [1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie. Mobile edge computing: A survey. *IEEE Internet of Things Journal*, 5(1):450–465, Feb 2018.
- [2] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, and M. Mohammadi. Toward better horizontal integration among iot services. *IEEE Communications Magazine*, 53(9):72–79, Sep. 2015.
- [3] Eduardo Casilari and Miguel A. Oviedo-Jiménez. Automatic fall detection system based on the combined use of a smartphone and a smartwatch. *PLOS ONE*, 10(11):1–11, 11 2015.
- [4] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645 – 1660, 2013. Including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond.
- [5] N. Kumar, S. Zeadally, and J. J. P. C. Rodrigues. Vehicular delay-tolerant networks for smart grid data management using mobile edge computing. *IEEE Communications Magazine*, 54(10):60–66, October 2016.
- [6] LEE. The internet of things (iot): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431 – 440, 2015.
- [7] Xiong Li, Shanpeng Liu, Fan Wu, Saru Kumari, and Joel Rodrigues. Privacy preserving data aggregation scheme for mobile edge computing assisted iot applications. *IEEE Internet of Things Journal*, PP:1–1, 10 2018.
- [8] N. Omnes, M. Bouillon, G. Fromentoux, and O. L. Grand. A programmable and virtualized network and its infrastructure for the internet of things:

How can nfv and sdn help for facing the upcoming challenges. In *2015 18th International Conference on Intelligence in Next Generation Networks*, pages 64–69, Feb 2015.

- [9] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb. Survey on multi-access edge computing for internet of things realization. *IEEE Communications Surveys Tutorials*, 20(4):2961–2991, Fourthquarter 2018.
- [10] G. Premsankar, M. Di Francesco, and T. Taleb. Edge computing for the internet of things: A case study. *IEEE Internet of Things Journal*, 5(2):1275–1284, April 2018.
- [11] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust. Mobile-edge computing architecture: The role of mec in the internet of things. *IEEE Consumer Electronics Magazine*, 5(4):84–91, Oct 2016.
- [12] Xiang Sun and Nirwan Ansari. Edgeiot: Mobile edge computing for the internet of things. *IEEE Communications Magazine*, 54:22–29, 12 2016.
- [13] G. C. Valastro, D. Panno, and S. Riolo. A sdn/nfv based c-ran architecture for 5g mobile networks. In *2018 International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, pages 1–8, June 2018.
- [14] X. Wang, K. Wang, S. Wu, S. Di, H. Jin, K. Yang, and S. Ou. Dynamic resource scheduling in mobile edge cloud with cloud radio access network. *IEEE Transactions on Parallel and Distributed Systems*, 29(11):2429–2445, Nov 2018.
- [15] Y.-P. Eric Wang, Xingqin Lin, Ansuman Adhikary, Asbjörn Grövlén, Yutao Sui, Yufei W. Blankenship, Johan Bergman, and Hazhir Shokri-Razaghi. A primer on 3gpp narrowband internet of things (nb-iot). *CoRR*, abs/1606.04171, 2016.
- [16] D. Sabella N. Sprecher Y. C. Hu, M. Patel and V. Young. Mobile edge computing—a key technology towards 5g, etsi, sophia antipolis, france. 11, 2015.
- [17] X. Zhao, P. Yuan, H. li, and S. Tang. Collaborative edge caching in context-aware device-to-device networks. *IEEE Transactions on Vehicular Technology*, 67(10):9583–9596, Oct 2018.
- [18] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong. An air-ground integration approach for mobile edge computing in iot. *IEEE Communications Magazine*, 56(8):40–47, August 2018.

Machine Learning for Video Encoding

Hongkuan Wang

hongkuan.wang@aalto.fi

Tutor: Gazi Illahi

Abstract

Machine learning has drawn increasing attention in the past few decades and is employed as a solution to a wide range of video and image processing tasks. This paper presents a survey on the development of machine learning algorithms, especially neural networks, for video encoding. Several types of metrics are utilized to measure the performance of different models. The performance of the machine learning algorithms is compared with traditional image compression algorithms, which shows the potential of implementing machine learning algorithms on video and image processing tasks.

KEYWORDS: Video Encoding, Machine Learning, Image Compression, Neural Networks

1 Introduction

Thanks to the rapid progress of web technology, online content that only includes texts and images can hardly meet people's requirements anymore. Instead, video streaming is now playing an increasingly important role in our daily life not only for entertainment but also for business due to

its superior visual experience. It is mentioned that internet protocol (IP) video accounts for 75% of all IP traffic in 2017 and will increase to 82% by 2022 [1]. Although the bandwidth of networks has developed dramatically in the past decade, the quality of videos that people can attain via their phones or cameras has also improved substantially. This situation leads to the remaining issue of transferring large videos through network efficiently. The videos have to be encoded into binary format in order to be delivered over the web. During the encoding process, there is a great deal of redundant information in video files that can be removed for delivery. This removed information can be reconstructed during the decoding process of the videos, leading to acceptable loss or even no loss in the end.

As a video is essentially composed of a sequence of frames, namely, static digital images, the encoding of a video depends greatly on the compressions of these digital images. A variety of algorithms, such as discrete cosine transform (DCT), fractal encoding and chroma subsampling, have been implemented to reduce the size of the digital images, thus improving the efficiency of the video encoding [2]. However, these conventional algorithms have some bottlenecking drawbacks, preventing them from achieving high compression performance [3]. Therefore, many researchers have paid increasing attention to machine learning algorithms to explore further possibilities in video encoding. Machine learning algorithms have the potential to optimize video encoding/decoding pipelines with better performance than traditional algorithms.

This paper will review state-of-the-art machine learning methods proposed in the whole process of video encoding and suggest directions of future research.

The rest of the paper is organized as follows. Section 2 gives an introduction to basics of video encoding. Section 3 discusses the basics of machine learning methods, especially neural networks. Section 4 reviews the related work of machine learning algorithms on video compression. Section 5 draws conclusions on the problem and proposes suggestions on future research.

2 Video Encoding

2.1 Video Encoding Basics

As mentioned before, a scene in a video consists of a collection of still pictures. Some statistical analysis has pointed out that there is a significant correlation both between the neighboring pictures and within the pixel patterns in a picture. In addition, human eyes are insensitive to the loss of specific spatio-temporal visual information, which can also be exploited to remove unnecessary data in the video file [2]. If the redundant information of each image is eliminated during the compression process, the whole video file can require less space for storage and transmission. This whole process can be referred to as image compression.

Image compression can be classified into two major types, lossless and lossy compressions. The lossless compression amounts to the compression where all data can be reproduced completely after reconstruction, while in lossy compression, some data is missing permanently during compression/decompression process [4]. In most applications, lossy compression is tolerable for the sake of fast access and transmission. However, in some critical applications, only lossless compression is used. For instance, all information must remain intact in medical images in order to observe the symptoms correctly [4].

The process of image compression can be illustrated by Fig. 1. The original image I is converted into compressed data I_c after the manipulation of the coder. The number of bits contained in original file is more than the number of bits for compressed data [4]. Then the compressed data is decoded to the reconstructed image I' , which can either be completely the same as the original image for lossless compression or inferior in some aspects to the original image for lossy compression.

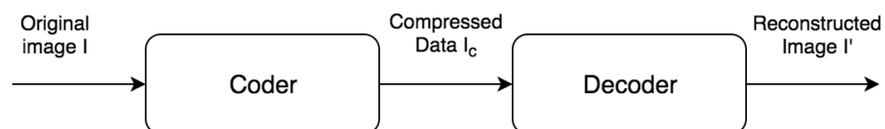


Figure 1. Process of image compression

The video encoding also includes audio encoding. However, it is not in the scope of this paper.

2.2 Metrics

When quantifying the degree of compression obtained by a specific compression scheme, a common approach is to calculate *compression ratio* of the result. The compression ratio is defined as

$$\text{Compression Ratio} = \frac{\text{Bits in original image}}{\text{Bits in compressed image}} \quad (1)$$

This exhibits a more accurate demonstration of the amount of data that is compressed during the process [5].

In addition to compression ratio specifying the quantity of compression, we also need to evaluate how well the schemes are doing in the lossy compression. Some standard metrics, such as Peak Signal to Noise Ratio (PSNR), typically serve as tools to compute the quality level of compression. The PSNR measurement is given by:

$$PSNR = 10 \log_{10} \left[\frac{(2^b - 1)}{MSE} \right] \quad (2)$$

in which b is the number of bits per pixel, and MSE is defined as follows:

$$MSE = \frac{1}{M \times N} \sum_{i=1}^M \sum_{j=1}^N (f(x, y) - g(x, y))^2 \quad (3)$$

where f and g are images of $M \times N$ size. The PSNR is calculated as the ratio of the size of the error to the peak value of the image, while Signal to Noise Ratio (SNR) is defined as the size of the error relative to the mean squared value of the image [6]. Therefore, the PSNR is more subjective qualitative measurement of quality compared to SNR. The PSNR method has the advantage of simple calculation, while it is not sufficient to define the quality of the reconstructed image since it is measured in a mathematical manner rather than perceived by a human viewer [5].

Considering the drawbacks of these metrics, it is ideal to implement a human visual system-inspired measurement which can simulate the way how humans perceive images. In that case, we can optimize the relevant parameters directly to remove the information that human eyes are not sensitive to, thus promoting the quality of reconstructed images [7]. Unfortunately, this type of measurement system has yet to be invented. Instead, various improved metrics, such as PSNR-HVS [8] and MS-SSIM [9], are exploited to evaluate the result of compression.

3 Machine Learning Algorithms

Machine learning algorithms have been developed rapidly in the past few decades as the significant power of machine learning has drawn people's attention. Neural Network is one of the major part in machine learning field and it has been designed and implemented in video encoding and image compression with considerable effort. This section introduces basics of machine learning algorithms, especially neural networks, which have created a profound impact on related areas.

3.1 Artificial Neural Networks(ANNs)

An artificial neural network is a simplified computational model inspired by biological neuron systems. It mimics the manner in which the human brain makes calculations and forms a judgement. It is highly abstracted compared to the real nervous system.

An artificial neural network consists of a large number of artificial neurons, which are also called nodes or units. An artificial neuron is basically a computational representation of a biological neuron. Similar to a biological neuron, an artificial neuron takes numerous inputs and transforms all the inputs into one output signal. This output signal can be used as the input for another neuron or as the final result of the network.

A typical artificial neuron is shown in Fig. 2. It receives n inputs (shown as x in Fig. 2) and a bias (shown as b in Fig. 2). Each input is associated with a weight (shown as w in Fig. 2) that shows how much influence the input can exert on the artificial neuron. The artificial neuron sums up all the weighted values and the bias, and transforms it via an activation function (shown as f in Fig. 2). Several common activation functions, such as sigmoid, tanh, relu, can be used in different scenarios due to their different characteristics. After the activation process, a signal (shown as Y in Fig. 2) is generated as the output of the neuron [10].

An ANN contains numerous artificial neurons arranged by layers. The adjacent neurons have tight connections between them. Each connection has the corresponding weight that extends the influence on the neurons in the next layer. Fig. 3 shows a typical structure of an ANN. The structure can be divided into three layers. The input layer includes the nodes that provide the incoming information from the outside world, which is normally the preprocessed data that we feed into the neural networks. The output layer contains the nodes that finalize the computations and

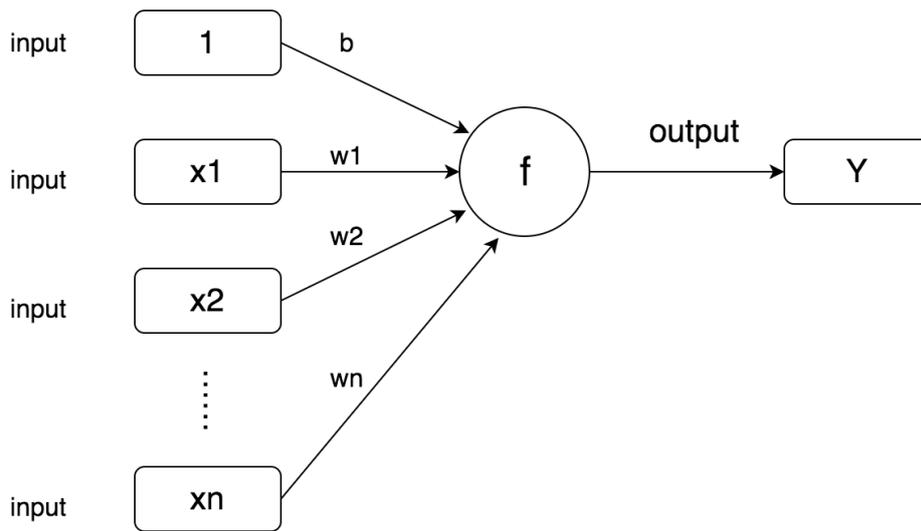


Figure 2. An artificial neuron

pass the result to the outside world. The hidden layer, which serves as the bridge between the input layer and the output layer, is invisible to the outside world. It can contain zero or multiple layers in the network.

The network shown in Fig. 3 is a feedforward neural network. The information in this network moves only in one direction, which means the output of all neurons in the networks only goes into the subsequent layer rather than the prior layer. For a feedforward neural network, the hidden layer can be empty, so the input layer connects to the output layer directly. This type of network is the simplest neural network, which is referred to as a single layer perceptron. When the ANN has multiple hidden layers of neurons, the network is classified as a multi-layer perceptron. It can also be referred to as a deep neural network. Due to several hidden layers, the multi layer perceptron has a better ability to model non-linear relationships compared to the single layer perceptron [10].

In order to make the neural network work, a tremendous amount of training datasets are necessary to be fed into the network. The way a neural network learns with the training data is called a back propagation algorithm. "Back propagation" is short for "backward propagation of errors", which means the error calculated by the activation function is "propagated" back to the preceding layer. This process can adjust the weights of the inputs accordingly. For each entry of the dataset, the weights of the neural network can be corrected from the error. The performance of the network is highly dependent on the number of training

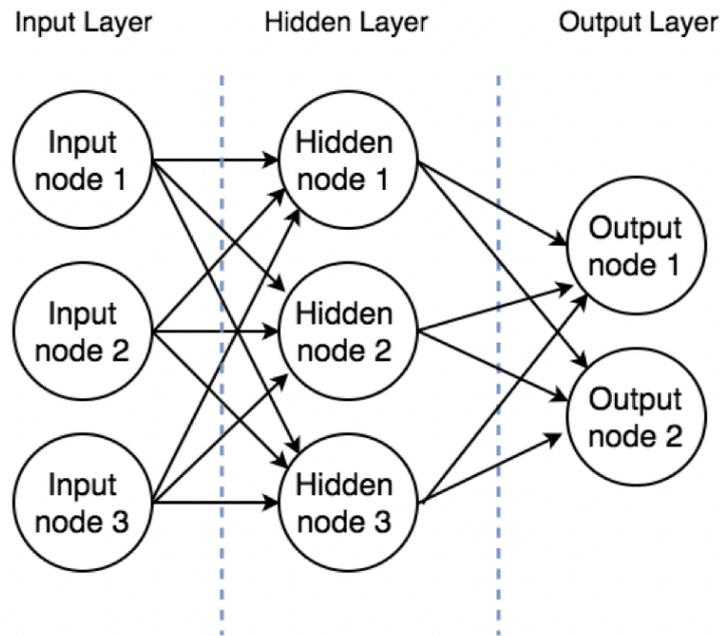


Figure 3. Neural networks for image compression

datasets. Its accuracy increases when more data is provided for the neural network. However, too much training data can lead to overfitting that jeopardizes the accuracy, since some imbalanced data has a side effect on the neural network. Back propagation is particularly suitable for training feed-forward networks. Even though it has considerable advantage, the back propagation can only be used when the activation function is differentiable. Researchers point out that the type of neurons or activation function a neural network utilizes characterize its performance and the complexity [10].

3.2 Other Neural Networks

In addition to ANNs, there are multiple other types of neural networks, such as convolutional networks, recurrent neural networks, which are based on the ANNs. These neural networks have some extraordinary features that enable them to perform better in different settings. They are also widely used in image and video compression.

4 Related work on Image Compression and Video Encoding

In [11], resampling block coding (RBC) compression method using the spatial sampling was introduced to fit in compact hardware architecture for onboard compression systems. It was applied for each unit block of the image with three sampling and quantization phases. It surpassed set partitioning in hierarchical trees (SPIHT) wavelet based image compression technique in terms of time complexity, maximum error and image quality within the reasonable compression ratio. It overcame the weaknesses of frequency transformation methods, including large processing units that is unacceptable for onboard compression systems and unnecessarily complex calculations that take more processing time. It was concluded that RBC compression method is applicable for aerial and remote sensing images.

In [10], two DNNs were designed and implemented for image compression. One is using logistic sigmoid unit, while the other is using hyperbolic tangent unit. Although these two methods have many identical properties, the tangent hyperbolic function outperforms its counterpart in many aspects because of the broader output space. This leads to a more efficient modeling process for a complex and abstract non-linear relation. The authors conducted the image compression with the two DNNs and compare the results measured by PDNR and the epoch vs. mean square error (MSE) curves. The simulation results concluded that the hyperbolic tangent neurons can not only gain a significant improvement on PSNR, but also converge several order of magnitude faster than sigmoid units.

A single recurrent neural networks proved to perform well on image compression, but was limited to images of a fixed size [12]. In order to provide a neural network which is capable of compressing images of arbitrary size, George Toderici et al. [7] introduced a series of full-resolution lossy image compression methods based on neural networks. Their compression networks included an encoding network, a binarizer and a decoding network. This architecture is distinct from that introduced in earlier work, in which the compression and decompression tasks were accomplished by only one network [10]. Furthermore, the authors applied the measurements of MS-SSIM and PSNR-HVS, which better represents human perception as metrics. The experimental networks were comprised of a combination of three types of recurrent units (LSTM, associative LSTM and gated recurrent units) and three reconstruction frameworks (one-shot

reconstruction, additive reconstruction and residual scaling). As the two metrics showed different training models with the best performance, it is not convincing to suggest the best models among them. However, many models perform better than JPEG performance on both MS-SSIM and PSNR-HVS on average, implying an improvement on image compression.

In [4], an adaptive neural network was designed to realize the desired compression of each picture block. The algorithm employed in the paper was a combination of motion detection, compression, and temporal subsampling of frames. It was able to reach the compression ratio of up to 500:1 for moving gray-scale images. The PSNR metrics was measured and a distinguished improvement was detected. The method was also computationally efficient, leading to potential implementation on real-time softwares.

In [13], a learning vector quantization (LVQ) has been implemented for image compression. This self-organizing network was trained very fast, reaching the maximum learning within dozens of thousand iterations. The PSNR was measured to illustrate the performance. Several other LVQs were also employed and the implementation of classified and multi-stage LVQs improved the performance.

In [14], a deep neural network architecture using a new training technique was designed to improve the performance of image compression. The new training method was referred to as Nested Training Algorithm (NTA), which has the advantages of enhanced computational efficiency. In order to evaluate the performance of the network, both synthetic and real-world data were used for simulation. The parameters of the network were controlled to obtain the optimal learning rates and the size of the training data for an image of a fixed size.

Fractal image compression is an important area in image compression. It is a lossy compression method for digital images that is based on fractals. In the fractal image compression, a recurrent neural network approach has already been proposed to improve the performance of image compression [15]. The partitioned iterated function system (PIFS) was implemented for image compression/decompression. The PSNR by applying the neural network approach was measured and desirable results were obtained compared to traditional PIFS method. Furthermore, due to the fact that neural network operations can execute in parallel, the compression/decompression can be processed faster than traditional methods.

In [16], a new single-structure image compression neural network based

on a new normalization scheme was described. According to the PSNR by several methods provided in this paper, the new neural network outperformed the mentioned multiple-structure neural network and a hierarchical neural network. It also improve the visual quality of the decompressed image.

5 Discussions and Conclusions

There is an increasing demand of utilizing digital videos and images because of an increasing demand of multimedia entertainments. It is very expensive to manipulate, store and transmit these resources in their raw forms. Therefore, image compression has become extremely crucial for modern digital communication and storage systems.

In this paper, we have discussed a great variety of state-of-the-art machine learning algorithms which can improve the performance of video and image compression/decompression process. It is clear that most algorithms show better performance than the traditional algorithms. These neural networks architectures also have less computational complexity, which reduce the processing time in both compression and decompression procedures. However, it is not evident to select the winning architecture for video encoding and image compression since there are many decisive metrics which denote different aspects of the designed model. In addition, as the experimental setups are not the same between the methods discussed in each paper, it is very difficult to compare the performance of the algorithms in different papers directly.

Machine learning algorithms have showed enormous potential for video and image processing tasks. Extensive evaluation for a broad range of different techniques and algorithms on multimedia computation is supposed to be carried out in the future.

References

- [1] "Cisco visual networking index: Forecast and trends, 2017–2022," Jan 2019.
- [2] M. Ghanbari, *Standard codecs: Image compression to advanced video coding*. No. 49, Iet, 2003.
- [3] J. Jiang, "Image compression with neural networks – a survey," *Signal Processing: Image Communication*, vol. 14, no. 9, pp. 737 – 760, 1999.
- [4] C. Cramer, E. Gelenbe, and H. Bakircioglu, "Low bit-rate video compression

with neural networks and temporal subsampling,” *Proceedings of the IEEE*, vol. 84, pp. 1529–1543, Oct 1996.

- [5] C. Cramer, “Neural networks for image and video compression: A review,” *European Journal of Operational Research*, vol. 108, no. 2, pp. 266 – 282, 1998.
- [6] A. Hussain, A. Al-Fayadh, and N. Radi, “Image compression techniques: A survey in lossless and lossy algorithms,” *Neurocomputing*, vol. 300, pp. 44 – 69, 2018.
- [7] G. Toderici, D. Vincent, N. Johnston, S. J. Hwang, D. Minnen, J. Shor, and M. Covell, “Full resolution image compression with recurrent neural networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5435–5443, July 2017.
- [8] P. Gupta, P. Srivastava, S. Bhardwaj, and V. Bhateja, “A modified psnr metric based on hvs for quality assessment of color images,” in *Communication and Industrial Application (ICCIA), 2011 International Conference on*, pp. 1–4, IEEE, 2011.
- [9] G. Toderici, S. M. O’Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, “Variable rate image compression with recurrent neural networks,” *arXiv preprint arXiv:1511.06085*, 2015.
- [10] F. Hussain and J. Jeong, “Exploiting deep neural networks for digital image compression,” in *2015 2nd World Symposium on Web Applications and Networking (WSWAN)*, pp. 1–6, March 2015.
- [11] A. Gohar, E. Osman, and H. Bayoumy, “A real time compression approach for aerial and remote sensing images,” in *2003 46th Midwest Symposium on Circuits and Systems*, vol. 2, pp. 548–551 Vol. 2, Dec 2003.
- [12] G. Toderici, S. M. O’Malley, S. J. Hwang, D. Vincent, D. Minnen, S. Baluja, M. Covell, and R. Sukthankar, “Variable rate image compression with recurrent neural networks,” 2015.
- [13] D. Erickson and K. Thyagarajan, “A neural network approach to image compression,” in *[Proceedings] 1992 IEEE International Symposium on Circuits and Systems*, vol. 6, pp. 2921–2924, IEEE, 1992.
- [14] A. Namphol, S. H. Chin, and M. Arozullah, “Image compression with a hierarchical neural network,” *IEEE transactions on Aerospace and Electronic Systems*, vol. 32, no. 1, pp. 326–338, 1996.
- [15] S. Lee, P. Wu, and K. Sun, “Fractal image compression using neural networks,” in *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227)*, vol. 1, pp. 613–618, IEEE, 1998.
- [16] Y. Benbenisti, D. Kornreich, H. Mitchell, and P. Schaefer, “A high performance single-structure image compression neural network,” *IEEE transactions on aerospace and electronic systems*, vol. 33, no. 3, pp. 1060–1063, 1997.

Android App Collusion

Ervin Oro

ervin.oro@aalto.fi

Tutor: Jordan Whitefield

Abstract

As people increasingly rely on their smartphones, both the number and complexity of malware attacks against Android are growing. While defending Android is an active area of research, a number of threats, e.g., app collusion, still cannot be reliably detected nor defended against. App collusion is a secret collaboration between apps with malicious intentions. This report surveys state-of-the-art literature discussing methods that can be used for collusion on Android, and known examples of such collusion. It also reviews existing approaches to app collusion detection, and their limitations. Existing literature has significant shortcomings, such as outdated or unbelievable claims and inconsistent definitions. Further research is required to develop feasible protections against Android app collusion, and the unified definition and criteria proposed in this report could be developed into a common framework for assisting the research community in achieving that goal.

KEYWORDS: *Collusion, Android security, Mobile security*

1 Introduction

Android is an operating system (OS) that is primarily designed for mobile devices, e.g., smartphones and tablets. With more than two billion active devices [1], it is estimated to be the most widely used OS, surpassing even Microsoft Windows [2, 3]. Android is designed to be an open platform: developed and maintained by Google LLC, but largely released as the Android Open Source Project for everyone to study and evaluate [4]. The Android OS includes support for apps, which are easily installable application packages that can extend the functionality of devices. Apps can be developed and distributed by anyone with a very low barrier of entry.

While this popularity of Android is not reflected by the proportion of malware attacks, most of which still target Windows, both the number and complexity of attacks against Android are increasing [5]. This is especially troublesome, as many people increasingly rely on their smartphones to store their personal data, online account credentials, financial information, and more. McAfee estimates that revenues for mobile malware authors could be in the billion-dollar range by 2020 [6].

Given the increasing potential damage from Android malware, defending against it is an active area of research. Android uses a multi-layer security approach, combining machine learning, platform security and secure hardware [1]. Machine learning methods are utilised by Google Play Store in an effort to prevent the uploading of potentially harmful applications, and by Google Play Protect [7] to scan apps locally on users' devices. The platform security of Android has been enhanced over the years with the addition of multiple security features, for example, SELinux protections [8, "Security-Enhanced Linux in Android"], exploit mitigations [9], privilege reductions [10], and encryption. Recent versions of Android leverage hardware security features, including keystore and remote key attestation [11], and receive regular software updates. These security mechanisms have been partially successful, as exploit pricing and difficulty are growing by some estimates [1].

However, malicious actors are continuously developing exploits to bypass existing protections, and a number of threats, e.g., app collusion, cannot yet be reliably detected nor defended against. App collusion is a secret collaboration between apps with malicious intentions (Section 2). This can be facilitated by any of the numerous ways for apps to communicate with each other that the Android system provides (Section 3). Methods for

apps to collude also exist on the iOS platform [12]. Given a malicious app that would be detected and blocked by state-of-the-art security systems, its functionality can be split into several apps, so that each of them would be categorised as benign when analysed separately [13].

Android app collusion is not a new concept [14], and multiple attempts have been made to develop suitable detection systems (Section 5). Computationally feasible but very coarse filters have been developed based on statically extracted features of apps [15, 13], while computationally expensive and more accurate filters have used formal modelling of Android apps [16] or modified versions of the Android OS [17] to track information flows. Heuristic approaches and manual analysis has also been proposed [18].

Despite this, there are currently no robust and usable ways to detect app collusions. Existing solutions have a large number of false negatives, false positives, or they are infeasibly difficult to implement. The number of possible combinations of N apps is N^N , and Google Play Store alone is reported to host more than 2.6 million apps [19]. Most proposed solutions therefore apply very aggressive filtering, causing only some malicious combinations to be included into analysis, and others to be reported as false negatives. Furthermore, most proposed solutions have a large number of false positives due to their inability to differentiate collusion from legitimate collaboration. Some research has experimented with checking apps manually to determine their intentions [18], but this demands even more aggressive filtering. Therefore, app collusion remains an open research challenge.

Furthermore, existing literature on the topic has significant limitations. Some influential work in the field is now outdated and has not been updated. Many authors have defined app collusion for themselves in ways that are simpler to detect but not applicable in the real world, often including large amounts of legitimate apps. Some published information also has credibility issues, either dismissing prior work by claiming that collusion is a new idea, or claiming that they have solved the problem without providing sufficient evidence.

This report provides an overview of the research of Android app collusion as follows: Section 2 discusses the nature and definitions of app collusion, Section 3 outlines the methods that can be used for colluding on Android, Section 4 describes known examples of colluding apps, and Section 5 reviews approaches that have been taken to collusion detection, and their limitations.

2 Description and definition of app collusion

The Oxford English Dictionary defines collusion as a “Secret agreement or understanding for purposes of trickery or fraud; underhand scheming or working with another; deceit, fraud, trickery” [20]. Asăvoae et al. [21] define collusion for the case of Android apps as the situation where several apps are working together in performing a threat. According to these definitions, app collusion must have the following three properties:

1. Colluding apps must be working together secretly. Conversely, apps working together in collaboration is a common and encouraged practice when such collaboration is well documented [22, “Interacting with Other Apps”].
2. All colluding apps must be in agreement. A distinctly different but related concept is the “confused deputy” attack [23], where one app mistakenly exposes itself to other installed apps.
3. Colluding apps must have malicious intentions. The intentions of Android app collusion would then be to violate one of the security goals of Android, which are defined in [8] as:
 - (a) protect app data, user data, and system resources (including the network).
 - (b) provide app isolation from the system, other apps, and the user.

It is important to note that the goal 3(b) is not to enforce isolation, but to provide isolation when required. As such, apps working together does not violate goal 3(b) in itself, but it would be a collusion if apps worked together to break isolation with some other app, the system, or the user.

This definition means that in addition to technical factors, non-technical factors, such as whether communication is secret or well documented, must also be taken into account. Additionally, this definition does not specify the channel or method used for colluding. However, many different definitions of app collusion have been used in the literature. One such example is by Asăvoae et al. [21], who further leave out from their definition all aspects relating to psychology, sociology, or documentation, such that, for example, a picture app which allows images to be sent through an email app, would be considered a collusion by them. Another alternative definition of app collusion is provided by Xu et al. [24], who define app collusion as user-unaware cross-app launch, therefore strictly limiting this definition to a single collusion method.

3 Methods for colluding

By default, all Android apps run in separate sandboxes [8, “Application security”], which are based on user separation by the Linux kernel, and enhanced by SELinux and `seccomp` [8, “Application Sandbox”]. By default, all communication between these sandboxes is blocked, but apps can open certain communication channels or prevent being separated into different sandboxes altogether. Some channels, so-called overt channels, are designed to be used by apps to communicate, while others, so-called covert channels, utilise functionalities which were originally intended for other purposes.

3.1 Overt channels

The Android OS has several channels designed for inter-app communication, which are described on pages “Application security” and “Application Sandbox” of [8].

Apps published by the same entity may share a sandbox. In this case, there are no restrictions for their communication, meaning that these apps can use any of the traditional UNIX-type mechanisms, including the filesystem, local sockets, or signals.

When apps are running in different sandboxes, the Linux kernel prevents these sandboxed apps from accessing any processes or files other than their own. In older Android versions, only Linux discretionary access control was used, allowing apps to make their files world-accessible, which was used internally by shared preferences – an inter-app communication method discussed by many works [25, 21]. Newer versions of Android forbid this by using SELinux mandatory access control rules, meaning that shared preferences are no longer relevant for this purpose. Apps can still use any file-based communication methods when they have permission to access the external storage, but this way users would be notified that such communication may take place when the Android system prompts them for permissions.

Android also provides a method for apps to communicate without any user-granted permissions or visibility. This is enabled by a remote procedure call mechanism called binder. Any app can send messages to binder arbitrarily, but other apps must explicitly start listening and accept incoming communications. The Android platform provides three methods for this:

- Services [22, “Services overview”]: Apps may start services, which can provide interfaces that are directly accessible using binder.
- Intent filters [22, “Intents and Intent Filters”]: Intents are message objects that represents an intention to do something. Apps may ask some part of them to be executed when an intent with certain properties matching their filter is initiated.
- ContentProviders [22, “Content providers”]: Apps can define ContentProviders to expose some of their data.

Binder provides a way for apps to communicate with each other, promoting openness and allowing a separation of concerns. Examples include apps using an intent to ask the camera app to take a photo instead of asking camera control permission, and communication apps allowing other apps to share data through itself. Since binder has a well-defined interface, information flow through it could be monitored.

3.2 Covert channels

In addition to overt channels, many covert channels have been discovered. Marforio et al. [26] propose a classification of communication channels based on whether application level APIs, OS native calls, or hardware functionalities are utilised. Al-Haiqi et al. [27] describe categorising covert channels as either timing or storage channels. However, neither of these categorising approaches provide clear boundaries in all cases, nor cover all possible covert channels. This section provides examples of covert channels in Android.

Schlegel et al. [14] show that any application can change the vibrate setting and use intent filters to be notified of changes to that setting without requiring specific permissions, therefore demonstrating that apps can create an information channel using the vibrate setting. Similarly, the volume setting can be used. While apps cannot subscribe to be notified when the volume is changed, and have to manually check this setting, it has the advantage of having 8 different states, as opposed to the vibrate setting, which is boolean. Both of these channels are invisible to users, as long as data transmission does not coincide with audio playback or receiving notifications.

Marforio et al. [26] describe how data could be exchanged between colluding apps by modifying and monitoring the number of threads, processor usage, and free space on the filesystem. However, the `/proc/stat`

APIs they used have since been deprecated [28]. Reviewing the current Android documentation with their ideas in mind suggests that alternative APIs might allow similar attacks to be mounted on recent versions of Android. For example, free disk space can be queried through the `StatFs#getAvailableBlocksLong()` API on latest Android versions [22]. A proof of concept could be created to verify the viability of this channel.

The system load can also be measured indirectly to transmit information, as described by Marforio et al. [26]. In this scenario, the transmitting app modulates the data payload by varying the load on the system. The receiving app then repeatedly runs a CPU-intensive computation and measures the time it takes to complete, in order to infer whether or not the transmitting app was loading the system. This approach was shown to work even when the receiver is JavaScript code running in a browser, and not an installed Android app.

Another approach is presented by Al-Haiqi et al. [27], where one app utilises the vibration motor to transmit data, and another app uses the accelerometer readings to receive that data. This is further developed by Qi et al. [29], who propose covert channels based on user behaviour. Instead of using the vibration motor, a transmitting app could prompt the user to move their phone in certain ways, for example, by posing as a rally game where the user needs to turn their phone at specific times based on a track generated by the malicious app.

These channels often rely on side-effects that are impossible or infeasible to eliminate, as they are caused by, for example, the limitations of real world hardware, or working principles of real world sensors. Research on detecting covert channels dates back to 1970's, and it has been shown to be a hard problem [26].

4 Examples of Android app collusion

Researchers have proposed several hypothetical and proof-of-concept collusion examples, and one example of app collusion has also been discovered in the real world. A common example of app collusion follows the pattern shown in Figure 1, and proceeds as follows:

- (1) APP_A obtains some private information.
- (2) APP_A transmits the information to APP_B using some overt or covert channel.
- (3) APP_B exfiltrates the information over the internet.

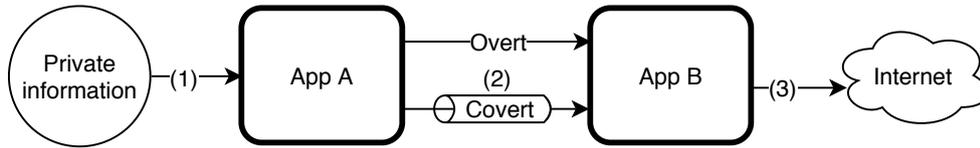


Figure 1. Structure of a basic app collusion example.

An early example of this kind of collusion was described by Schlegel et al. [14]. In their case, in place of the APP_A was an app called Soundcomber, which obtained private information using the microphone. To avoid detection, Soundcomber did not have permission to access the internet, instead they proposed to use a second app to exfiltrate the information. A similar hypothetical example is also described by Asăvoae et al. [21], where the APP_A would be a contacts app with `READ_CONTACTS` permission, and the APP_B would be a weather app with `INTERNET` permission. No examples following this pattern of collusion have been reported in the real world.

There is one known example of app collusion in the wild reported by Blasco et al. [30], and it has a different structure than described above. It is a set of apps from different vendors, all of which include a library known as the MoPlus SDK. The collusion was noticed after a manual review of apps that had already been previously categorised as potentially harmful, and were capable of accessing shared preferences files from different apps. The MoPlus SDK was already known to contain remote-control capability, and is now confirmed to also collude between different instances of itself.

The MoPlus SDK may be embedded into many different apps, with 20 such being identified by [30]. It can then open a local HTTP server allowing the attacker to abuse any permissions given to its host app. With potentially many instances of the MoPlus SDK running on the same device with different permissions, they would work together to guarantee that only the instance with highest level of permissions is the one that starts accepting remote commands (Figure 2):

- (1) Each instance stores a priority value based on the permissions it has
- (2) Each instance queries all other instances for their priorities
- (3) The instance with the highest priority is called

No sensitive information is exchanged between the apps, but still all three properties of app collusion as presented in Section 2 are present.

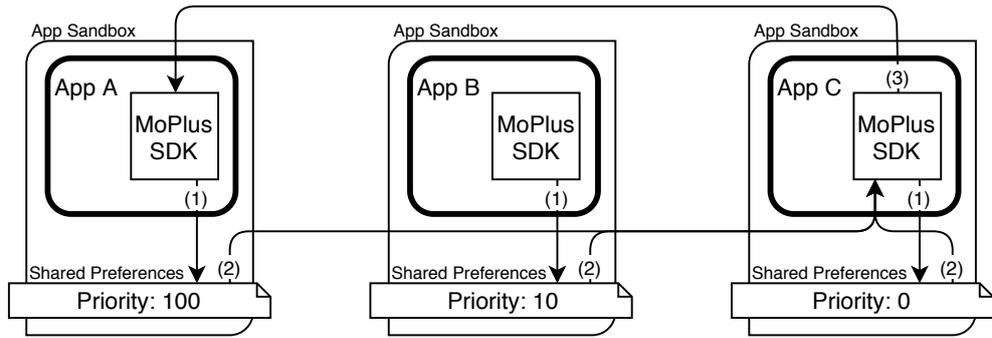


Figure 2. Collusion between instances of the MoPlus SDK [30]

5 Existing methods for detecting collusions

Attempts with various scopes and outcomes have been made to detect app collusion. Results range from computationally feasible but very coarse filters to more accurate but very expensive approaches.

5.1 Based on permissions/interfaces

The simplest approach is to base the analysis on statically extracted features of apps, such as the set of permissions declared or the set of Android APIs imported. Only considering such features makes it feasible to scan large amounts of apps. However, the obtained results only provide a coarse estimation of app collusion.

One example by Asavaoae et al. [15] is a filtering method based on the API calls and declared permissions of apps. They define collusion as a pair of apps that satisfy these three conditions:

- C1 First app declares a permission giving access to sensitive information.
- C2 Second app declares a permission giving the ability to send information to the outside world.
- C3 Finally, on some channel the first app must be able to send and the second app able to receive. As channels they consider Intents (each action separately), External storage, and shared preferences (each file separately; but now deprecated as discussed in Section 3.1).

It is not clear from their description, but can be assumed, that they further ignore apps that match both criteria C1 and C2, since otherwise their filter would detect any pair of apps that can communicate using one of these channels. They claim that while not all apps flagged by this approach are necessarily colluding, all colluding apps are flagged by this filter. However,

it must be noted that in reality they only consider a narrow subset of app collusions, and, for example, the MoPlus SDK would be labelled as benign by this filter.

Chen et al. [13] propose a similar approach. First, they observed that exists a machine learning algorithm that, given the set of all API calls in an app, can predict whether the app is malicious or not. Only a single overt channel is captured for communication between apps, as their work only considers intents. For each app they detect which intents it can send and receive, and then group together apps that could communicate with each other. To classify, whether a group of apps is benign or malicious, they pass the union of all API calls used by the apps in the group to the aforementioned machine learning algorithm.

Neither of these approaches can detect whether apps communicate with each other, nor what kind of information is exchanged between them, since they use no information about the order of operations within apps. Additionally, they both consider only a limited set of overt channels.

5.2 Based on control flow analysis

A more accurate detection of collusion can be achieved when the code of apps is analysed to detect whether or not they exchange data with other apps. Two approaches have been taken to that: model-based and runtime.

Asăvoae et al. [16] describe a model-based method for checking whether information theft through collusion exists within the set of possible control flows for an app. They propose disassembling apps and analysing Dalvik virtual machine (VM) instructions with their input and output objects. In all possible control flows, when one of predefined Android APIs returns an object, it is annotated with the initiating app ID and a boolean “sensitive”. Whenever an object is used as an argument to an instruction, its sensitivity and app ID values are propagated to all resulting objects. Input set of apps is considered to be colluding if there exists a control flow in which an object marked sensitive is passed to one of predefined exporting Android APIs, so that the current app ID is different to the initial app ID from the object. They explore modelling apps with different levels of abstraction, including more direct modelling in which case the analysis takes longer, or even forever when the app code contains any loops, and a higher level of abstraction, which is less accurate, but guaranteed to provide an answer. They only consider intent based inter-app communication channel, and do not analyse native code components of apps.

Another approach is to modify the Android OS to track information flow at runtime. Enck et al. provide an example of this called TaintDroid [17], which can track information flows from sensitive sources to sensitive sinks system wide. Within the Java code of apps it adds annotations for each variable separately (by augmenting the Java VM), but for communication through binder these are at message accuracy, in native system libraries with method call accuracy, and for disk access with file accuracy. TaintDroid cannot track information flow in native app components, which is solved by authors here by banning them entirely, breaking 5% of apps by some estimates. Use of annotations is similar to [16], but the amount of false positives is smaller, because the corresponding exit state must be reached for it to be reported. However, due to the limited granularity of annotations, their propagation can be overly conservative, which still results in false positives. False negatives are also possible, because annotations are not propagated through covert channels. Additionally, TaintDroid incurs a 14% CPU overhead.

5.3 Other

None of the aforementioned research has addressed detection of collusion that uses covert channels. Some work has looked into closing specific covert channels [28, 29], but it is not reasonable to assume that all covert channels can be found, especially since there exist known covert channels that have not been closed (Section 3.2).

Muttik [18] proposes augmenting approaches mentioned above with additional heuristic rules. He argues that colluding apps can also be detected using indirect signals, such as whether apps come from the same source, explicitly encourage co-installation, are frequently installed together, use similar libraries, etc. He also notes that signals like the publication date, app market and installation method can be used.

According to Muttik, McAfee has a product for defending against app collusion by combining the methods described in Sections 5.1 and 5.2, the abovementioned heuristics, and manual review and reverse engineering of apps. In 2016 McAfee claimed that their product is able to detect colluding mobile apps and stop any malicious combination of apps from running [31]. However, based on known information about the state of the art, including the limitations outlined above, this is unlikely to be entirely true.

6 Conclusion

This report presented an overview of Android app collusion. It surveyed state-of-the-art literature discussing methods that can be used for collusion on Android, and known examples of such collusion. This report also discussed existing approaches to app collusion detection, and their limitations.

The number and complexity of malware attacks against Android are increasing, which is especially concerning because mobile devices are rich sources of data. One of the threats that cannot yet be reliably detected nor defended against is app collusion, which can be defined as a secret collaboration between apps with malicious intentions. The Android OS has multiple overt and covert channels for apps to exploit for colluding. However, some communication channels that had been described, such as the ones based on `/proc/stat` or shared preferences, were determined to no longer be relevant due to changes in Android. An interesting future work would be to reproduce covert channels used for collusion, in order to confirm, which channels are still exploitable.

Existing literature on app collusion detection was also determined to have significant limitations. Multiple proposed solutions used some definition of app collusion that did not differentiate collusion from legitimate collaboration, or only included a subset of collusions, e.g., information exfiltration, making these definitions not applicable in the real world. One source has also made what seems to be an incredible claim of having solved the problem of app collusion.

From the literature it is clear that further research is required to develop feasible protections against Android app collusion. This report provided a comparison of the proposed collusion methods and protections using a common definition and criteria. These definitions and criteria could be further developed into a common framework, and could assist the research community in achieving results in the detection of app collusion.

Bibliography

- [1] Android Open Source Project *et al.*, “Android security 2017 year in review,” Mar. 2018.
- [2] Awio Web Services LLC. (2018, Dec.) Browser & platform market share. [Online]. Available: <https://www.w3counter.com/globalstats.php?year=2018&month=12>
- [3] StatCounter. (2018, Dec.) Operating system market share worldwide. [Online]. Available: <http://gs.statcounter.com/os-market-share>
- [4] Android Open Source Project. (2019) The Android source code. [Online]. Available: <https://source.android.com/setup>
- [5] AV-TEST GmbH, “Security report 2017/18,” Jul. 2018.
- [6] McAfee, “Mobile threat report,” Apr. 2018.
- [7] Android Open Source Project. (2019) Google Play Protect: securing 2 billion users daily. [Online]. Available: <https://www.android.com/play-protect/>
- [8] ——. (2019) Security. [Online]. Available: <https://source.android.com/security>
- [9] J. Edge. (2016, Aug.) Hardened usercopy. [Online]. Available: <https://lwn.net/Articles/695991/>
- [10] P. Lawrence. (2017, Jul.) Seccomp filter in Android O. [Online]. Available: <https://android-developers.googleblog.com/2017/07/seccomp-filter-in-android-o.html>
- [11] S. Willden. (2017, Sep.) Keystore key attestation. [Online]. Available: <https://android-developers.googleblog.com/2017/09/keystore-key-attestation.html>
- [12] L. Deshotels *et al.*, “SandScout,” in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*. ACM Press, 2016. doi:10.1145/2976749.2978336
- [13] H. Chen *et al.*, “Malware collusion attack against machine learning based methods: issues and countermeasures,” in *Cloud Computing and Security*, X. Sun, Z. Pan, and E. Bertino, Eds. Cham: Springer International Publishing, 2018, pp. 465–477. doi:10.1007/978-3-030-00018-9_41
- [14] R. Schlegel *et al.*, “Soundcomber: A stealthy and context-aware sound trojan for smartphones,” in *Proceedings of the Network and Distributed System Security Symposium, NDSS'2011*, Jan. 2011.
- [15] I. M. Asavoaie *et al.*, “Towards automated Android app collusion detection,” *arXiv preprint arXiv:1603.02308*, 2016.
- [16] I. M. Asăvoae, H. N. Nguyen, and M. Roggenbach, “Software model checking for mobile security – collusion detection in \mathbb{K} ,” in *Model Checking Software*, M. d. M. Gallardo and P. Merino, Eds. Cham: Springer International Publishing, 2018, pp. 3–25. doi:10.1007/978-3-319-94111-0_1
- [17] W. Enck *et al.*, “TaintDroid: an information-flow tracking system for realtime privacy monitoring on smartphones,” *ACM Transactions on Computer Systems (TOCS)*, vol. 32, no. 2, pp. 5:1–5:29, Jun. 2014. doi:10.1145/2619091

- [18] I. Muttik, "Partners in crime: investigating mobile app collusion," in *McAfee Labs threats report*. McAfee, Jun. 2016, pp. 8–15.
- [19] Statista. (2018, Dec.) Number of available applications in the Google Play Store from December 2009 to December 2018. [Online]. Available: <https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/>
- [20] "collusion, n." in *OED Online*. Oxford University Press, Dec. 2018.
- [21] I. M. Asăvoae *et al.*, "Detecting malicious collusion between mobile software applications: the Android™ case," in *Data Analytics and Decision Support for Cybersecurity*. Springer International Publishing, Aug. 2017, pp. 55–97. doi:10.1007/978-3-319-59439-2_3
- [22] Android Open Source Project. (2019) Android developers. [Online]. Available: <https://developer.android.com/>
- [23] N. Hardy, "The confused deputy: (or why capabilities might have been invented)," *SIGOPS Oper. Syst. Rev.*, vol. 22, no. 4, pp. 36–38, Oct. 1988. doi:10.1145/54289.871709
- [24] M. Xu *et al.*, "AppHolmes: detecting and characterizing app collusion among third-party Android markets," in *Proceedings of the 26th International Conference on World Wide Web - WWW '17*. ACM Press, 2017. doi:10.1145/3038912.3052645
- [25] S. Bhandari *et al.*, "Android inter-app communication threats and detection techniques," *Computers & Security*, vol. 70, pp. 392 – 421, 2017. doi:10.1016/j.cose.2017.07.002
- [26] C. Marforio *et al.*, "Analysis of the communication between colluding applications on modern smartphones," in *Proceedings of the 28th Annual Computer Security Applications Conference on - ACSAC '12*. ACM Press, 2012. doi:10.1145/2420950.2420958
- [27] A. Al-Haiqi, M. Ismail, and R. Nordin, "A new sensors-based covert channel on Android," *The Scientific World Journal*, vol. 2014, pp. 1–14, 2014. doi:10.1155/2014/969628
- [28] nn...@google.com. (2017, Mar.) Android O prevents access to /proc/stat. [Online]. Available: <https://issuetracker.google.com/issues/37140047>
- [29] W. Qi *et al.*, "Construction and mitigation of user-behavior-based covert channels on smartphones," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 44–57, jan 2018. doi:10.1109/tmc.2017.2696945
- [30] J. Blasco *et al.*, "Wild Android collusions," in *VB2016*, Oct. 2016.
- [31] McAfee, "Safeguarding against colluding mobile apps," May 2016.

State of the Art of IoT Data Model Translation

Kidus Mammo

`kidus.mammo@aalto.fi`

Tutor: Ari Keränen

Abstract

In order to reap the full benefit that IoT promises, it is necessary to have a flow of information between connected things. Currently, this is not the case as the IoT landscape is populated with heterogeneous platforms that lack interoperability. In order to tackle this problem, various alliances and organisations have been established with different standardization proposals for the future of IoT. However, there is still no consensus about the most appropriate approach. Therefore, it is necessary for the time being to develop methods of interworking between the existing standards. This paper focuses on current attempts to consolidate the issue of interoperability by reviewing methods of data model translation in the IoT space. It discusses current solutions for interworking (openHab, Vorto), those under development (Semantic Translation) and Direct translation specifications between standards.

KEYWORDS: *IoT, Interoperability, Data Model Translation, Interworking, Ontologies*

1 Introduction

The Internet of Things (IoT) is a rapidly growing field with tremendous economic, social and technical potential. It envisions to digitize the physical world, ushering in a future of real-time healthcare, smart-homes, smart-cities, fully automated industries, connected-vehicles, and many more potential benefits. A recent forecast made by Ericsson claims that there will be around 31.4 billion connected devices by 2023, out of which around 19.8 billion will be IoT-related [3].

A cornerstone in achieving this vision of IoT lies in the ability of a vast number of heterogeneous devices cooperating with each other to reach common goals. According to a study by McKinsey Global Institute, “of the total potential economic value that IoT enables, interoperability is required for 40 percent on average and nearly 60 percent in some settings” [10]. However, the current state of IoT development faces a challenge in terms of interoperability in different aspects [14]. This is due to multiple factors such as the existence of a wide range of devices, different equipment manufacturers, proprietary systems, a variety of standards, and diverse platforms (operating systems).

The requirement of achieving interoperability at large is not a new notion. The seamless internet we take for granted today was riddled with problems of interoperability in its infancy. This included varying communication protocols, different wireless standards, and the browser wars of the early 1980's. In order to achieve the inter-networking we have today, various standard bodies such as the World Wide Web Consortium (W3C), Internet Engineering Task Force (IETF) and the International Organization for Standardization (ISO) published a multitude of standards to guide the development of the internet. Currently, the same effort is being made in the world of IoT, with working groups, task forces, and the same standards organizations contributing to guaranteeing interoperability in IoT.

There have been many strides made towards achieving the interoperability of IoT devices over the past few years. A significant portion of this effort has been put to guaranteeing network interoperability with the development of communication standards such as IPV6 over Low-Power Wireless Personal Area Networks [9], IPv6 over BLUETOOTH(R) Low Energy [13], and ZigBee adoption of Internet Protocol Version 6. In addition, there has also been work done in the application layer with the

introduction of the constrained application protocol (CoAP) [25] and Constrained RESTful environments (CoRE) Link Format [24] for devices with a constraint in power, memory, and processing resources. The usage of Internet protocol in IoT devices has led to a majority of platforms supporting web services through REST APIs and similar data formats such as JSON, XML, and EXI [14]. However, this has not been enough to achieve the desired level of interoperability between IoT devices. There is still a challenge posed in exchanging data in a meaningful way so that IoT devices can understand and utilize the data. This discrepancy is caused by the different data models and schema used by IoT devices, and the challenge is referred to as semantic incompatibility [14].

Semantic interoperability aims to achieve data exchange across systems regardless of the format of the data while ensuring the exchanged data maintains the same meaning across the system. One approach to achieving this would be to seamlessly translate between different data models in IoT. This paper addresses this topic by conducting a survey of methods used for data model translation. Section II discusses two real world methods already under implementation to solve issues with interoperability, especially in regards to data model translation. Section III lists approaches towards data model translation that are still in development. Section IV evaluates all the discussed methods and identifies their advantages and disadvantages. Section VI concludes the paper by indicating possible research areas for future studies.

2 Current Solutions to Interoperability

The current state of IoT is often described as vertical silos of proprietary systems that provide little or no interoperability with similar systems. These silos are formed from the areas where IoT has made advancements such as smart homes, healthcare, retail, industry and connected vehicles. The current problem IoT faces is the lack of interoperability between these silos, and also within themselves due to various vendors utilizing different platforms, protocols and data representations in their devices.

In recent years, a cluster of organizations, alliances, working groups and task forces have proposed data models, vocabularies, information models and ontologies to tackle this problem. Some of the key industry suggestions in this venture include Internet Protocol for Smart Objects smart objects, Zigbee's Cluster Library, Open Mobile Alliance's LWM2M, Open

Connectivity Foundation's IoTivity, Allseen Alliance's AllJoyn, oneM2M's base ontology and W3C's Thing Description. In the current state of IoT, there exists no particularly dominant architecture to coerce an industry-wide standard to be adopted. For the foreseeable future, there will remain heterogeneity in protocols and schema resulting in a need for interoperability between different organizations. For this reason, there is a need to translate between endpoints of different protocols and schema.

While the standardization bodies race to provide the ideal IoT solution to provide interoperability for the future, IoT products have still been rolling out in the market. In 2018, it was reported there were around 8.6 billion IoT connected devices [3]. The full benefit of IoT lying in the ability of connected things to cooperatively automate the physical world, there is a demand for a timely solution to interconnect already existing devices. In the heart of this desire, several solutions have come to fruition that could offer interoperability frameworks for already existing devices. Here we discuss two of these solutions.

2.1 openHAB

openHAB is a vendor and technology agnostic open source automation software for the home [22]. It is an open source platform that provides users with tools to integrate their IoT devices across different data models, protocols and communication technologies.

openHAB achieves this integration by allowing IoT device manufactures or developers to define bindings to their devices. A binding is a software adapter that exposes the functionalities that are offered by a device and abstracts away the details of implementation. A binding in openHAB is written in Java by the manufacturer and it serves as a medium to translate services, data and protocols to and from the device. When developing a binding, the manufacturer defines the functionalities the device is capable of performing. For instance, this could be delivering some data such as sensor values or performing a certain action such as decreasing temperature in case of a smart thermostat. These functionalities that are defined in the bindings are referred to as Items in openHAB terminology.

Items are represented in openHAB as control mechanism such as a switch, or pre-defined data types such as strings or numbers. A user can install the bindings for the IoT devices they own. When a binding is installed, the items defined in the binding are available for interaction with the users. When a user performs an action with these items, the bind-

ing translates the command received to a request for the IoT device, and then returns the response to the user using the pre-defined data types of openHAB. The translation here is done by the manufacturer, who parses through the data that is returned from the device and provides only the relevant information in pre-defined data types.

Extending on this technique, openHAB allows for custom home automation by allowing users to write scripts that can be performed based on event triggers. These event triggers are referred to as Rules. Each rule can be triggered to invoke a script that performs a task. Triggers for rules could be interaction with an item, change in a device status or time based triggers. For instance, one can specify a rule where when a mobile device status becomes online, automatically turn on the lights and air conditioner. openHAB has a highly integrated, powerful rules engine that can be leveraged to do real home automation.

2.2 Vorto

Along the lines of IoT device interoperation, another project aiming to provide interoperability is Eclipse Vorto. The goal of the Vorto project is to push for standardization of IoT information models [4]. Vorto allows device manufacturers to describe device functionality and characteristics as device information models based on Vorto DSL (Domain Specific Language). The Vorto DSL gives the manufacturer the capability to model all relevant aspects of device interfaces in an easy to understand language, while at the same time ensuring the information remains machine parsable. The information models is then shared and managed in a central Vorto repository.

To create an information model of a device, Vorto requires device manufacturers to define the functionality of the device using Vorto DSL function blocks. Each function block allows definition of configuration, status, events and operations allowed by the device. This is then saved as an information model on the Vorto repository. The exact method the values sent by a device are converted to fit into this format is defined by mapping that is also specified by the manufacturer.

A mapping adds platform specific information to an information model. For each function block in the information model, the manufacturer specifies how to obtain values from the data received from the device using XPath expressions. Behind the scenes, the engine uses JXPath to apply XPath expressions on a java object graph to obtain a value. To add func-

tionality that may not be possible using XPath, it is also possible to add custom JavaScript or Java functions.

Once an information model is specified, Vorto offers a code generation tool that translates the information model into another format. The translation is based on Eclipse Xtend templates. This allows Vorto to generate output in different formats, be it in the form of code such as Java, or a documentation format such as Markdown.

The method through which Vorto maintains standardized information models is by using the Vorto meta model. Vorto meta model defines the relationship between data types, function blocks and information models, as well as their structure. This is based on the Eclipse Modeling Framework, and it makes sure that all models are machine-parsable which guarantees that they are validatable as well as translatable.

3 Approaches to Data Model Translation

This section discusses two approaches to data model translation. The methods under study here are related to translating ontologies, which is a concept that is omnipresent in IoT standardization efforts. An ontology can be defined as “a formal, explicit specification of a shared conceptualization” [8] and it can be used to represent knowledge as a set of concepts related to one another. Ontologies provide a machine-understandable description of entities, relationships and individuals. It is fundamental data structure for conceptualizing knowledge. Ontologies in IoT can represent shared domain knowledge and enable semantic interoperability [11].

Utilizing ontologies to solve the problem of IoT interoperability is a notion that is being pursued by multiple organizations and projects [2]. The advantages of ontologies is that they can be expanded to include new devices and made backwards compatible. Ontologies would allow devices from different vendors to communicate and also make it easier for third party developers to develop applications for IoT devices.

However, In the current attempt to use ontologies to solve the issue of interoperability, there has also appeared diversification with stakeholders in IoT having developed many different ontologies. As indicated in [7], there are almost 500 ontology based project for IoT spanning over 20 domains relevant to IoT (building, healthcare, robotics etc). As such, there is motive in allowing for interworking between ontologies. Some methods for achieving this goal are discussed hereafter.

3.1 Semantic Translation

Semantic translation is the process of translating one data model to another using semantic information to aid in the translation. Given some information described semantically in terms of a source ontology, it is transformed into information described in terms of a target ontology. Semantic translation attempts to translate from one IoT ontology to another by performing ontology alignment. Developing such a method of translating IoT data models is among the undertakings of the INTER-IoT project, one of the European Union's Horizon 2020 projects.

INTER-IoT project involves the design, implementation and experimentation of an open cross-layer framework and associated methodology and tools to enable voluntary interoperability among heterogeneous IoT platforms. The tasks completed in achieving semantic interoperability, among others, is to implement a semantic translator which can translate between the various ontologies that exist in the IoT landscape to a central ontology (Generic Ontology of IoT Platform). By defining bidirectional alignment with the central ontology, it is then possible to translate between the various ontologies that are in existence. For instance, IoT ontologies such as SAREF, OneM2M and OpenIoT have similar scope thus making it possible to translate between them.

INTER-IoT achieves this goal by designing and implementing an Inter Platform Semantic Mediator (IPSM) component that performs semantic translation based on ontology alignment [26]. Ontology alignment refers to the process of finding correspondence between ontologies which can either be a predicate about similarity or a logical axiom. These processes are called matching and mapping respectively. It then proceeds to output a file which specifies the alignment between the ontologies. IPSM was created based on Alignment API and EDOAL [6], and outputs an alignment format based on RDF/XML.

IPSM utilizes these alignment files from the artifact ontology to the central ontology to translate messages to and from the central ontology. In the case of an addition of a new artifact, all that is required is a pair of alignments between the new artifact and the central ontology which would guarantee interoperability without jeopardizing the existing channels.

3.2 Direct Translation

In contrast to generic data model translation methods under development, there are also attempts made to allow interoperability of two models by directly translating from one to the other. Here, we will see discuss some of those methods.

OCF to Alljoyn

Open connectivity foundation (OCF) is an industry group formed by 300 member companies with the mission of developing specification standards, interoperability guidelines and certification for devices involved in IoT. OCF utilizes RESTful API modeling language (RAML) and JSON for data modeling and serialization, and exposes its resources using the REST architecture. OCF has defined generic data models for a wide range of IoT sensors, actuators and composite device [21]. In addition, it has an online tool called oneIoTa that allows developers to create their own models from a set of basic models, and guarantees that each model is interoperable with others.

OCF has made several strides in providing a draft for data translation between OCF clients and non-OCF devices. It achieves this translation by using an OCF bridging device to translate data models and protocols from OCF clients to and from another device called a bridged client. With this bridge, the foundation has already mapped its own resources with that of AllSeen Alliance's AllJoy framework. An OCF bridged device structure is provided in (Fig. 1). An OCF bridging device has virtual clients and servers to interact with devices that are OCF compliant and has a virtual bridge server and client to interact with devices that don't conform to the standard. In the heart of the bridge device is the translator that transforms data and protocols from a specified bridge protocol to OCF and vice versa. There could be more than one translator in a bridged device.

The translator in a bridged device offers two possible types of translation: deep translation and on-the-fly translation. Deep translation requires an intricate understanding of the bridged device's data model and representation to provide mapping. An OCF bridge has implemented deep translation to the AllJoyn framework after the two organizations merged in 2016[20]. On-the-fly mapping on the other hand requires no prior knowledge on device specific schema. In the case of translation to AllJoyn, it can either utilize introspection that is provided by extra metadata provided or do a translation without introspection. If metadata is

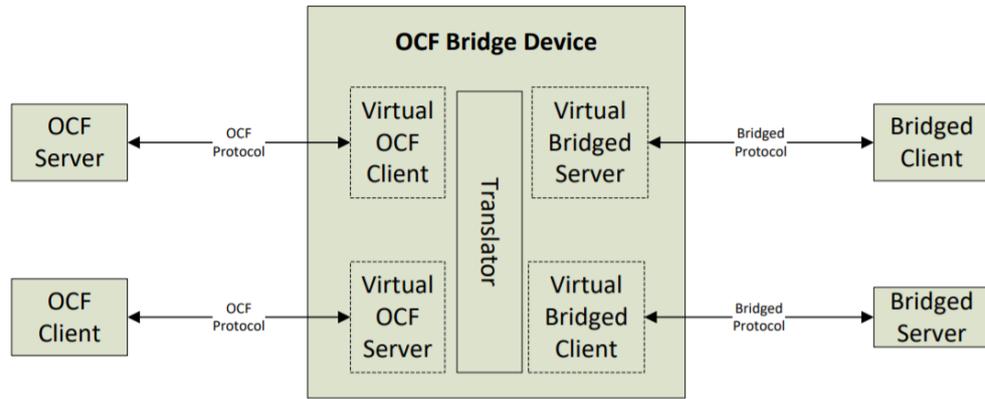


Figure 1. OCF Bridge Device Components [19]

available, the translator can use the extra metadata provided by the service to expose a higher-quality reply to the other side.

oneM2M Interworking Proxy Entities (IPE)

oneM2m is a global initiative to standardise horizontal machine to machine (M2M) and the IoT service layer specifications for globally interoperable M2M/IoT solutions. It is one of the key players in the attempts to provide a global IoT platform which allows for interoperability. However, oneM2M identifies there are several standards in existence that don't comply with it, therefore it provides an Interworking Proxy Entity (IPE) that allows the system to interact with non-oneM2M systems.

The aim of an IPE is to create an interface from a device technology/specific network to the oneM2M standard. It is a software entity characterized by the capability to remap non-oneM2M data models to oneM2M resources and maintain bidirectional communication with [18]. Depending on the complexity of the non-oneM2M data model, the IPE can be used to define a new complex set of resources based on the the basic oneM2M ones or simply directly map the communication. This approach enables a unique solution for semantic information exchange and data sharing among the different solution and deployments. Using IPE's, oneM2M has defined translation specifications with Alljoyn, OIC, and LWM2M. A recent study [1] goes to suggests using the same technique to support translation between oneM2m and FIWARE.

In addition, oneM2M defines an approach to translating data between Smart Device Template (SDT) conforming non-oneM2m devices by using the IPE. SDT is a reference template to model most home appliance functions in a unified way, which is a result of consensus among various stan-

standard developing organizations and alliances. In this specification [16], specific rules for mapping between SDT and oneM2M resources are defined. Procedures to using an IPE to perform the mapping are defined in [17].

4 Evaluation

In the above sections, we have seen several methods for data model translation. Here, we will discuss about their benefits and drawbacks.

openHAB provides a much needed timely solution to the lack of interoperability with IoT devices. It offers flexibility and a practical solution to automation. However, it requires a dedicated computer or a device such as a raspberry pi to serve as a hub. In addition, it is not a plug and play system and requires a certain degree of technical knowledge to get the full benefits. Though it may serve as a solution for home owners, the translation offered by openHAB only plays a small role in the grand vision of machine to machine communication.

Eclipse Vorto utilizes the notion of using meta-models stored in public repositories for data model translation. In order for the concept of meta-modeling to work, there needs to be someone or some organization that creates the models and decides what is important to include. Vorto gives this responsibility to individual device manufacturers, leading to the existence of different models for the same device. Meta-models can help solve the interoperability issue, but similar to other efforts, there needs to be a body that organizes it [15].

Semantic translation offers a generic approach to data model translation. It recognizes the significance that semantic knowledge and ontologies have in the future of IoT devices and bases its solution along those lines. However, the semantic translator is still at an early age of development and has only scratched the surface of its full potential. Studies done in using a semantic translator to translate between multiple geospatial data models reports that the the current existing automated methods to find alignments are not reliable for practice in the real world [5]. Another initial attempt to use semantic translation to translate from Semantic Sensor Networks ontology to Smart Appliances REference (SAREF) ontology reported a 60 percent of semantics maintained between the original and the generated ontology [12].

Direct translation methods discussed are a good step towards offering

interworking between several standards. These solutions might work in a scenario where there are only a few standards to translate to and from. However, from the current fragmented IoT perspective, this methods suffer from the so-called N-squared connectivity challenge [23]. This states that in order to facilitate information exchange between N connected devices with different data models, it requires building $N^2*(N-1)/2$ bridges. This approach is infeasible with the numerous models that are already in existence and its poor scalability with the addition of new models.

5 Conclusion

The current effort in achieving interoperability is a good step towards enabling the full potential of IoT. Even though a unified platform for all IoT devices would be the ideal solution, it is much more plausible there will exist multiple standards which are domain specific. In this regard, interworking between platforms is critical and data model translation plays a key role in that. In this paper, we have reviewed some of the approaches in data model translation in the IoT landscape. We have identified their benefits and potential drawbacks. To the awareness of the author, there are no pre-existing surveys in this specified topic therefore further research is recommended. Possible future research in this area include creating a taxonomy for the different approaches to data model translation and identifying common patterns in translation for the creation/recommendation of a generic data model translator.

References

- [1] J. G. An, F. Le Gall, J. Kim, J. Yun, J. Hwang, M. Bauer, M. Zhao, and J. S. Song. Towards Global IoT-enabled Smart Cities Interworking using Adaptive Semantic Adapter. *IEEE Internet of Things Journal*, March 2019.
- [2] Garvita Bajaj, Rachit Agarwal, Pushpendra Singh, Nikolaos Georgantas, and Valérie Issarny. A study of existing Ontologies in the IoT-domain. July 2017.
- [3] Ericsson Mobility Report, June 2018.
<https://www.ericsson.com/assets/local/mobility-report/documents/2018/ericsson-mobility-report-june-2018.pdf>.
- [4] Eclipse Foundation. Vorto Project Proposal. <https://projects.eclipse.org/proposals/vorto>. Accessed: March 29, 2019.
- [5] M. Ganzha, M. Paprzycki, W. Pawłowski, P. Szymeja, and K. Wasielewska. Alignment-based semantic translation of geospatial data. In *2017 3rd International Conference on Advances in Computing, Communication Automation (ICACCA) (Fall)*, pages 1–8, September 2017.
- [6] Maria Ganzha, Marcin Paprzycki, Wieslaw Pawlowski, Paweł Szymeja, Katarzyna Wasielewska, and Giancarlo Fortino. Tools for Ontology Matching—Practical Considerations from INTER-IoT Perspective. In *Internet and Distributed Computing Systems: 9th International Conference, IDCSS 2016, Wuhan, China, September 28-30, 2016, Proceedings*, pages 296–307, September 2016.
- [7] A. Gyrard, C. Bonnet, K. Boudaoud, and M. Serrano. LOV4IoT: A Second Life for Ontology-Based Domain Knowledge to Build Semantic Web of Things Applications. In *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, pages 254–261, August 2016.
- [8] Sara Hachem, Thiago Teixeira, and Valérie Issarny. Ontologies for the Internet of Things. December 2011.
- [9] N. Kushalnagar, G. Montenegro, and C. Schumacher. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. RFC 4919, The Internet Engineering Task Force, August 2007. <http://ietf.org/rfc/rfc4919.txt>.
- [10] Michael Chui Peter Bisson Jonathan Woetzel Richard Dobbs Jacques Bughin Dan Aharon Manyika, James. The Internet of Things: Mapping the Value Beyond the Hype . Technical report, McKinsey Global Institute, June 2015.
- [11] Sanju Mishra Tiwari and Sarika Jain. Ontologies as a semantic model in IoT. *International Journal of Computers and Applications*, pages 1–11, August 2018.
- [12] João Luiz Moreira, L.M. Daniele, Luis Ferreira Pires, Marten J. van Sinderen, Katarzyna Wasielewska, Paweł Szymeja, Wieslaw Pawlowski, Maria Ganzha, and Marcin Paprzycki. Towards IoT platforms’ integration: Semantic Translations between W3C SSN and ETSI SAREF. In *SIS-IoT: Semantic Interoperability and Standardization in the IoT*, November 2017.

- [13] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez. IPv6 over BLUETOOTH(R) Low Energy. RFC 7668, The Internet Engineering Task Force, October 2015. <http://ietf.org/rfc/rfc7668.txt>.
- [14] Mahda Noura, Mohammed Atiquzzaman, and Martin Gaedke. Interoperability in internet of things: Taxonomies and open challenges. *Mobile Networks and Applications*, July 2018.
- [15] Hilde Marita Oen. Interoperability at the Application Layer in the Internet of Thing. Master's thesis, Norwegian University of Science and Technology, 2015.
- [16] oneM2M. Home Appliances Information Model and Mapping Technical Specification. http://www.onem2m.org/images/files/deliverables/Release2/TS-0023-Home_Appliances_Information_Model_and_Mapping-V2_0_0.zip , August 2016.
- [17] oneM2M. Developer guide: Interworking Proxy using SDT. http://www.onem2m.org/images/files/deliverables/Release2A/TR-0039-Developer_guide-SDT-based_implementation-v_2_0_0.pdf , March 2018.
- [18] oneM2M. Functional Architecture Technical Specification. http://www.onem2m.org/images/files/deliverables/Release2A/TS-0001-Functional_Architecture-v_2_18_1.pdf , March 2018.
- [19] Open Connectivity Foundation. *OCF bridging specification*, 2017. Available at https://openconnectivity.org/specs/OCF_Bridging_Specification_v1.3.0.pdf.
- [20] Open Connectivity Foundation. *OCF resource to AllJoyn interface mapping*, 2017. Available at https://openconnectivity.org/draftspecs/OCF_Resource_to_ASA_Interface_Mapping_v1.0.0.pdf .
- [21] Open Connectivity Foundation. *OCF Resource Type Specification*, June 2018. Available at https://openconnectivity.org/specs/OCF_Resource_Type_Specification_v2.0.0.pdf.
- [22] openHAB. Documentation. <https://www.openhab.org/docs/>. Accessed: March 29, 2019.
- [23] Rajive Joshi, Paul Didier, Jaime Jimenez, Timothy Carey. The Industrial Internet of Things Volume G5: Connectivity Framework. https://www.iiconsortium.org/pdf/IIC_PUB_G5_V1.01_PB_20180228.pdf , February 2018.
- [24] Z. Shelby. Constrained RESTful Environments (CoRE) Link Format. RFC 6690, The Internet Engineering Task Force, August 2012. <http://ietf.org/rfc/rfc6690.txt>.
- [25] Z. Shelby, K. Hartke, and C. Bormann. The Constrained Application Protocol (CoAP). RFC 7252, The Internet Engineering Task Force, June 2014. <http://ietf.org/rfc/rfc7252.txt>.
- [26] Paweł Szmeja, Maria Ganzha, Marcin Paprzycki, Wiesław Pawłowski, and Katarzyna Wasielewska. Declarative Ontology Alignment Format for Semantic Translation. In *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*, pages 1–6, February 2018.

Electronic identification for citizens

Timur Kartaev

timur.kartaev@aalto.fi

Tutor: Sanna Suoranta

Abstract

Commonly, for identification of citizen paper based documents, like passports and identity cards are used. At present, with the rapid growth of the Internet and increasing number of services provided by the authorities, resulted in development of electronic identification systems. There are different ways of authentication used by the governments to ensure the person's identity. This paper attempts to provide information about current state of the electronic identity in the world, to describe different ways of identity check and to examine certain countries on how efficient those systems are.

KEYWORDS: electronic, identity, authorization, citizens, eID, Smart cards

1 Introduction

Since establishment World Wide Web(Internet) on 20 of December in 1990 the number of Internet users has drastically increased. In 2000, 413 millions of users consumed services on the Internet and from that time in 2016 the number of Internet users reached 3.4 billions [8]. According to rapid increase of users, most offline services shifted to online now. Electronic services are implemented in servers on Internet and provide access

for users. Nowadays, governments also restructured their departments to bring their services online. They are willing to supply increasing demand of online sector to provide citizens with access to public government services.

As a result, e-governments were established. E-government, also known as digital government, refers to "use of the internet technology as a platform for exchanging information, providing services and transacting with citizens, businesses and other arms of government" [15]. One of the main concern that arises for governments is to truly verify person identity. In order to provide access for public services to residents government need to develop an identification system to verify the information about users of the platform. However, documents that could provide the identification of a person are still physical, like passports and identity cards. These physical documents are not suitable for online identity checking. Moreover, in order to reduce the paperwork and increase the amount of served users to government services, governments strive to create electronic systems to identify their citizens.

Electronic identity that represents a certain person should ensure high standards of information security and provide accurate authentication in e-services. The main goal of the electronic identity is to be able to prove electronically the identity of the person and give an access to authority services. Commonly, electronic identity implies a physical card containing special electronic chip. The electronic identity card, usually, is issued by the police, post offices and other governmental offices. As an example, in Finland electronic identity card is issued by the police and abroad by Finnish consulates. The card has the same size as a credit bank card but includes personal information of the citizen. Electronic cards store different information varying on countries. For instance, in Brazil, the electronic identity card includes following sections: information about gender, nationality, date of birth, and origin, photo, signature and other personal information. It could not only be used to identify the person, but also as a voter pass [13]. Also, electronic identity cards could be used to encrypt emails, documents and create an electronic signature.

This paper is organized in a following way. First we will observe different authentication methods that are used by the governments, such as passwords or smart cards with cryptographic keys. Also, we will cover other methods of citizen identification, for instance, hybrid methods, alternative approaches involving non-governmental agencies and countries

where citizens rely more on private sector and private sector of identification. Furthermore, we will concentrate on analysis of different methods that are used by selected countries.

2 Current state of citizen identification

This section describe current state of digital authentication in some European countries. Each country applies different approaches for citizen authentications. Common ways of authentication include password based systems, smart cards and combined methods. Figures 1 demonstrates different approaches used by governments to authenticate citizens.

Country	Service name	Auth method
Denmark	NemID	Password based
New Zealand	RealMe	Password based
Netherlands	DigiD	Password based
Estonia	e-Estonia	Smart card
Italy	CIE	Smart card
Germany	Personalausweis	Smart card
Chile	eID	Smart card

3 Username and Password based authentication systems

The most simple way to authenticate citizen in a system is to use username and password. Password based services are more common to people. Nevertheless, password based systems have both advantages and disadvantages. The main advantage is users got used to passwords, easier maintenance for services and low cost of implementation. Nonetheless, strong password are tough to memorize and having different password for each service is a real burden for users. In general, password based system requires users to updated password periodically, since they are prone to leaking [14]. It is considering the main reason why password based systems for electron authentication is not popular among countries. Next section will examine several examples of countries that user username and password based systems.

One of the country that uses password authentication for residents is Denmark. Denmark has a system called NemID, which came into use in July 1 2010. The system was developed and maintained by Danish Agency

for Digitisation [5]. Denmark provides several ways how to obtain NemID such as citizen service center, in bank or request online. Every citizen who is over 15 years old and has a CPR-Number(Central Person Register) is eligible to apply for a NemID. The system implemented two factor authentication. NemID could be considered as legally binding signature that provides access to bank accounts and protects personal information. After citizen has entered username and password, user needs to enter a special key that corresponds to a number of NemID. There limited number of private keys but user can request new ones by mail. The private keys that resident received are kept in a central server, which makes it main target of attacker and considered to be one of the vulnerabilities of the system. On 11 April of 2013, the system was shut down because of massive DDoS attack, causing chaos in Denmark which eventually made Internet banking not possible to use during the attack [3].

Another country that use password based authentication system is New Zealand. New Zealand started the development of the system in 2000. At present, the system is maintained by Department of International Affairs. As it stated on the official website, the main goal of the system is to provide a secure way to prove person identity online. The system is called RealMe and provides sign-on using username and passwords [9]. To make passwords more secure and protected from brute-force attack, a password must include letters, numbers and special characters. Identity system provides two type of accounts verified and unverified. Unverified account allows access to services that do not require strong authentication. On the other hand, verified allows to open bank account, to apply for your first password, enroll to vote and to login to any government or private companies that have RealMe integration. The verification process based on four factors about citizen: full name, date of birth, place of birth and gender [9]. Additionally, through the system user could register a verified address in order to prove to any organization his or her legal place of location. The electronic identity could be obtained by anyone living in New Zealand. In order to get digital identity resident needs to visit local post office where resident has to submit a photo which will be attached to his or her digital identity.

Netherlands proposed a system called DigiD which is acronym for Digitale Identiteit. DigiD is also based on username and password that are issued by Dutch government. DigiD is used to identify citizens to provide access for online municipal services, such as tax authority, child ben-

efit and other governmental departments [10]. Dutch government provides digital identity online and citizen needs to follow several steps that are required to get digital identity. The following steps are only available in dutch language. First step is to enter personal information like BSN(burgerservicenummer), postal code, date of birst and date of birth into specified form. Next step concentrate on security, DigiD offers to enable two factor authentication, which is optional, by receiving sms every time when system is used. Third step requires user to decide on username and password that will be used in future. Lastly, email message is send to specified electronic address which must be entered as last verification step. After following steps, citizen finally receives electronic identity and ability to use public service online [7]. Dutch government do not requires citizens to be presented in police offices or in any other municipal organizations to receive digital identity.

4 Smart-Card-Based Systems

Another common method to identify citizens online is to use smart-card-based systems. Usually, smart card includes special chip that stores sensitive information about person. Currently, that kind of chip can be embedded in a variety of different plastic cards such as bank cards, student cards, loyalty cards and other type of cards. In general, smart card might have one of the two communication interfaces, it could be either contact and contactless, but some of them may provide both. A contact card requires physical contact, in order to use this kind of cards, they need to be inserted into special card reader. Contrarily, contactless card can be used wirelessly within a relatively short distance using electromagnetic induction technologies such as RFID or NFC. Smart cards are equipped with storing capacity for user sensite information like credentials. Additionally, smart cards also able to store such information as name, identification number, date of birth and a photo.

One of the primal example of smart-card-based systems is Estonia. The system considered to be one the first in a world. The card is issued by the police. At present, the e-card is a mandatory document for all citizens in Estonia. It is used to establish user identity in digital environment and also provides digital signature. As it stated on official website, nowadays, 98 percent of the citizens received their ID-cards. Moreover, in 2011 24.3 percent of Estonians voted online using their digital identity [4]. The chip

stores embedded files and uses 2048-bit public key which could be considered as resident signature that can be used as proof of resident identity. Electronic cards are used in various services, such as digital signatures, proof of identification when logging into bank accounts, for i-Vorint and in other private or government resources [4]. Recently, Estonia introduces new type of cards which provide more capabilities to their citizens. New card contains two interfaces contact and contactless. Contactless interface allows to authenticate user by wave of the card. The capacity is also increased allowing to use cards for public transportation. The new card is also differs from the old one by appearance.

Next country adapting electronic identity is Italy. The first card was issued in 2001, it is available to every citizen of Italy from 15 years of age. The card is known as Carta d'Identità Elettronica (CIE) could be used to identify citizen both online and offline. The technical specification of system was developed in cooperation with university of Rome Tor Vergata [12]. Card contains electronic chip that is responsible for storing digital certificate for online authentication and optional certificate for digital signatures. It is issued by Italian municipalities. However, infrastructure and service availability is provided by Ministry of Internal Affairs. Also, Italy provides citizen second card which is Carta Nazionale dei Servizi (CNS). This card could be used to gain access to public services, but it only could be used for online services and not as an offline identification of the citizen.

Germany introduced eID card on 1st November, 2010. The card contains contactless chip that carries personal information about citizen. All the information which is printed on the card is also available on the chip, personal information, including height, signature, eye color and other attributes that uniquely identifies a person. Also, two fingerprints could be added in to the chip without any additional cost. Every action performed under card is secured by cryptographic protocols. Smart card uses two factors in order to authenticate the user. First factor called "possession" which refers to the card itself and a PIN number "knowledge". German identification system using a concept called mutual authentication. Mutual authentication could be described as authentication of both parties, which means not only the holder of card is authenticated but also the service authenticates to the card which does not involve third-party applications. It has benefits that no third-party service involved which makes system more secure and reliable. Only after successful authentication of

both parties access is granted. In order to sign documents electronically the bearer of the card needs to explicitly require the signature function which is not available by default [6].

Digital identification systems are also used in South American countries. One of the example is Chile. The system was introduced in September 2013. Chilean identity cards has an Automated Fingerprint Identification System and a Facial Recognition System. It is stated that these feature helps to avoid duplication of person's identity. Currently, over 10 millions card has been issued and the government of Chile plans to provide identity cards to whole population in 2022 [1]. Another example in South America is Argentina. Argentine citizens have identity system called SID. The validation could be done with a personal mobile device that is equipped with a camera. The process happens in real-time remotely. The identification of person goes through validation of identity by face caption [2].

As it could be observed, one of the most popular way for citizen identification is based on smart cards. Since smart cards provide convenient way of storing data this approach of authentication is more preferable and trend goes to switch from usual national identity cards to electronic cards with smart chips embedded.

5 Non Governmental identity verification providers

Usually, the role of validation person identity is delivered by government, but in some cases it could be other entities that has been validated and has been given rights to perform identification action. This section describes other approaches for citizen identification performed by non-governmental agencies.

5.1 Mobile phone based identification

At present usage of mobile devices has rapidly increased. More services are tend to link user identity with a mobile phone. In addition to post office authentication, some countries implemented authentication through mobile devices. Almost every person owns a mobile device, which in turn makes him or her have a SIM card. Mobile Certificate is consist of chain of cryptographic keys that are stored on SIM card. With SIM card user can be authenticated on different services, encrypt and verify data and

confirm transactions. Commonly, the process of authentication through mobile devices involves 5 steps. First step you choose mobile certificate as an authentication method. Second, during login stage a user enters his or her phone number. The third step, in order to use mobile authentication a bearer has to accept identification request on their phone. Fourth step, a pass code should be entered. Last, after verification user granted a secure access to the services. In case when mobile device get stolen the credentials could be revoke immediately.

Another way how mobile devices used is two factor authentication. It serves as trusted channel between service and user. As an example New Zealand system send a verification code for extra security. User needs to enter a verification code to the system to login. Two factor authentication is considering more secure, since just stealing user credentials is not enough, but mobile device needed. Nowadays, several vulnerabilities to were discovered against mobile phone verification. One of the example is malware that is also installed on mobile phone. The malware that is installed on mobile device could intercept a message, which makes system vulnerable. Another possible attack on mobile device is hijacking the message by exploiting the cell network. An attacker is able to connect to the network and receive the message before it will be delivered to a target device.

5.2 Post Offices based identification

In some cases citizen identification goes through delivering mails with secret credentials. Moreover postal services, most of the time, are owned by the governments. Due to these factors post offices could be considered as a trusted agents. Moreover, post offices are responsive of delivering important mails, registering identification letters and other actions toward official documents.

In Switzerland, identity verification could be passed in Swiss Post office. Every citizen has two options either get USB stick with identity or a smart card. The identification of person goes in-person directly in Post office. There the identity of the recipient is verified using national identity card or passport. In order to use smart card or USB stick resident has to decide on PIN code. There are different security risks and vulnerabilities to the system, as an example, USB stick could be used on compromised computer, which allows an attacker to sniff personal information of the resident and steal the identity [11]. The SwissID, electronic identification

system of Switzerland, was developed in May of 2010 as a economic stabilization measurement by Federal Council. The system also developed a mobile phone application that allows to use personal mobile device for public services. Nowadays system supports various devices, including Apple Ipad, iPhone and other android mobile devices.

6 Discussion

In order to reduce paper work and be able to serve more clients, governments establish more online services. Considering information above, a smart-card for citizen identification is common standard. It could server as two factor authentication system, since person need to have a card and know the PIN code. In most of the countries government considered to be a primary source of digital identification. Since government provides other official documents, like birth certificates and passports, it already has infrastructure for citizen identification, as a result to implement on-line validation of the person becomes trivial. However, electronic identification is not developed in every country. With establishment of identification by government could be seen as a loosing privacy and freedom for citizens. There are could be several reasons behind it. First, private information of citizens could be compromised. Since some countries implemented bio-metrical data, after loosing control over data it could not be reissued which results in fear of providing this information to any organization. Second reason, the government is able to track every movement of their citizens, both online and offline. Due to this, some citizens could consider this as loosing privacy of their actions. The attitude toward electronic identity is different from country to country. In countries where citizens don't trust their government, it preferable to have electronic identification system by private sectors. In other countries where trust between government and citizens strong the identification usually provided by the government offices.

7 Conclusion

To sum up, the digital authentication is still in evolving phase. As it was observed, there is no particular way of identification, it may varies from countries. Current approaches of authentication could be spitted

in two categories. First, countries that use password based authentication for citizens. Second, other countries that decided to use smart cards. The identification system could be maintained by the government or outsourced to a different companies that are accredited by the law of that country. There are various additional features implemented on top of these methods, but in general it is either smart card or username and passwords. Additionally, depending on the country different information is stored on the cards, only some countries decided to store bio-metric information on the card. Moreover, different method of data encryption is used. This article inspects different approaches used by countries and both pros and cons of selected countries. However, to certainly identify a person on the internet is still a challenge and it is not only about the technology.

References

- [1] Steve Atkins. Digital identity for all! – safran delivers 10 million electronic id cards in chile. 2016. <https://silicontrust.org/2016/10/14/digital-identity-for-all-safran-delivers-10-million-electronic-id-cards-in-chile/>.
- [2] bitfinance.news. Argentina innovates with digital identity system. <http://bitfinance.news/en/innovates-with-digital-identity-system/>.
- [3] Charlene Chin. Denmark vision for digital government. 2017. <https://govinsider.asia/digital-gov/exclusive-denmarks-vision-for-digital-government/>.
- [4] Enterprise Estonia. e-estonia. <https://e-estonia.com/solutions/e-identity/id-card/>.
- [5] The Danish Agency for Digitisation. Nemid. <https://www.nemid.nu/dk-en/>.
- [6] Bundesamt für Sicherheit in der Informationstechnik. German eid. https://www.bsi.bund.de/Content/DE/Anliegen/eID/german-eID_node.html.
- [7] IamExpat. Digid. <https://www.iamexpat.nl/expat-info/official-issues/digid-netherlands>.
- [8] Julia Murphy and Max Roser. Internet. *Our World in Data*, 2019. <https://ourworldindata.org/internet>.
- [9] Department of Internal Affairs. Realme. <https://www.realme.govt.nz>.
- [10] Ministry of the Interior and Kingdom Relations. Digid. 2019. <https://www.digid.nl/en/>.
- [11] Swiss Post. Efficient, broad-based solution for a swiss digital id. <https://www.post.ch/en/about-us/company/media/press-releases/2017/efficient-broad-based-solution-for-a-swiss-digital-id>.
- [12] Sicurezza Pubblica. Identificazione di persone. 2013. <http://sicurezzapubblica.wikidot.com/identificazione-di-persone7>.
- [13] Eduardo Soares. Brazil: New national id card launched. 2011. <http://www.loc.gov/law/foreign-news/article/brazil-new-national-id-card-launched/>.
- [14] Rob Thubron. Leaked database exposes 87gb of emails and passwords. <https://www.techspot.com/news/leaked-database-exposes-87gb-emails-passwords.html>.
- [15] Zhang Nanping ; Li Yuan. Study and application of the soa based e-government system. <https://ieeexplore.ieee.org/document/4737689>.

Lightweight modern convolutional object detectors

Sriram Varun Vobilisetty

sriram.vobilisetty@aalto.fi

Tutor: Xu Jiayue

Abstract

This paper explores the current state-of-the-art convolutional object detectors and compares them according to their speed and computational resources required, and reviews the key architectures used in the best lightweight models suitable for deployment on mobile devices. Based on the limitations of the current mobile devices, methods that are not based on Deep learning are also considered.

KEYWORDS: Convolutional Neural networks, Lightweight, Object detection

1 Introduction

The prospect of computers and mobile devices being able to automatically detect and label objects within images or videos is fascinating. This functionality is even more practical if available on a mobile device in real time. However, most of the object detection methods existing today are computationally demanding, which does not allow for widespread deployment on mobile devices, or any device with low battery or processing power. Is this limitation an inherent one or is it just that the present state-of-the

art is not advanced enough to be able to perform efficiently? How much can the choices of architecture, platform, feature extractor etc. in deep learning methods help increase efficiency if one is to exhaustively try all the combinations? Is there any promising new approach which hints at a solution for computationally-inexpensive object detection? We intend to seek answers to the above questions and a few more in this paper, by surveying a variety of literature on the current state of object detection.

2 Literature review

So, my literature review was mainly divided into two parts: one part where I went through papers describing the best Deep Learning methods for object detection that could be run feasibly on mobile devices, and another where I searched for other non Deep Learning based methods for the same.

Out of many survey papers on Deep Learning based object detection, I found the one by Huang. et. al (2017) [1] to be the most informative as it gives a good comparison of three different architectures implemented on the same platform and trained with a variety of feature extractors. In this way, it fulfills the purpose of doing a fair 'apples-to-apples' comparison among models and also experimenting with different critical aspects of each of the models. Namely, Faster R-CNN, R-FCN, and SSD style of architectures are implemented on a Tensorflow framework for uniformity in comparison. Also, various convoluted feature extractors are experimented with each of the above frameworks and assessed for speed vs accuracy trade-offs.

The paper by Liu. W. et al., [7] elaborates on the SSD method which used a single deep neural network to generate the proposals and also do the detection training. They claim that SSD is better than other methods because it eliminates the need for a separate proposal generation stage and also the subsequent pixel or feature resampling stages.

The YOLO model proposed by Redmon. et al.,(2016) [8] also involves a single network which can be trained end-to-end on images but is based on a regression model. It is claimed to outperform other methods like DPM and R-CNN when generalizing from natural images to other domains

like artwork. They also improve upon their work, proposing YOLO9000 (Redmon. et al., 2017) [9] which is state of the art on several standard data-sets like PASCAL VOC and COCO. Its novel method of multi-scale training allows it to be run at different image sizes and the joint training allows it to predict detections for object classes which dont have labelled detection data! It also predicts detections for a much larger set of classes(around 9000) than any of the other methods.

To summarize, our main contributions are:

2.1 Model Comparison

So, among the various deep learning based models and others, how do we decide which one would be most suitable for deployment in mobile devices? Utilizing standard accuracy metrics, such as 'mAp' (mean Average precision) is not sufficient, as there are also other concerns for deployment. The methods which win competitions have the best accuracy, but are too slow for practical usage [1]. How do the models compare in terms of speed(images/frames per second), accuracy(mAp), and computational requirement(number of parameters, computations per iteration)? Also, we would rather be concerned with the test time performance as this is a more relevant criterion for the users of this technology than the training time. Let us begin by trying to understand the high level picture of some of the best known models and then look at the performance data for various models.

2.2 SSD

SSD or Sigle Shot Detection refers to an architecture or rather a meta-architecture where there is a single deep neural network through which an image is passed and the output is labels and positions of the detected objects. The most successful model of this type, also named as SSD is the one presented by Lie. et al., whose schematic is shown in Figure 1 taken from [7].

SSD is simple relative to other models that require object proposals because it totally eliminates the need for a proposal generation and subsequent pixel or figure resampling stages, making it easy to train and incorporate into a bigger system which requires an object detection component.

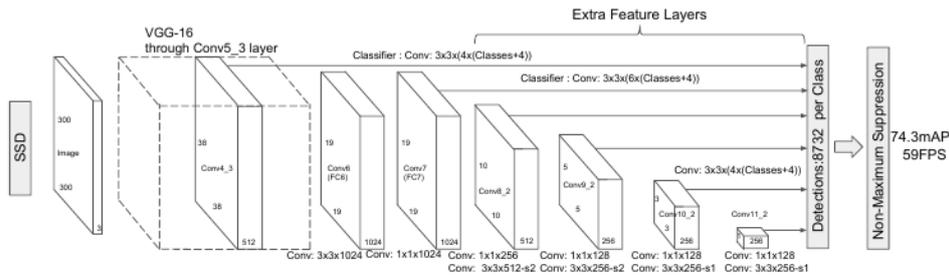


Figure 1. A schematic showing the SSD architecture

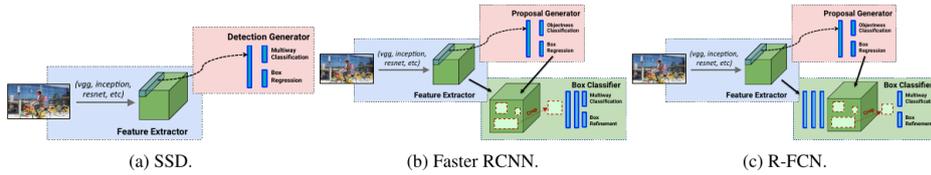


Figure 2. A schematic of the three main meta-architectures for object detection

2.3 Faster R-CNN

The faster R-CNN which is an improvement over the well known R-CNN model has two stages of detection. The first stage, called the RPN (Region Proposal Network) extracts features from raw images and also generates class-agnostic box proposals using a grid of default anchors tiled over the image. In the second stage, these (typically 300) box proposals are fed to the remaining of the feature extractor to predict a class and a class-specific box refinement for each proposal [1].

The notable improvement over R-CNN is that we do not crop proposals from the image, and instead crop them from a middle stage of feature extraction, which reduces some of the redundant computation. The running time was found to depend on the number of proposals generated in the first stage.

The schematic for a typical Faster R-CNN architecture can be found in 2 taken from [1].

2.4 R-FCN

Dai. et al. [11] took this idea of pushing cropping closer to the prediction layer a step further and proposed the R-FCN or Region based Fully Convolutional Networks in which crops are taken from the last layer of features prior to prediction. According to them, this approach minimizes the amount of per-region computation that must be done. They also proposed a position-sensitive cropping mechanism as an alternative to the

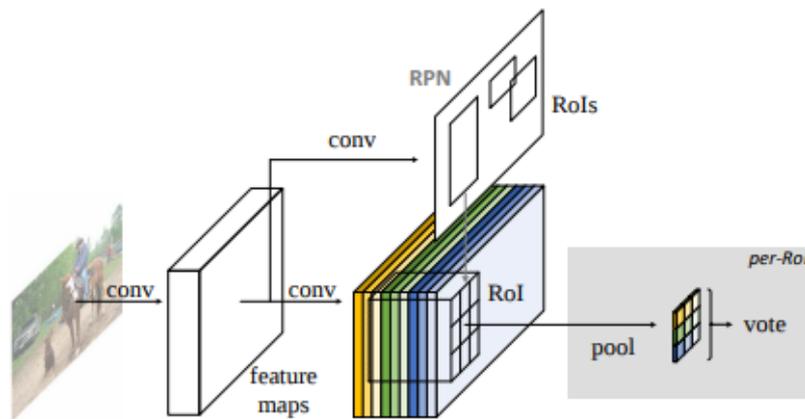


Figure 3. Schematic of R-FCN architecture

standard ROI pooling operations used in the other models. The schematic is shown in 3 taken from [11] and has been proved to achieve an accuracy comparable to that of Faster R-CNN but at faster running times [1].

3 Challenges in 'Mobile' Object Detection and Tracking

Although a lot of work has been going on in the Computer Vision field for improving the speed and efficiency of detection algorithms, the present state-of-the-art are still too computationally heavy and slow to run smoothly on the current mobile devices [13]. The common bottleneck of almost all of the methods seem to be the 'convolutional layers' [14].

There are cloud based server-client solutions where part or whole of the heavy-lifting is carried out by remote servers and then relayed to the mobile devices [13], but purely mobile based detection systems are yet to be developed. This may be a limitation posed by the current processing capabilities of mobile phones. For example, the latest mobile phones have a maximum of 8 GB RAM. The GPUs on most smartphones use shared memory and do not support general computation, which are the two main reasons why we can't viably do on-board object detection tasks at a meaningful speed.

3.1 Problems tackled by above models

Table 1 presents the salient features of the three architectures discussed above in a tabular form.

Architecture	Key Feature
SSD	A single network for both proposal generation and prediction.
Faster R-CNN	Two stage detection, cropping of proposals at an intermediate feature extraction layer
R-FCN	Minimizes amount of per-region computation to be done by pushing cropping to the last layer

Table 1. A table showing some key aspects of the discussed architectures

3.2 Unsolved challenges

III IOOAKFJ ;PSKJV SLKNVM LZSKDVM ;ZSLDKVMA S;LKDBM A;SLBVK'
ASLDVLKM 'SADLV KM'AS;LDVM ';SLDM V;ASLKMD V;LSAKDM VSALV

4 Experimental Findings / Performance Comparison on COCO dataset

Performance Comparison (COCO Dataset)			
Model and Feature Extractor	Inference time (ms/image)	Accuracy (mAp)	Computational requirement
SSD+Mobilenetv1	30	21	Nvidia GeForce GTX TITAN X
SSD+Mobilenetv2	31	22	Nvidia GeForce GTX TITAN X
SSD+Resnet 50	76	35	Nvidia GeForce GTX TITAN X
SSD+Inceptionv2	42	24	Nvidia GeForce GTX TITAN X
Faster R-CNN+Inceptionv2	58	28	Nvidia GeForce GTX TITAN X
Faster R-CNN+Resnet 50	89	30	Nvidia GeForce GTX TITAN X
Faster R-CNN+Resnet 101	106	30	Nvidia GeForce GTX TITAN X
R-FCN+Resnet 101	92	30	Nvidia GeForce GTX TITAN X

Table 2. A table showing the performance metrics of combinations of various Deep Learning models and feature extractors on the task of Object Detection. Data from [12]

5 Future prospects and further research

Till date, most of the work in object detection has been focused on detection in 2D images/videos. But with coming-of-age technologies such as self-driving cars and use of Augmented Reality in games, construction, health-care etc., the prospect of object detection in 3D is picking up interest among researchers.

Pioneering work has been done in this domain by Ali. et al., (2018) [10] using data from LiDAR point cloud. In their paper [10], they describe their 3-D object detector *YOLO3D* based on the one-shot regression meta

architecture and extended from their 2D version by including the 3D data attributes such as "yaw-angle", the "3D box center" etc in their loss function. Their model shows promising results on KITTI benchmark, with real time object detection achieved at 40 fps on Titan X GPU.

Still, there seems to be space for improvement in this domain as this is relatively low compared to the achieved speeds and accuracy in the 2D domain. Also, an interesting application would be smart eyewear or a visor equipped with this functionality, for which we would need to be able to do this task with relatively less computational requirements.

The latest mobile phones have a maximum of 8 GB RAM, and usually no Graphic card(GPU) which can do general computations. If somehow in the near future, GPU's become an essential feature of mobile phones and other devices, frameworks like the *DeepMon* proposed by Huynh. et al, [14] may be built upon and developed to full fledged mobile object detection software systems.

References

- [1] Jonathan Huang, Vivek Rathod, Chen Sun Menglong Zhu, Anoop Korattikara Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, Kevin Murphy. *Speed/accuracy trade-offs for modern convolutional object detectors*.
http://openaccess.thecvf.com/content_cvpr_2017/papers/Huang_SpeedAccuracy_Trade-Offs_for_CVPR_2017_paper.pdf
- [2] Anonymous [*Object Detection Using Convolutional Neural Networks*].
https://gluon.mxnet.io/chapter08_computer-vision/object-detection.html
- [3] Zhong-Qiu Zhao, Member, IEEE, Peng Zheng, Shou-tao Xu, and Xindong Wu *Object Detection with Deep Learning: A Review*
<https://arxiv.org/pdf/1807.05511.pdf>
- [4] Karanbir Singh Chahal, Kuntal Dey *A Survey of Modern Object Detection Literature using Deep Learning*.
<https://arxiv.org/abs/1808.07256>
- [5] Yuxi Li, Jiuwei Li, Weiyao Lin, Jianguo Li *Tiny-DSOD: Lightweight Object Detection for Resource-Restricted Usages*.
<https://arxiv.org/abs/1807.11013>
- [6] Sara Robinson, Aakanksha Chowdhery, and Jonathan Huang *Training and serving a realtime mobile object detector in 30 minutes with Cloud TPUs*.
<https://medium.com/tensorflow/training-and-serving-a-realtime-mobile-object-detector-in-30-minutes-with-cloud-tpus-b78971cf1193>

- [7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg "Ssd: Single shot multibox detector," in *European conference on computer vision, 2016*, pp. 21–37..
<https://arxiv.org/abs/1512.02325>
- [8] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi "You Only Look Once: Unified, Real-Time Object Detection; *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016*.
<https://arxiv.org/abs/1506.02640>
- [9] Joseph Redmon, Ali Farhadi "YOLO9000: Better, Faster, Stronger"; *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017; DOI:10.1109/CVPR.2017.690*.
<https://arxiv.org/abs/1612.08242>
- [10] Waleed Ali, Sherif Abdelkarim, Mohamed Zahran, Mahmoud Zidan, Ahmad El Sallab., 2018 "YOLO3D: End-to-end real-time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud"; *CoRR, abs/1808.02350*, [viewed March 2019]
Available from <https://arxiv.org/abs/1808.02350>
- [11] Jifeng Dai, Yi Li, Kaiming He, Jian Sun., 2018 "R-FCN: Object Detection via Region-based Fully Convolutional Networks"; *arXiv preprint: arXiv:1605.06409* [viewed March 2019]
Available from <https://arxiv.org/abs/1605.06409>
- [12] Github handle :pkulzc,tombstone,nealwu "Tensorflow detection model zoo"; *tensorflow/models* [viewed March 2019]
Available from https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md
- [13] Chen Tiffany, Balakrishnan Hari ,Ravindranath Lenin ,Bahl, Paramvir. (2016). "Glimpse: Continuous, Real-Time Object Recognition on Mobile Devices."; *GetMobile: Mobile Computing and Communications*. 20. 26-29. 10.1145/2972413.2972423. [viewed March 2019]
- [14] Nguyen Huynh, Loc Lee, Youngki Krishna Balan, Rajesh. (2017). "Deep-Mon: Mobile GPU-based Deep Learning Framework for Continuous Vision Applications". p 82-95., DOI:10.1145/3081333.3081360.
- [15] Zhang Wenxiao Lin, Sikun Hassani Bijarbooneh, Farshid Fei Cheng, Hao Hui, Pan. (2018). "CloudAR: A Cloud-based Framework for Mobile Augmented Reality".
Preprint Available from https://www.researchgate.net/publication/3250-34028_CloudAR_A_Cloud-based_Framework_for_Mobile_Augmented_Reality

Analysis of fallback authentication mechanisms

Satenik Hovsepyan

satenik.hovsepyan@aalto.fi

Tutor: Siddharth Rao

Abstract

Reclaiming access to an account after the password is lost plays a major role in the security of authentication systems. Different authentication systems implement a wide variety of account recovery mechanisms. Most prevalent ones include security questions, SMS and email-based recovery, social authentication and manually checking credentials.

In this paper, we analyze the usability and security of those mechanisms, in particular, different types of security questions, their comparison with each other as well with alternative techniques.

KEYWORDS: *security, fallback authentication, security questions*

1 Introduction

Mainstream authentication systems develop and apply numerous methods to ensure the security of the system. They commonly use passwords, biometrics, physical tokens or private keys as the main authenticator. However, users forget passwords, lose tokens and private keys, biometric features become temporarily or permanently unavailable. This raises the requirement of having a secondary authentication scheme (also referred

to as fallback authentication, backup authentication or account recovery).

Clearly, the overall security of the fallback authentication scheme should be at least as high as that of the main authentication scheme. Unfortunately, in practice this is not always viable. Secondary authentication schemes oftentimes have security weaknesses that substantially decrease the overall security of the system.

Since fallback authentication is not used every day, it must have better memorability. For this reason, various authentication systems take advantage of the knowledge users already have, in contrast with the information they purposefully memorize, through the use of security questions. Others use a predefined email address or phone number to deliver a one time recovery token. In some particular cases, manual check of credentials or social authentication is used.

The goal of this paper is to examine currently applied and proposed fallback authentication mechanisms, compare them based on their security, usability and deployability features.

The rest of the paper is structured as follows: section 2 describes general ideas behind authentication schemes and motivation for the study; section 3 is the main study, that presents several types of security questions (predefined, user-chosen, preference-based, activity-based and location-based) and their comparison, after which it presents several other fallback authentication schemes such as SMS and email-based authentication, social authentication and manually checking credentials as well as the use of machine learning in authentication; finally, section 4 summarizes the discussed topics and concludes the paper.

2 Background and Motivation

Authentication is the process of verifying that someone or something is, in fact, who or what it declares itself to be. In general, authentication can involve verifying user's identity based on identity documents, known secrets, biometrics and other factors [13]. Authentication is important as it allows organizations to keep their resources secure and share them only with authorized users. In contrast with authentication - the process of verifying the identity of the user, authorization is the process of verifying user's permissions. Users may be authenticated, but fail to perform specific actions if they are not authorized, i.e., given specific permissions. Although terms authentication and authorization are often used inter-

changeably, they have two distinct functions [13].

In order to authenticate to the system, users should present a piece of evidence that only they own and the server can verify. This piece of evidence is called an authentication factor. Most commonly used authentication factors include

- *Knowledge factor*: Something the user knows. This can be a password, pin or any other information that the user possesses.
- *Possession factor*: Something the user has. This can be an ID card, a physical token, one-time password token or any other item that only the user can own and can carry with him/her.
- *Inherence factor*: Something the user is. This can be any biometric data, such as fingerprint, facial recognition or any other biometric identification.

The above-listed factors are typically combined in order to improve the security of the system.

The problem, however, is that none of those factors are persistent. In case of knowledge-based authentication, secrets can be forgotten; in case of possession-based authentication, owned evidence can be lost or stolen; in case of inherence-based authentication, biometrics can be temporarily or permanently unavailable (e.g., fingerprint changed by a cut, voice changed because of illness) [9]. In order to address the issue, authentication systems develop **fallback authentication mechanisms**. Fallback authentication provides a means to regain access to the account after the loss of the primary authenticator.

Fallback authentication usually involves two stages. During the first stage, enrollment, users provide some information, such as answers to security questions, email address or phone number and biometrics. This information is used during the second stage, recovery, when a password reset is required. Oftentimes, the time range between the two stages is very long. Thus, memorability and durability of the provided information are major aspects to be considered in the usability of the mechanism. Another aspect affecting usability is the duration of enrollment and recovery stages.

3 Main Study

Numerous implementations of fallback authentication mechanisms are deployed in practice in different systems. However, there is no single established standard for it [5].

Below, we discuss some of the commonly used techniques.

3.1 Security questions

Perhaps one of the most widely known techniques for fallback authentication is **security questions**.

In this scheme, the user selects several questions that the website provides (e.g., "Mother's maiden name" and "Frequent flyer number") and gives correct answers to them. Later, during the recovery, the user should provide the same answers in order to regain access to the account. The advantage of this method over regular passwords is that the answers are supposed to be something the user already knows, in contrast with the specifically created and memorized passwords. In addition, questions give hints about the answers. However, it has been shown that these answers have poor memorability which decreases over time, and they are vulnerable to guessing attacks [14]. The reason for this vulnerability is that some questions have common answers shared by many users. For example, if the attacker guesses "Maria" for "Mother's maiden name" question, there is a high chance that it is the victim's mother's maiden name. Other questions, such as "Who is your favorite superhero?", have very few plausible answers. Another notable reason for security questions' low entropy is that users often try to make their answers harder to guess by providing untruthful responses. However, research shows that users "harden" their answers in a predictable way. A study was conducted at Google to analyze the motivation for untruthful responses [3]. Figure 1 illustrates the goals of users who admitted having provided fake answers to security questions.

The study shows that the majority of users claim to fake the answers to improve the security or to make the answers easier to remember, though the outcome is the exact opposite. This happens since when trying to give an incorrect answer to a questions people tend to choose answers that other users are also prone to choose. For example, when answering the question "What city were you born in?", a lot of people may choose Paris making it a popular choice between truthful and untruthful responses.

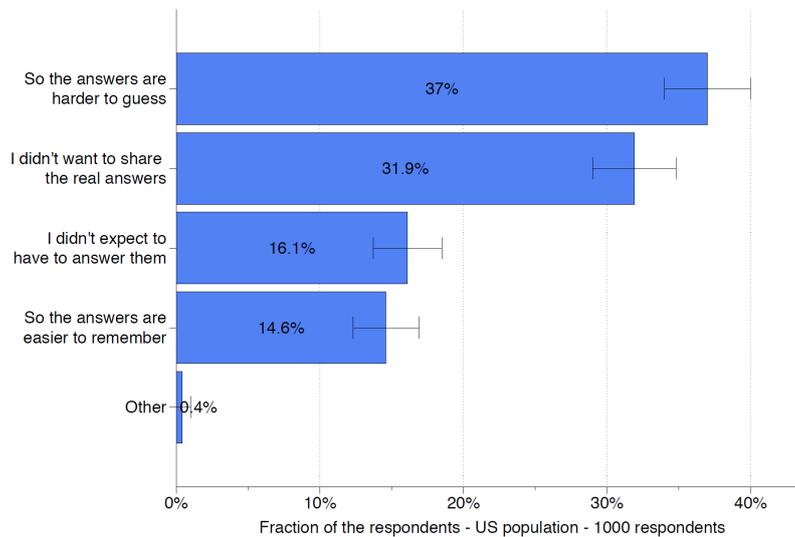


Figure 1. Survey answers for the question "Why did you provide fake answers to your password recovery question?" [3]

In other cases, people may decide to give a completely irrelevant answer such as "Megatron" or, even better, "NvX8z4yJzmu". While the second approach is secure in contrast with the first one, it still decreases the usability drastically, since the answer is no more a true memory the user holds.

Nonetheless, personal knowledge questions are acceptable in cases when the security risk is low or if they are combined with other methods [3].

In order to address memorability and security issues of personal knowledge questions, some authentication schemes support **user-chosen security questions**. In this design, users are supposed to choose questions themselves. It is assumed that this way questions will be more applicable to the user and thus be harder to guess. However, research shows that unguided user-chosen questions still have poor memorability and low entropy [10].

A potentially more secure alternative form of authentication questions is called **preference-based authentication** [8]. In this scheme, users answer a large set of personal preference questions (e.g., "Do you like country music?" and "Do you like game shows?"). Later, during authentication, users are presented with a preset of the original list and, if their answers are close enough to their initial preferences, they are authenticated. In particular, the approach distinguishes between small and big errors, i.e., if the answer initially was "Really like" in 3-point Likert Scale,

while later it is changed to "Really dislike", the error is considered big. A small error is accounted if the answer is changed by one point, i.e., from "Really like" to "Don't care". A few minor deviations like this are allowed.

This scheme prevents statistical guessing since an illegitimate user can only guess the answers to the questions for which the user has strong opinions. Thus, an illegitimate user will make a significant amount of big errors. Listed claims are backed by experiments [8].

The approach is motivated by the higher durability of people's likes/dislikes compared to their memory, and the absence of preferences in public records or online databases.

Nevertheless, what prevents preference-based questions from being widely deployed in practice is that they are more time-consuming at the enrollment stage than individual questions.

An nonintrusive approach with no user friction at the enrollment stage is based on **activity-based personal questions** [2]. It benefits from the massive amount of data that is already stored on servers or user's devices and dynamically generates activity-based ephemeral authentication questions. Correct answers to them are automatically extracted from user's web activities without user's participation (e.g., the answer to "Who was the last person you sent mail to today?" is retrieved from the mail server). Proposed activity-based security questions are divided into three categories:

- *Network Activity*: Questions that focus on user's online activity (browsing history, emails)
- *Physical Events*: Questions based on user's planned events from emails, calendars, social networks
- *Conceptual Opinions*: Questions that take advantage of user's opinions extracted by analyzing user's browsing history, read articles and email content.

Overall security of the approach depends on the popularity of the websites user visits, the popularity of the events user attends and the number of people who know about it, the possibility of randomly guessing an answer to opinion-based questions and several other factors. Security improvements can be reached by requiring users to answer multiple questions. It is also important to note that types of activity-based questions should be carefully analyzed and selected since the questions themselves can leak information about the user's activity.

Similar research suggests using dynamic security questions for fallback authentication on smartphones using their app usage, calls or text messages [6].

Another design addressing fallback authentication security and usability issues uses **location-based security questions** [7]. This approach resembles traditional security questions in the sense that it is based on personal information. However, it differs from the later in question types and its form of providing the answers. Questions focus on episodic memories such as "Where did your first kiss take place?" with location-based answers. In contrast with the traditional approach where users type answers as a text, here, users select a location on a map. The hypothesis behind this approach is that episodic memories are easier to remember than personal facts. Selecting the location, on the other hand, prevents issues such as repeatability and error-proneness. Experiments show that adversaries (both close friends/family and strangers) are unable to guess the answers with required accuracy most of the time.

Experiment results of the concept lead the authors to believe that the approach has the potential to replace traditional security questions in the future [7].

A limitation of this concept is that security questions should be thoroughly designed and evaluated.

An even more optimal solution proposes a combination of the last two approaches. It generates **dynamic location-based challenge questions** based on users' tracked locations. After tracking users' locations for an extended period of time, the algorithm detects locations that are more "interesting", i.e., are rarely visited and so are more likely to be remembered by the legitimate user and less likely to be guessed by an adversary, and asks different kinds of questions about them (e.g., locations visited at a certain date and the order and time gap between visiting several locations). To further improve the security, the approach applies Bayesian classifier to authenticate legitimate users based on their performance/success rate patterns [1].

The study concludes that an accurate design of the approach can significantly raise the overall security and usability of an authentication system.

3.2 Comparing security question types

Investigations show that question styles greatly affect users' performance in account recovery [1].

In this section, we compare above-discussed question types from security, usability and deployability points of view. *Security* considers robustness against attacks from adversaries such as strangers, close friends and family as well as brute force guessing attacks. Privacy concerns are also considered in the overall security level.

Usability is discussed by *memorability* and *ease-of-enrollment* points of view separately. In *ease-of-deployment* we consider how effortful the development and deployment process of the system is and how widely it can be deployed.

Figure 2 represents a comparison of personal-knowledge, activity-based, preference-based and location-based questions. The chart is created based on discussions from [2], [3], [6], [7], [8]. Note that user-chosen security questions are omitted since their overall security, usability and deployability are similar to predefined personal-knowledge questions. In addition, dynamically generated location-based questions are considered under activity-based questions category.

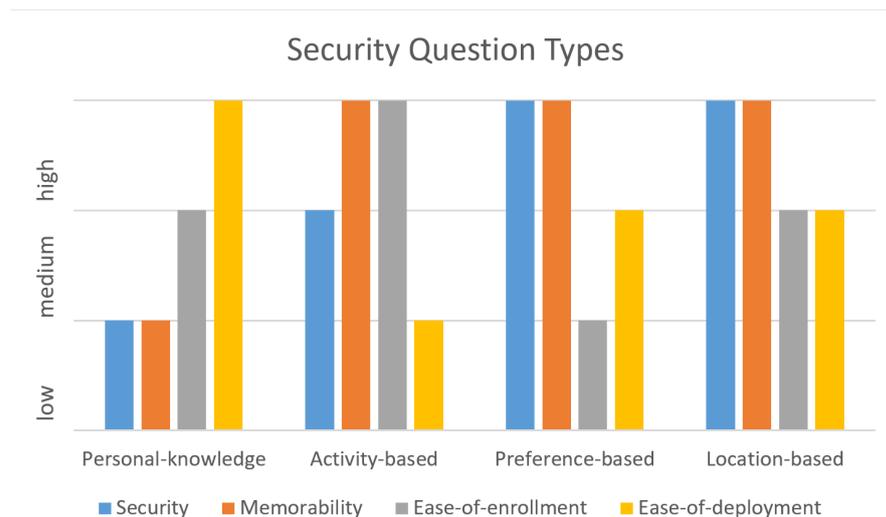


Figure 2. Security, Memorability, Ease-of-enrollment and Ease-of-deployment comparison of different types of security questions

As we can see, none of the discussed types is ideal.

The lowest *security* is achieved by personal-knowledge questions. Other three methods have higher *security*, activity-based questions being medium, because of privacy reasons (the information may be leaked by the questions and, based on the architecture, the service may need to track additional user activity). From *memorability* point of view, the last three types achieve higher results than personal knowledge questions. *Ease-of-enrollment* is the highest for activity-based questions since no user action

is required at that stage. Preference-based questions recede in *ease-of-enrollment* measure because of the need to answer a multitude of questions during enrollment. Considering the *ease-of-deployment*, personal-knowledge questions beat the others since the implementation, deployment and usage is rather straight forward and applicable to most of the systems. Activity-based questions have a lower rate of *ease-of-deployment* since some of the planned questions may not be relevant to the user or it may be complicated to find the answer.

Although based on the chart the highest overall grade is achieved by location-based questions, we believe that the nonintrusiveness of activity-based questions at the enrollment stage makes them the better option from the usability point of view. By accurately addressing privacy issues, this method can achieve advantageous results in security as well.

3.3 SMS and Email-based recovery

An alternative recovery method with higher reliability than security questions is authentication by email or SMS (Short Message Service). In this approach, a temporary password or a recovery URL is sent to a predefined email address or phone number. Figure 3 shows the success rate of SMS and email-based recovery compared to security questions [3]. Success rate of a fallback authentication mechanism is defined as the fraction or percentage of success among the total number of attempts to recover access using the mechanism.

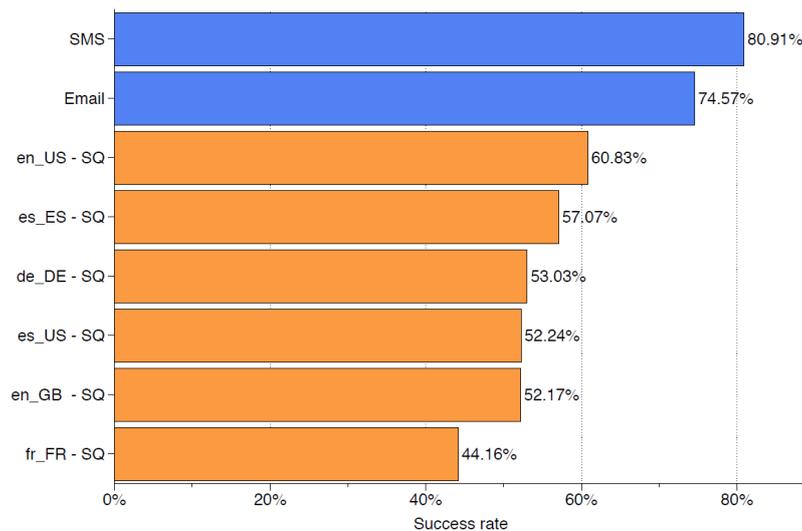


Figure 3. Success rate of different recovery methods [3]

The data was collected by Google based on recovery claims during a month. It shows that email increases the success rate of recovery by 13.74% over the most successful security questions. An even higher success rate is achieved by SMS-based recovery, due to some potential drawbacks of email-based recovery. The drawbacks include the likelihood of losing access to the email along with the other account (e.g., because of using the same password for both accounts or because of losing the browser cache where both passwords were stored).

In addition, both phone numbers and secondary emails may be recycled by service providers due to inactivity. Telecommunications service providers have different policies on phone number recycling that can typically be after 3-6 months of inactivity. Email providers generally recycle emails after longer periods of inactivity (e.g., Microsoft accounts are closed in 5 years of inactivity [4]). It was estimated that in 2014 7% of users' secondary emails were recycled, becoming available for anyone to register and use them again [3].

The figure also shows that cultural differences considerably impact the success rate of security questions. Different language/culture groups have different memorability for the same question and top-performing groups change depending on the question type. For example, the gap between US-English speaking users' and UK-English speaking users' recall for the question "Father's middle name?" reaches 10% after 12 months: 74.43% vs 64.12%. Shifts like this are hard to predict and impose a challenge for designing an international recovery system [3].

3.4 Social authentication

Another recently developed approach for fallback authentication is called social authentication [9]. Here users choose a set of trusted contacts that, in case of forgotten password, need to delegate them one-time secrets, which combined allow the users to reset their password.

In practice, social authentication can be realistically implemented in social network sites. Although the approach is promising, several studies found vulnerabilities in all social authentication mechanisms applied by social network sites. In particular, a relatively new attack against Facebook's social authentication mechanism, called Trusted Friend Attack, circumvents the security of the scheme. The attack is implemented by adding fake accounts to the victim's friends list and then using these accounts to obtain the secret code for the recovery [9].

3.5 Manually checking credentials

Some websites utilize the approach of manually checking the credentials. Here, the users need to either personally meet the authorities or send in a copy of their passport. This method can be useful when security demands are high and in corporate settings, where users are in near proximity. The disadvantage of it is that it scales badly in internet-wide services.

3.6 Machine Learning for authentication

Each of the above-discussed methods have their advantages/disadvantages, usability-security trade-offs and their use cases. However, all of them require some kind of user action. Some of the approaches are substantially more difficult (such as manually checking credentials) or more time-consuming (such as preference-based authentication).

With the accelerating growth of machine learning, a new approach of **implicit authentication** (also referred to as continuous authentication, active authentication and transparent authentication) is emerging [12]. Implicit authentication helps to minimize user friction while ensuring higher security for authentication.

Continuous authentication approaches include touch dynamics, face recognition, gait dynamics and behavior-based profiling. In case of using one of those approaches, methods that are otherwise used for primary authentication can be used for fallback authentication (e.g., a user may be required to enter a pin, in case of behavior recognition failure).

Furthermore, several studies suggest using machine learning for designing **risk-based authentication systems** [11]. This design suggests an improvement over static authentication methods by adjusting the method with the user's risk profile. Parameters affecting the level of risk include login time, location, IP address, number of failed attempts and some other contextual information. The approach provides higher level of security without harming usability, in contrast with the traditional approaches that make a trade-off between the two.

The conducted research particularly relates to primary authentication schemes. However, similar approaches can be applied to ensure the security and usability of fallback authentication schemes. Such an approach was represented in [1] that, in addition to checking the answers to location-based security questions, uses a machine learning classifier to authenticate real users based on their recall rate and performance patterns.

4 Summary

Achieving high level of security in fallback authentication remains a crucial aspect in designing an authentication system. In this paper, we reviewed several approaches that are widely applied in existing systems as well as approaches that are at the research stage. Overall, none of the approaches is perfect.

It is confirmed that security questions have poor memorability and their security level is far lower than that of user-chosen passwords [14].

Although SMS and Email-based recovery are proven to be more secure and are the preferred approach by large service providers such as Google, they still have their limitations and security questions remain useful [3].

Possible improvements to traditional personal-knowledge security questions (such as preference-based, activity-based and location-based questions) can increase their overall security and usability. In addition, security questions provide a satisfactory experience when the risk-level is considered low or if they are combined with other methods.

Less common approaches, that are beneficial in particular cases are social authentication and the manual check of credentials.

Finally, integrating machine learning to authentication system designs gives the ability to improve both security and usability without harming either.

References

- [1] Yusuf Albayram, Mohammad Maifi Hasan Khan, Athanasios Bamis, Sotirios Kentros, Nhan Nguyen, and Ruhua Jiang. Designing challenge questions for location-based authentication systems: a real-life study. *Human-centric Computing and Information Sciences*, 5(1):17, Jun 2015.
- [2] Anitra Babic, Huijun Xiong, Danfeng Yao, and Liviu Iftode. Building robust authentication systems with activity-based personal questions. pages 19–24, 01 2009.
- [3] Joseph Bonneau, Elie Bursztein, Ilan Caron, Rob Jackson, and Mike Williamson. Secrets, lies, and account recovery: Lessons from the use of personal knowledge questions at google. In *WWW'15 - Proceedings of the 22nd international conference on World Wide Web*, 2015.
- [4] Microsoft Corporation. Microsoft Services Agreement. <https://www.microsoft.com/en/servicesagreement/>, 2018. [Online; accessed 05-April-2019].
- [5] The OWASP Foundation. Cheatsheetseries. <https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets>, 2019.

- [6] Alina Hang, Alexander De Luca, and Heinrich Hussmann. I know what you did last week! do you?: Dynamic security questions for fallback authentication on smartphones. 04 2015.
- [7] Alina Hang, Alexander De Luca, Michael Richter, Matthew Smith, and Heinrich Hussmann. Where have you been? using location-based security questions for fallback authentication. 07 2015.
- [8] Markus Jakobsson, Erik Stolterman, Susanne Wetzel, and Liu Yang. Love and authentication. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 197–200, New York, NY, USA, 2008. ACM.
- [9] Ashar Javed, David Bletgen, Florian Kohlar, Markus Durmuth, and Jorg Schwenk. Secure fallback authentication and the trusted friend attack. pages 22–28, 06 2014.
- [10] Mike Just and David Aspinall. Personal choice and challenge questions: A security and usability assessment. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, pages 8:1–8:11, New York, NY, USA, 2009. ACM.
- [11] M. Misbahuddin, B. S. Bindhumadhava, and B. Dheeptha. Design of a risk based authentication system using machine learning techniques. In *2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld / SCALCOM / UIC / ATC / CBDCOM / IOP / SCI)*, pages 1–6, Aug 2017.
- [12] V. M. Patel, R. Chellappa, D. Chandra, and B. Barbello. Continuous user authentication on mobile devices: Recent progress and remaining challenges. *IEEE Signal Processing Magazine*, 33(4):49–61, July 2016.
- [13] Margaret Rouse. authentication. <https://searchsecurity.techtarget.com/definition/authentication>, 2018. [Online; accessed 27-February-2019].
- [14] S. Schechter, A. J. B. Brush, and S. Egelman. It's no secret. measuring the security and reliability of authentication via "secret" questions. In *2009 30th IEEE Symposium on Security and Privacy*, pages 375–390, May 2009.

Is Narrowband-IoT an efficient alternative for large-scale connected device systems?

Tolgahan Akgün

tolgahan.akgun@aalto.fi

Tutor: Pranvera Kortoçi

Abstract

In 3GPP Release 13, a narrowband system, called as Narrowband Internet of Things (NB-IoT), has been introduced to address the requirements of the Internet of Things. The new Narrowband system provides improved indoor coverage, low delay sensitivity, ultralow device cost, low power consumption and optimized network architecture. This system, based on Long Term Evolution (LTE) technology, supports most LTE functionalities with some limitations. This paper provides an overview of NB-IoT technology, including alternative technologies to NB-IoT in terms of efficiency for large-scale connected device systems.

***KEYWORDS:** Narrowband-IoT, IoT, LPWAN, Sigfox, LoRa, large-scale connected device systems*

1 Introduction

In the last decade, IoT technologies and IoT (Internet of Things) devices have evolved with the developing wireless connection technologies, low-cost sensors and SoC (system on chip) modules. IoT devices such as wearable gadgets, smart home devices and self driving cars, are key compo-

nents for the fourth industrial revolution. This vast amount of devices requires different communication systems to communicate with each other and the cloud technologies. The communication systems will need to support more than twenty-five billion connected devices, including IoT devices and mobile devices, by the year 2020 [19]. To support the large amount of connected devices, a number of candidate protocols have been designed, including ZigBee, Z-Wave, SigFox and LoRa.

Standards institutions have been working to design a connection protocol to fulfill the requirement of the IoT industry. A suitable connection infrastructure for IoT devices has couple of challenges which was addressed by 3GPP (3rd Generation Partnership Project) organization [11]. Some of the important challenges consist of low cost, limited computation power, support for densely located devices (40 per household) and long battery life (10 years) [11]. Specific application areas need different requirements. Some specific application areas may require low latency, high bandwidth. However, other may require low bandwidth. NB-IoT offers quality-of-service (QoS), low latency and higher bandwidth than LoRa and Sigfox at the expense of high power comparing to LoRa [19]. NB-IoT is more suitable for some specific application area such as real time monitoring systems. The problem arises here while deciding whether NB-IoT is efficient or not. Although, NB-IoT uses licensed frequency band which costs about 500 million dollars and it is expensive, in cases where low latency is required NB-IoT is preferred [19].

This paper analyses qualifications of large-scale connected systems. It describes advantages and drawbacks of NB-IoT for different application scenarios. It mentions requirements (e.g., bandwidth, battery life, latency) for large-scale connected device systems. In addition, different aspects of NB-IoT are compared with other communication standards such as LoRa, SigFox in terms of QoS, battery life, bandwidth and cost.

The paper is structured as follows, Section II describes the NB-IoT protocol, Section III describes the alternative technologies to NB-IoT. Section IV overviews the requirements for large-scale connected device systems. Finally, Section V concludes the paper.

2 NB-IoT

The low-power wide-area network (LPWAN) market has existed for about 10 years; it's not a new thing. The current technologies supporting this

market are fragmented and non-standardized. Therefore there are shortcomings like poor security, poor reliability high operational costs. Furthermore, the network deployment is complex [9].

For years, 3GPP focused on the cellular communications to supported voice service, human-oriented and mobile broadband service. 3GPP started a deep research on LPWAN technologies with promotion of other LPWAN technologies including LoRa and Sigfox. In Release 13, 3GPP set 5 objectives for for Machine-Type Communication services including enhanced indoor coverage, lower terminal complexity and cost, supported massive small-data terminals, higher energy efficiency and supported various latency features [8]. In June 2016, 3GPP introduced a new Narrow Band, LPWAN IoT technology to satisfy these objectives [19]. NB-IoT can operate with GSM (global system for mobile communications) and LTE (long-term evolution) under licensed frequency bands [13]. NB-IoT uses the single-carrier frequency division multiple access (FDMA) in the uplink and orthogonal FDMA (OFDMA) in the downlink, and employs the quadrature phase-shift keying modulation (QPSK) [13]. It can offer 100 kbps throughput in 200 kHz bandwidth [11]. Within this frequency bandwidth, it supports maximum 250 kbps uplink and 170 kbps downlink [17]. The maximum payload size for each message is 1600 bytes [13].

NB-IoT technology can be regarded as a new air interface from the protocol stack point of view, while being built on the well-established LTE infrastructure [13]. The unnecessary parts of LTE are omitted to fit the protocol for IoT applications. For example, broadcasting mechanism is kept to a minimum size and handover mechanism is omitted to consume less power [13] [10]. The main design principles behind NB-IoT are low cost, low latency, high coverage, long battery life, and easy deployment of a large number of devices [12], [4].

Figure 1 [8] shows the NB-IoT network architecture. The architecture consists of 5 parts:

- NB-IoT terminal: Each IoT device has access to the NB-IoT network as long as it has a valid SIM card;
- NB-IoT base station: It refers to the base station which is deployed by network operators;
- NB-IoT core network: Through the NB-IoT core network, NB-IoT base

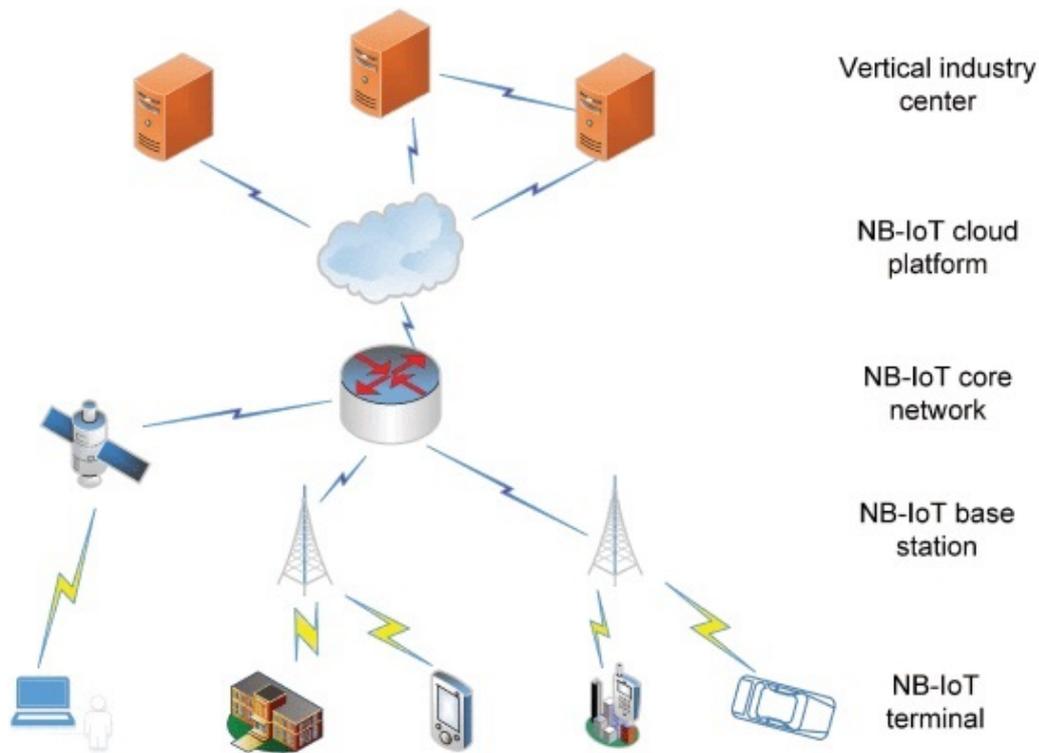


Figure 1. NB-IoT network architecture [8]

station can connect to NB-IoT cloud;

- NB-IoT cloud platform: NB-IoT cloud platform can process various services and results are forwarded to the vertical business center or NB-IoT terminal;
- Vertical industry center: It can store and process data generated by NB-IoT terminals and can control NB-IoT terminals.

NB-IoT can offer latency below 10 seconds and 10 years of battery life if it transmits 200-byte a day [11], [13]. NB-IoT is capable of delivering both IP and non-IP data [10]. The existing LTE hardware can support NB-IoT devices with a software upgrade [4]. Although using the same LTE hardware decreases the cost, the frequency band fee, which is as high as 500 million euro per MHz, may increase the cost of NB-IoT deployment [19]. A LTE cell can support up to 100,000 devices [16].

The improvement of NB-IoT has continued with Release 14 and Release 15 of the 3GPP [13] [18]. According to the 3GPP's current plan, NB-IoT will be improved to provide lower latency, lower power consumption and wider cover range [1].

	Sigfox	LoRaWAN	NB-IoT
Modulation	BPSK	CSS	QPSK
Frequency	Unlicensed ISM bands	Unlicensed ISM bands	Licensed LTE frequency
Max. data rate	100 bps	50 kbps	200 kbps
Max. messages/day	140 (UL), 4 (DL)	Unlimited	Unlimited
Range (urban)	10 km	5 km	1 km
Range (rural)	40 km	20 km	10 km
Authentication & encryption	Not supported	Yes (AES 128b)	Yes (LTE encryption)
Spectrum cost	Free	Free	>500 M€/MHz
Deployment cost	>4 K €/base station	>100€/gateway >1 K €/base station	>15 K €/base station
End device cost	<2€	3-5€	>20€
Battery life	10 years	10 years	10 years

Table 1. Overview of LPWAN technologies: Sigfox, LoRa, and NB-IoT [13], [7]

3 Alternative technologies to NB-IoT

NB-IoT is an LPWAN technology. The LPWAN technology include other protocols such as Sigfox, LoRa, RPMA (random phase multiple access) and Weightless [15], [5]. This paper focuses on NB-IoT, Sigfox and LoRa in terms of efficiency for large-scale connected devices systems. Table 1 presents an overview of LPWAN technologies in various aspects.

3.1 Sigfox

Sigfox is an LPWAN technology which is developed by the Sigfox company.¹ It operates in the unlicensed ISM (industrial, scientific and medical) band, for example, 868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia. It offers maximum throughput of 100 bps in ultra-narrow band (100 Hz) subGHz ISM band. Sigfox can offer very low power consumption, high receiver sensitivity, and low-cost antenna design due to its ultra-narrow band [13].

In initial design, Sigfox could support only uplink connection. Later it evolved to bidirectional technology. The uplink is limited to 140 messages per day, whereas the downlink is limited to 4 messages per day. The maximum payload size for an uplink message is limited to 12 bytes and the maximum payload size for a downlink message is limited to 8 bytes [13]. The uplink communication reliability of the Sigfox network relies on spa-

¹<https://www.sigfox.com/en/sigfox-iot-technology-overview>

tial diversity coupled with the time and frequency diversity of the radio frame repetitions. ¹ Each message is transmitted multiple times over different frequency channels from end-devices [13].

Sigfox offers wider cover range compared to LoRa and NB-IoT. A base station (range >40 km) is capable of covering an entire city.

3.2 LoRa

LoRa, which stands for “Long Range”, is a long-range wireless communication system, introduced by the LoRa Alliance in 2015 [3]. LoRa operates between 137 MHz to 1020 MHz. Therefore it operates in both licensed and unlicensed bands i.e., 868 MHz in Europe, 915 MHz in North America, and 433 MHz in Asia. [5] [13]. However, it is often deployed in ISM band, which is unlicensed [5].

LoRa can offer network throughput between 300 bps and 50 kbps depending on the environmental conditions. The maximum payload size of a message is limited to 243 bytes. LoRa is based on proprietary modulation. Although the modulation mechanism is proprietary, the LoRa wide area network (LoRaWAN) is an open standart being developed by the LoRa Alliance [3]. LoRaWAN with LoRa gateway offers local are network operations and LoRaWAN with base station offers public network operations [14]. In LoRaWAN, each transmitted message is recieved by all the base station within the range, then received duplicate messages are filtered by the backend system. Moreover, receiving the transmitted messages by multiple base stations avoids the handover mechanism in LoRaWAN network. This mechanism improves the successfully received messages ratio while incresing the deployment cost [13]. LoRa offers wider cover range compared to NB-IoT. For example, in Belgium, a country with total area of 30500 km^2 can be covered with only 7 base stations [19].

End-devices in LoRaWAN may have different requirements. In order to optimize the different end-device requirements, LoRaWAN supports 3 different classes, named Class A, Class B, Class C. The Class A, which is the most energy efficient class, is for battery powered sensors. The Class B is for battery powered actuators. The Class B provides latency controlled downlink. The Class C is for main powered actuators. The Class C provides no latency for downlink [2].

4 Requirement analysis for large-scale connected systems

Requirements in IoT differ from system to system. The IoT system defines its own requirements. Although latency is critical in a real-time monitoring systems, in smart farming, latency is critical [13]. Therefore, efficiency evaluation of the LPWAN technology depends on the application design criterias and high-priority requirement of the application.

4.1 Requirements in IoT

Low power consumption

Low power consumption is one of the key design principles in most IoT connection protocols [20]. Usually, IoT devices are battery-powered. Therefore, such devices are expected to operate without battery replacement for years.

Scalability

Currently, 16 billion connected devices exist worldwide [4]. Ericsson estimates that by 2021 there will be 28 billion connected devices [4]. In this rapidly growing industry, scalability is a critical requirement including the platform's load balancing capabilities for optimal performance of the cloud services.

Low latency

Low latency is a fundamental requirement in real-time monitoring systems such as electric metering. In fact, companies may need real-time data monitoring to make immediate decisions and take precautions according to the real time data [13].

Low cost

Decreasing the operational costs is the starting point of some IoT projects such as smart metering systems [6]. In a large-scale device ecosystem, there might be thousands of end devices. While calculating the cost, various factors should be considered including the spectrum, deployment and end-device cost.

Ease of deployment

Ease of deployment of the IoT devices is a critical requirement as Ericsson estimates that there will be 28 billion connected device by 2021. Additionally, ease of the deployment reduces the cost of the system.

Coverage

IoT applications such as smart farming and disaster management systems operate in rural area. The applications which is deployed in a rural area, require wide coverage area.

4.2 Comparison in terms of efficiency

Requirements in a large-scale connected system have different priorities as aforementioned. Priority of the requirements determine the efficiency of the LPWAN technologies. None of the LPWAN technologies, which are mentioned in this paper, can fulfill all requirements as Figure 2 shows [13].

In order to compare efficiency of the technologies, each large-scale connected system has to be analyzed separately. Large-scale connected systems can be categorized in terms of their high-priority requirement as follows:

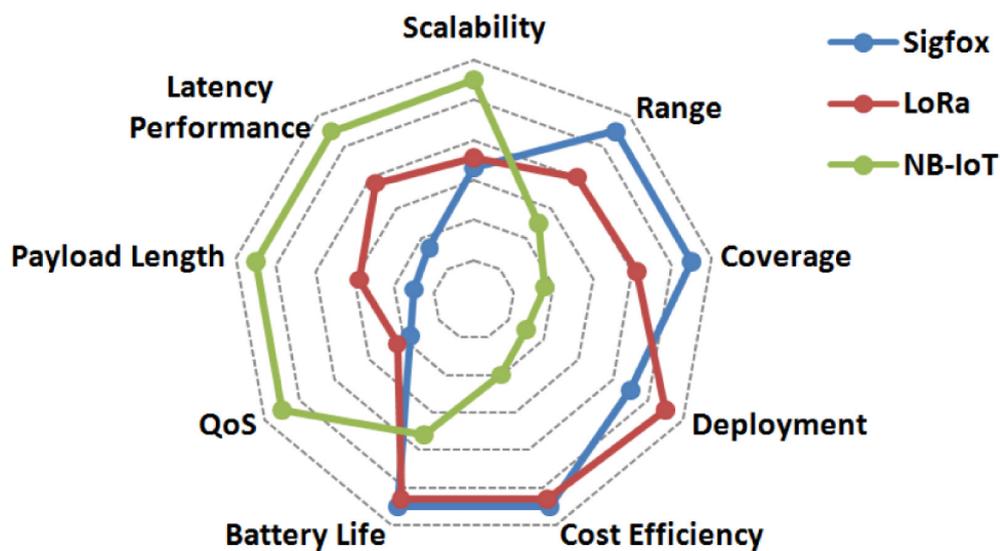


Figure 2. Comparison of Sigfox, LoRa, and NB-IoT in different aspects [13]

– *Real-time monitoring*: Real-time monitoring systems such as electric metering, intelligent traffic information system require high network throughput and QoS. Typically, these systems do not require low power consumptions as the systems are near to power sources. Only NB-IoT can offer high network throughput and QoS. Therefore, for such systems, NB-IoT fits best. Unlike the real time monitoring systems, various systems such as smart farming, smart building do not require QoS and high network throughput. Therefore, Sigfox and LoRa fit better for this type of applications.

- *Rapidly growing*: Rapidly growing connected device systems require scalability. NB-IoT is capable of handling up to 100,000 end devices per cell. Unlike NB-IoT, Sigfox and LoRa are capable of handling up to 50,000 per cell [13]. Since NB-IoT is more scalable compared to Sigfox and LoRa, for rapidly growing systems, NB-IoT is more efficient.

- *Gradually growing*: In gradually growing connected device systems such as remote water metering systems in places where have low population growth rate, scalability is not a fundamental requirement. Therefore, Sigfox and LoRa is more efficient.

- *Location, density*: Location of the system defines need for coverage area. In a connected device system such as smart farming where devices are located sparsely in a wide area, the protocol should offer wide range transmission. As mentioned before, Belgium can be covered with only 7 LoRa base stations. Therefore, for a system which end devices are located sparsely in a wide area, Sigfox and LoRa is more efficient. Since a NB-IoT cell can handle up to 100,000 end devices, the systems which end devices are located densely, NB-IoT is more efficient.

- *Low cost*: Cost effectiveness is one of the fundamental design principles in all systems. As Table 1 shows, NB-IoT is not cost efficient compared to the other protocols. For high latency tolerable low-cost systems, Sigfox and LoRa are more efficient.

5 Conclusion

This paper has summarized Sigfox, LoRa, and NB-IoT and discussed efficiency of NB-IoT for large-scale connected device systems. Each technology has its superiority in different features. NB-IoT provides very low latency and high quality of service with high cost. Unlike NB-IoT, LoRa and Sigfox provide wide coverage area and very long battery lifetime with low cost. There is a trade-off between the cost of the technology and its features. Therefore, efficiency of NB-IoT for a large-scale connected device systems should be evaluated according to the requirement priority of the system.

Despite the synthetic tests, the lack of long-life real world NB-IoT de-

ployment leaves open questions on the performance in the real-world conditions. The NB-IoT and its alternative technologies are still evolving. In the future, the scenario may change in favor of NB-IoT with the wide deployment of 5th generation (5G) mobile communication technology.

References

- [1] 3GPP Work Item Description Further NB-IoT enhancements RP-182389, December 2018.
- [2] LoRa Alliance. Technical marketing workgroup 1.0, "lorawan what is it. A technical overview of LoRa and LoRaWAN", 24, 2015.
- [3] Aloÿs Augustin, Jiazi Yi, Thomas Clausen, and William Townsley. A study of LoRa: Long range & low power networks for the internet of things. *Sensors*, 16(9):1466, sep 2016.
- [4] Yihewnew Dagne Beyene, Riku Jantti, Olav Tirkkonen, Kalle Ruttik, Sassan Iraji, Anna Larmo, Tuomas Tirronen, , and Johan Torsner. NB-IoT technology overview and experience from cloud-RAN implementation. *IEEE Wireless Communications*, 24(3):26–32, jun 2017.
- [5] Martin C. Bor, Utz Roedig, Thiemo Voigt, and Juan M. Alonso. Do LoRa low-power wide-area networks scale? In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems - MSWiM '16*. ACM Press, 2016.
- [6] Richard E. Brown. Impact of smart grid on distribution system design. In *2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century*. IEEE, jul 2008.
- [7] Smilty Chacko and Mr. Deepu Job. Security mechanisms and vulnerabilities in LPWAN. *IOP Conference Series: Materials Science and Engineering*, 396:012027, aug 2018.
- [8] Min Chen, Yiming Miao, Yixue Hao, and Kai Hwang. Narrow band internet of things. *IEEE Access*, 5:20557–20577, 2017.
- [9] Huawei. NB-IoT - Enabling New Business Opportunities - Whitepaper. Technical report, Huawei Technologies CO., LTD, 2015.
- [10] D. Raddino J. Schlien. Narrowband internet of things - whitepaper. Technical report, Rohde & Schwarz GmbH & Co. KG, 2016.
- [11] Mads Lauridsen, Istvan Z. Kovacs, Preben Mogensen, Mads Sorensen, and Steffen Holst. Coverage and capacity analysis of LTE-m and NB-IoT in a rural area. In *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*. IEEE, sep 2016.
- [12] Yuke Li, Xiang Cheng, Yang Cao, Dexin Wang, and Liuqing Yang. Smart choice for the smart grid: Narrowband internet of things (NB-IoT). *IEEE Internet of Things Journal*, 5(3):1505–1515, jun 2018.

- [13] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express*, jan 2018.
- [14] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. Overview of cellular LPWAN technologies for IoT deployment: Sigfox, LoRaWAN, and NB-IoT. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, mar 2018.
- [15] Juha Petäjäjärvi, Konstantin Mikhaylov, Matti Hamalainen, and Jari Iinatti. Evaluation of LoRa LPWAN technology for remote health and wellbeing monitoring. In *2016 10th International Symposium on Medical Information and Communication Technology (ISMICT)*. IEEE, mar 2016.
- [16] Juha Petäjäjärvi, Konstantin Mikhaylov, Marko Pettissalo, Janne Janhunen, and Jari Iinatti. Performance of a low-power wide-area network based on LoRa technology: Doppler robustness, scalability, and coverage. *International Journal of Distributed Sensor Networks*, 13(3):155014771769941, mar 2017.
- [17] Lagusson Petter and Nordlöf Johanna. A study of low-power wide-area networks and an in-depth study of the LoRaWAN standard. Master's thesis, KTH, Machine Design (Dept.), 2017.
- [18] Rapeepat Ratasuk, Nitin Mangalvedhe, Yanji Zhang, Michel Robert, and Jussi-Pekka Koskinen. Overview of narrowband IoT in LTE rel-13. In *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE, oct 2016.
- [19] Rashmi Sharan Sinha, Yiqiao Wei, and Seung-Hoon Hwang. A survey on LPWA technology: LoRa and NB-IoT. *ICT Express*, 3(1):14–21, mar 2017.
- [20] Reza Tadayoni, Anders Henten, and Morten Falch. Internet of things - the battle of standards. In *2017 Internet of Things Business Models, Users, and Networks*. IEEE, nov 2017.

Survey on touch dynamics based continuous authentication

Matti Mäki-Kihniä

matti.maki-kihnia@aalto.fi

Tutor: Sanna Suoranta

Abstract

Smart phone security often is enforced by entry authentication mechanisms, implemented with a PIN code or pattern. After the initial authentication, no further steps are taken to authenticate the user, generally allowing wide access to personal data and services. Continuous authentication, based on touch gestures, has been suggested as an option for enforcing smart phone security. Gesture based continuous authentication allows authenticating the user in real time, based on their interactions with the device, thus making unauthorized usage more challenging. In this article, five different touch gesture based continuous authentication approaches are reviewed and compared, with focus on differences in data acquisition, feature selection and classification. The survey finds that touch interaction-based authentication is a feasible, however, questions about the implementation, such as deployment and model training, still remain open.

KEYWORDS: *Continuous authentication, touch dynamics*

1 Introduction

It could be assumed that most of the readers of this review will have a smartphone in their possession. This assumption is backed by Statistics Finland data [14] which states that 80 percent of the population of Finland own a smartphone. Smartphones are commonly regarded personal and they hold sensitive data, as well as access to different services, due to which they are likely to be secured in some way. The security is established through entry authentication: an approach where user authentication is performed once at the start of a usage session to unlock the device, and no further authentication is done before the next session starts.

Entry authentication is often implemented with something that the user knows, which can be a PIN code, pattern or a passphrase. The selected authentication method can be optionally augmented with one or more forms of biometric authentication, such as fingerprint or facial features. Even when biometric authentication is enabled, the secret that the user selected can be used for unlocking the device. Thus, the biometrics used in entry authentication could be considered to rather improve the user experience of unlocking a device, than to make the device more secure. Furthermore, it has been shown that users choose weak passwords and PIN codes [3] consequently making entry authentication even weaker.

Because the user's secret can be always used to unlock the device, it is the unique element keeping an attacker away from the user's data, in case the device is stolen. In order to prevent unauthorized usage, even when the initial entry authentication is compromised, researchers have proposed a number of continuous authentication methods that evaluate the authenticity of the user during all interaction with the device and lock it in case of unauthorized usage. The key difference between the proposed methods and entry authentication is that the former are non-obtrusive and real-time, and in order to be so, the methods rely on biometrics or user behavior, both of which are present in all interaction with a device.

Vishal et al. [15] review different approaches to continuous authentication in 2016, covering what was then the most recent research. They divide continuous authentication into four groups: touch dynamics, facial recognition, gait dynamics, and behavioral profiling. Each of the categories shares the same approach: firstly, during a training or enrollment period, user interaction is recorded to gather data to distinguish the user. Once a sufficient number of distinctive features are gathered, a classifier

Study	Year	Abbreviation	Users
Lin et al. [8]	2013	Lin13	75
Serwadda et al. [13]	2013	Ser13	190
Feng et al. [7]	2012	Fen12	40
Frank et al. [10]	2013	Fra13	41
Ali et al. [16]	2016	Ali16	6

Table 1. Reviewed articles and their abbreviations

is trained based on them. The classifier is then provided with user input, possibly in real time, to be able to determine if a certain user is performing those inputs.

Upon training a classifier with sufficient performance, the user’s interactions can then be supplied to it as input and actions can be performed on the classification output. Continuous authentication could be summed up as extracting features from user interactions in real time, providing them as input to the classifier and acting on the classification results. The actions could be, for example, to prompt the user for a password or in the case of successful classification, do nothing. Even though fingerprint readers are commonplace in smartphones, continuous authentication methods are not available to customer devices.

This review assesses five research articles on continuous authentication implemented through the use of touch dynamics. The aim is to highlight the discrepancies in both the settings and methods applied in the research. The review first compares the differences in the selection of touch dynamics properties and data acquisition techniques. Finally, the chosen classification techniques, and how each approach evaluated their results, are compared. By the end, the reader should have a solid understanding of the methods applied in the current research on touch dynamics for continuous authentication.

2 Continuous authentication using touch dynamics

Table 1 presents five continuous authentication articles selected for this review. From here on onwards they will be referred to by the abbreviations listed in the table. Four of the selected research articles, Lin13, Ser13, Fen12 and Fra13, are from Vishal et al.’s [15] overview while the fifth, Ali16, was selected as it proposes using accelerometer data in con-

junction with touch dynamics to further improve the accuracy of authentication [16]. The four articles selected from Vishal et al.'s [15] overview are from 2012 and 2013 while Ali et al.'s [16] article is from 2016. There exists newer publicized research on the topic, such as Iuzbashev et al.'s [9] article from 2018 and Putri et al. [12] article from 2016; however, they are not a part of this review, as they provide little or no information on the properties that are compared here.

2.1 Data acquisition

All the reviewed articles implemented their concept only on the Android platform, giving no information as to why iOS platform has not been considered. Searches for similar concepts designed for the Apple's platform, on research databases have not yielded results. Missing tools are unlikely to be a cause for the lack of implementations, as APIs for gathering touch data are available both on the Android [6] and iOS platforms [5]. Therefore, it remains unclear why similar implementations, to the articles reviewed here, have not yet been implemented on Apple's platform. One reason could be that, as the iOS build tools are available only on the MacOS, all developers would be required to have a Macintosh.

Gathering touch data on the Android platform can be done through either the MotionEvent API [6], or the by capturing touch data on the kernel level. Lin13 takes the latter approach, while the other articles employ the MotionEvent API. The difference between the two options is that the MotionEvent API only allows capturing touch events from the specific application in which it has been implemented, while intercepting the touch data on the kernel level allows gathering all events. Thus, the approaches employing the MotionEvent API have to create a specific application for data gathering, which might cause test users to alter their behavior from normal. While the kernel level approach allows gathering data from all applications, it requires root access to be installed.

In addition to different data acquisition approaches, each of the reviewed articles has different requirements for the amount of test users, as well as the amount of data gathered from those users. The number of users, with which each of the implementations have been tested, ranges from 6 [16] to 190 [13]. Lin13, the approach in which data is collected on the kernel level, gave out devices to the test users to use for days, in order to ensure that the data would be reflect realistic situations [8]. Ali16 has another approach, in which the users were asked to install the data col-

lection application on their devices, and proceed to use it multiple times over five days to generate data [16]. While Lin13 mentions that all users were given the same kind of a phone [8], Ali16 does not mention whether or not the test users' devices were the same model [16], which raises a question whether touch data from different devices is comparable.

In the other three approaches, Ser13, Fen12 and Fra13, the test users were doing tasks on applications created specifically for the purpose of recording touch data [13, 7, 10]. The tests were performed in a testing environment where the users were invited.

The commonality between the articles, besides Fen12, is that they recorded data from different usage sessions, even days apart. The reason for gathering data on multiple different sessions is not explicitly mentioned, but it could be speculated that this is to ensure variety of data, in case user behavior varies between the sessions.

Ser13 goes into most detail in explaining the quantity of data they required for performing classification. The specified amount is 80 swipes, both vertical and horizontal, in two device orientations. This requirement is made to ensure that the classifier will not have any performance bias caused by varying number of training data for different users [13]. For Lin13 the amount of required swipes is 150, both vertical and horizontal, as well as 300 tap gestures, however no specific reason behind these figures is given [8].

2.2 Touch dynamics

Users generally interact with a touch screen of a mobile device by performing gestures with their fingers. The gestures can be split into two categories: single finger gestures and multi-touch gestures. Single finger gestures include, for example, horizontal and vertical swiping, while examples of multi-touch gestures include pinching and spreading. The finger with which a gesture is performed is not directly relevant, as different users use different fingers to perform the same gestures [8], but instead, the shared properties of gestures, such as touch pressure and starting and ending positions, are significant. Analyzing these properties enables distinguishing the user who is performing them.

All the articles build their recognition functionality on swiping features, but each of them has differences regarding what additional features are included. The different gestures, and other data sources, have been listed in Table 2. For example, four out of the five articles, Lin13, Ser13, Fra13,

Study	Gestures	Other
Lin13	Swipe, tap	-
Ser13	Swipe	Device orientation
Fen12	Swipe, pinch	Sensor glove
Fra13	Swipe	Device orientation
Ali16	Swipe, tap	Accelerometer data

Table 2. Gestures used for touch analytics

Ali16, apply metrics only from gestures performed with one finger [8, 13, 10, 16], and, out of these four, Lin13 and Ali16 are the only ones which also employ data from taps. The decision not to use taps is not necessarily described, but, for example, Fra13 mentions that it provides “too few features to be discriminative for users” [10], moreover, it has been included in Lin13, it is only used as an “auxiliary gesture” [8], as it does not provide any information on finger movement.

All approaches employing only single finger gestures have the same reason for this decision, which is that these gestures represent the majority of interactions that a user performs, thus making multi-touch gestures too rare to be used for continuous authentication. The argument is backed by data collected in Lin13, which shows that 88.8 percent of gestures are performed with a single finger [8]. This information is noteworthy, as it was collected from normal user interactions by capturing the data from the kernel level [8], instead of using the MotionEvent API, which only allows each app to record its own interactions. It could be concluded, that gathering the data unobtrusively, as compared to laboratory conditions, would lead to more genuine data.

The approach of Fen12 is the only one to also incorporate multi-touch features, as in addition to swiping, it also takes into account zooming in and zooming out [7]. No specific reasoning for incorporating multi-touch is given; however, it should be noted that this research also involves data from a digital sensor glove. The glove provided accelerometer data for two of the user’s fingers and was used to enhance existing data. Accelerometer information is also used in Ali16, to recognize the position and orientation of the user’s device, thus increasing the accuracy of the results [16].

2.3 Touch properties

As mentioned in the previous chapter, the distinctiveness of gestures is in their properties. Different features can be derived from the gesture's main properties, including starting and ending positions as well as the pressure on and in between these points. The level of detail of feature extraction and selection in each of the reviewed articles varies, which makes comparison difficult. Each of the different approaches is reviewed below with Fra13 used as a reference point, as it provides the most in-depth analysis of the features.

Fra13 extracted 30 features from each of the gathered gestures, including a non-touch feature, phone orientation. Aside from the main properties, the list includes interesting features, such as mid-stroke area covered and median velocity of the last five points. The latter feature distinguishes users' swiping habits by describing whether they lift off their finger at the end of a swipe or keep it on the screen. The informativeness of each feature was rated, with mid-stroke area covered and mid-stroke pressure scoring among the highest. However, the authors also mention that more information can be discovered by combining complementary features. Further analysis was done by calculating the correlation coefficients of all feature pairs and using the information to remove redundant features from the classification. [10]

Lin13 propose using 13 slide features and three tap features. The assessment of a good metric is derived by comparing how well it can be used to distinguish a single user. This assessment is made by calculating a distribution of that metric from each test user's data, and then comparing those distributions with each other; if most of the distributions are similar, then the feature is left out. The conclusion of the assessment is that pressure related features, including average pressure and first touch pressure, are not good for distinguishing users. This is contrary to Fra13, which lists a pressure metric as one of the top features. Similarly, to Fra13 the correlation between pairs was measured, however, the results did not result in leaving out any metrics. This comparison reveals interesting correlations, such as the similarity between first touch area and the average touch area, which Lin13 speculates to be due to users' swiping on the same area of the screen [8].

Ser13 declare 28 different features of a stroke. The features are similar to those both Lin13 and Fra13 use: the coordinates of the gesture, dif-

ferent measurements of its velocity and acceleration as well as pressure. Ser13 hypothesizes that gestures are performed differently based on the orientation of the device, and thus they should not be compared between each other [13]. This estimation would appear logical, as smartphone screens are commonly rectangular, and thus the maximum distance of a swiping motion is affected by its orientation.

According to Fen12 each touch gesture includes 53 features, but these features are not listed and only the aforementioned properties are mentioned. On top of the common features, also multi-touch features are considered, yet no especially imaginative features are highlighted [7]. Fen12 does not provide reasoning or data about feature selection or whether features of multi-touch gestures are more informative than those of single finger gestures.

Lastly, Ali16 fails to mention any features or feature selection for that matter. The agenda is aimed towards distinguishing differences between the accuracy of predicting users from based on different gestures [16].

It could be speculated that due to the proof of concept nature of the approaches, attention to detail has been aimed more towards analyzing the outcomes of the implementations rather than describing the means of feature selection.

3 Classification methods and performance

Authenticating a user based on their touch properties is a matter of analyzing given touch features and then matching those to a certain user. The considerable number of features, and their variance, in a touch gesture makes it unfeasible to manually write rules for effectively matching users, which is why all the articles approach the task as a classification problem. To solve a classification problem, a classifier has to be first trained with training data, during which it will learn to associate certain properties with specific labels. After the training period, the classifier should be able to match input, that has not been present in the training data to a certain class. Examples of classes could be, for example, a specific user and all other users, the index finger or thumb. The selection of a classifier depends on the given problem domain. However, different classifiers can be applied to the same problem as seen in this chapter.

The articles estimate the performance of the chosen classifiers with one or more of the following metrics: False Acceptance Rate (FAR), False Re-

Study	Classifier	EER (%)
Lin13	SVM [8]	3.0 *
Ser13	8 classifiers [13]	10.5 - 42.0
Fen12	Random Forest, J48, Bayes Net [7]	7.8 - 13.0 * **
Fra13	kNN, SVM [10]	0.0 - 4.0
Ali16	LibSVM [16]	N/A

Table 3. The chosen classifiers. * Values taken from [13] as the original author does not provide numerical percentages. ** The percentages include the usage of the sensor glove.

jection Rate (FRR) or Equal Error Rate (EER) [10]. FAR represents the percentage of swipes that an intruder performs, which are then classified as legitimate user’s input. FRR is the opposite: it represents the percentage of swipes that the legitimate users perform, which are not correctly classified. Finally, EER is the percent at which both the FRR and FAR of a classifier are equal.

Table 3 presents the different classifiers used in the articles. The most popular classifiers among the approaches are the Support Vector Machine (SVM) classifiers. The SVM classifiers are binary classifiers, meaning that they can be used for two-class classification [11]. SVM requires a training phase, after which it discards the individual data points from the training period, and only stores their generalizations by which the classifications are later derived [10].

The k-nearest neighbors algorithm (kNN) is used as one of the classifiers for both Ser13 and Fra13. The kNN algorithm assigns a class to an unclassified input based on the which class data points have the greatest presence in the new point’s neighborhood [4]. Unlike the SVM, the kNN algorithm does not require a specific training period, as it only needs access to the existing data points and is able to place new inputs in the same feature space [10].

As observed by both Fra13 and Ser13, the kNN classifier performed worse in comparison to the SVM in all scenarios [10, 13]. The results cannot be compared between the studies, but both studies have described their results accurately. For Ser13, the equal error rates (EER) for the kNN classifier ranged from 14 to 27 percent, depending on the orientation of the device, as well as the orientation of the strokes [13]. Fra13 reports EER percentages between 0 and 4 for both scrolling and horizontal classifiers, and generally even lower for the SVM classifier [10].

Ser13 benchmarks eight different classifiers to establish the one with the best performance in the given conditions. Interestingly even the best performing classifier, logistic regression, exceeds double-digit percentage EERs on all scenarios, while the worst performing classifier, J48, performs merely better than guessing, as it reaches EERs ranging from 30 to above 40 percent [13]. The performance of the J48 is also the worst out of those evaluated in Fen12, however with considerably better rates than those observed by Ser13 [7], as can be noticed from Table 3.

Fen12 use one classifier for their approach, with the selected setting they are able to reach classification accuracies ranging from 80 percent to above 95 percent, depending on the device orientation and given gesture [7]. It is noted that the two best performing gestures, in terms of EER, are sliding left and sliding up [7]. Ali16 reports their observations on the gesture level as well; their results being similar to Fen12 [16]. It should be noted, that even though Ali16 is strongly focused on distinction between which finger is used for a gesture, the aforementioned observation is done without including labels for the user's fingers [16].

Perhaps the only solid conclusion, that can be drawn from the results provided by the articles, is that more reliable classification results can be derived by including both sensor data and touch data, as shown by both Fen12 [7] and Ali16 [16]. Fen12 notes that by combining external sensor data from the sensor glove used in the research, they were able to drop FAR rates from 11.96% to 2.15% and FRR rates from 8.53% to 1.63% [7]. Meanwhile, Ali16 mentions that the average accuracy for predicting the finger, with which a gesture is performed, increased from roughly 84% to close to 98% by augmenting the touch data with sensor data.

Performing classification in real-time on-device will cause an overhead on the system. An example of the overhead is given in Lin13, where it is mentioned that filtering and extracting features of 427 gestures takes 648 milliseconds on a low-end device [8]. If such computation could be done on the background, without causing any noticeable decrease in performance for the user, the latency could be considered acceptable for a consumer setting. Retraining a classifier to adjust to the changing touch dynamics' patterns could be a more resource intensive task, thus required to be executed on a separate server. Lin13 proposes a system which performs classification on the user's device, while performing occasional retraining of a model on a server [8]. Additional attention is given to the security aspect, where anonymous group messaging and message shuffling would

be applied to enforce anonymity between touch data and a user [8]. Other approaches do not mention how the training would be done in practice.

4 Results

Each of the reviewed approaches provide a concept which works for continuous authentication, at least within the limitations and requirements set in each of the articles. The fact that the reviewed approaches vary in all of the inspected aspects could be considered a strong indication that promising classification results could be achieved with many different approaches.

Even though the evaluations of the classification results are positive, many questions regarding practical implementation and feasibility still remain open. An undecided question is which features to use; for example, should taps and pressure be evaluated? The most descriptive touch gestures should be established so that further valid datasets could be generated, and different approaches could be evaluated based on similar inputs. Ser13 indicates to have given free access to their data set [13], however, it does not appear to be available any longer, which means that future approaches have to, once again, gather new datasets.

More issues arise with the implementation of such systems, starting with gathering both training and evaluation data. As mentioned previously, barring rooted and jailbroken devices, the Android and iOS platforms only allow gathering data through their respective APIs. This limitation means that all interaction with a device should happen through one application, which records the touch interactions through the APIs. Implementing such data collection applications might be possible with sandboxing platforms, such as Boxify [1] or NJAS [2].

An issue for continuous authentication based on classifiers that need training is the apparent need for retraining of the model on certain intervals. This issue has been highlighted by Fra13, where it is mentioned that the authentication difficulty increases based on the temporal distance to the initial training data [10]. The retraining frequency would likely depend on the accuracy requirements of the system, which brings up the question of whether or not to implement the retraining on the end user's device or a separate server.

Due to the issues highlighted in this section, it remains an open question how unobtrusive continuous authentication based on touch dynamics

could be implemented. It is, however, likely that a user could be authenticated based on their touch dynamics for other purposes that do not require continuous authentication. For example, for the Fra13 approach, it takes 11 swipes, in each direction, for an initial classification to be made [10], while for the Lin13 approach a minimum of 13 swipes is required for a classification [8]. This means that it could be possible to request a user to perform a certain amount of gestures, in order to authenticate the user on demand, for example, for digital signature purposes.

5 Conclusion

The summary reviewed five different articles on continuous authentication using touch gestures. The focus of the review was on highlighting the differences between the approaches, focusing on data acquisition, feature selection and classification. The discovered differences were highlighted in a manner that enables the user to critically assess the variations between the approaches. While it was established that a user can be identified based on their interaction with a touch screen, none of the proposed solutions have addressed the difficulties of deploying such a system. Future work could involve a more profound investigation into any of the areas reviewed here.

References

- [1] Michael Backes, Sven Bugiel, Christian Hammer, Oliver Schranz, and Philipp Von Styp-Rekowsky. Boxify: Full-fledged app sandboxing for stock android. In *Proceedings of the 24th USENIX Conference on Security Symposium*, 2015.
- [2] Antonio Bianchi, Yanick Fratantonio, Christopher Kruegel, and Giovanni Vigna. Njas: Sandboxing unmodified applications in non-rooted devices running stock android. In *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, 2015.
- [3] Joseph Bonneau. *Guessing human-chosen secrets*. PhD thesis, University of Cambridge, May 2012.
- [4] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 1967.
- [5] Apple developers documentation. *UITouch*. 2019-02-12.
- [6] Android Developers documentation. *MotionEvent*. 2019-02-12.
- [7] T. Feng, Z. Liu, K. Kwon, W. Shi, B. Carpunar, Y. Jiang, and N. Nguyen. Continuous mobile authentication using touchscreen gestures. In *2012 IEEE*

Conference on Technologies for Homeland Security (HST), 2012.

- [8] Xinxin Zhao Lingjun Li and Guoliang Xue. Unobservable re-authentication for smartphones. In *20th Annual Network and Distributed System Security Symposium*, 5 2013.
- [9] A. V. Luzbashev, A. I. Filippov, and K. G. Kogos. Continuous user authentication in mobile phone browser based on gesture characteristics. In *2018 Second World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*.
- [10] M. Frank and R. Biedert and E. Ma and I. Martinovic and D. Song. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE Transactions on Information Forensics and Security*, 2013.
- [11] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.
- [12] A. N. Putri, Y. D. W. Asnar, and S. Akbar. A continuous fusion authentication for android based on keystroke dynamics and touch gesture. In *2016 International Conference on Data and Software Engineering (ICoDSE)*, 2016.
- [13] A. Serwadda, V. V. Phoha, and Z. Wang. Which verifiers work?: A benchmark evaluation of touch-based authentication algorithms. In *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, 2013.
- [14] Tilastokeskus. Suomen virallinen tilasto (svt): Väestön tieto- ja viestintätekniikan käyttö [verkkójulkaisu]. 2019-02-10.
- [15] Vishal M. Patel and Rama Chellappa and Deepak Chandra and Brandon Barbello. Continuous User Authentication on Mobile Devices: Recent progress and remaining challenges. In *IEEE Signal Processing Magazine*, 2016.
- [16] Z. Ali and J. Payton and V. Sritapan. At your fingertips: Considering finger distinctness in continuous touch-based authentication for mobile devices. In *2016 IEEE Security and Privacy Workshops (SPW)*, 2016.

Adversarial Attacks in Deep Reinforcement Learning

Miska Kankkonen

miska.kankkonen@aalto.fi

Tutor: Buse Gul Atli

Abstract

Machine learning algorithms are integrated with various applications, and this trend only seems to accelerate. Increasingly, the applications are also taught to make decisions on their own. As the use of smart algorithms in decision-making changes the industry, autonomous machines have started to control certain systems of cars in order to navigate and drive the car while providing safety. As a result, it is important to understand how these algorithms work and what are the possible pitfalls. This paper will summarize basic architectures behind these algorithms and an effective method used to deceive them. To achieve this, the paper explains concepts such as reinforcement learning (RL), Markov decision process (MDP) and adversarial attacks. Finally, modern reinforcement learning algorithms are considered from the adversarial attack point of view, and possible future work is discussed.

The paper is written in such a manner that no previous expertise of reinforcement learning is required. Therefore, it works as a comprehensive starting point for anyone who has found interest in self learning algorithms. The paper structure is kept compact and most of the topics are discussed in general level, excluding the most relevant topics which are explained in detail. This approach aims for a faster familiarization with basic concepts and shows a variety of different possibilities for future research.

KEYWORDS: deep reinforcement learning, adversarial attack, FGSM, MDP

1 Introduction

Machine learning has become a central topic in numerous applications, such as autonomous driving, image recognition and automated decision making [9, 13, 23]. Even though machine learning has been the focus of extensive research for decades, it was only after the invention of deep learning that the results reached unprecedented accuracies [9]. A deep learning algorithm tries to find patterns from an input data in order to produce accurate estimations for inputs that it has never seen before. To achieve this, a deep learning algorithm requires extensive amounts of input data in a separate automated training process. Before the invention of deep learning, machine learning algorithms mostly focused on the implementation process. However, with the deep learning, the focus is on the results. Instead of defining exhaustively how the algorithm should work, it is now defined what the algorithm should produce.

The deep learning is considered as the state-of-the-art in many applications. However, as the whole training process is automated, this can lead to a situation where there is a little knowledge on how the decision was eventually made. These algorithms can be seen as black boxes [20]. They produce the expected result, but the reasoning for the final interpretation remains unclear. Black box algorithms becomes especially problematic when they start to produce wrong estimations. For example, a specific procedure where precisely crafted input manipulates the application to give incorrect results [21]. As a result, the attacker becomes capable to manipulate the predictions. This process is referred as adversarial attack.

Adversarial attacks are currently a heavily studied research area. However, studies on this area often focuses on supervised learning methods. In particular, they are focusing on the methods where the input data and expected results are well defined before the training process. This leaves other research areas, such as reinforcement learning, untended. In reinforcement learning, the input data and expected results are not fully known beforehand. Instead, those are results of actions taken by an agent in a continuously changing environment. An example of a reinforcement learning scenario is autonomous driving where the agent would be the car, environment would be the the driving area and the actions as an example could be "turn left" or "keep driving" [23].

This research studies the consequences of adversarial attacks with the reinforcement learning by reviewing the recent literature and discuss the directions for a possible future work.

2 Deep Reinforcement Learning

Machine learning can be divided into three distinct categories: Supervised learning, unsupervised learning and reinforcement learning [9]. In supervised learning, the task is to automatically find mapping between input and output data by examining examples. Unsupervised learning on the other hand tries to find similarities in an unlabeled data and group them based on those common characteristics. Reinforcement learning has a very different approach. Reinforcement learning can be seen as a setting, where an agent tries to learn the most profitable action sequence to achieve its goal in an environment through trial and error. The tools, that the agent can use to examine the environment has advanced drastically since the invention of deep neural network [15]. Especially, convolutional neural networks (CNNs), which are also part of the supervised learning, have enabled a mapping between a high-dimensional state of an agent to actions. Previously, this have been way too complex for simpler agents to process. This combination of deep neural networks and reinforcement learning is called deep reinforcement learning (DRL), and it can be seen as a fusion of deep learning and reinforcement learning [15].

2.1 Deep and convolutional neural networks

Deep neural networks are supervised learning models. These models maps a high dimensional input x (image, text etc.) into an output y . The model architecture is composed using layers and the layers are stacked in such a way that the output of the previous layer is fed as an input to the next layer. The type of a single layer can vary but in general, each layer is constructed from neurons and activation functions. When considering the difference between deep neural networks and shallow neural networks, the word "deep" refers to the fact that there exists at least two layers between the input and output. These layers between input and output are commonly referred as hidden layers, and together they compose the deep neural network.

Convolutional neural network (CNN) is one example of the deep neural

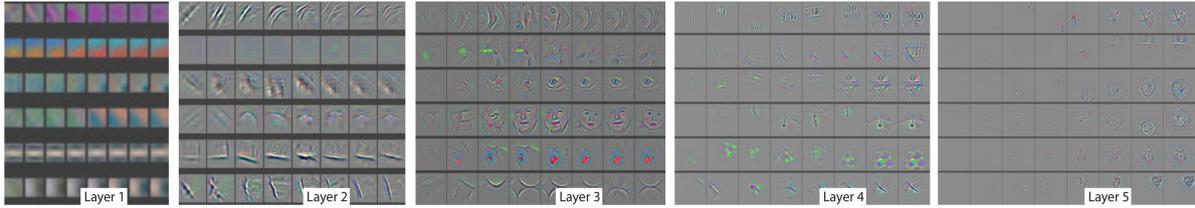


Figure 1. Different layers of abstraction with CNN image recognition domain. Layers 1 and 2 learns simple features such as edges, whereas Layer 3, 4 and 5 learns more complicated structures such as faces. Source: Adapted from [26].

networks. CNNs are constructed using different types of hidden layers such as convolutional layers, pooling layers and fully connected layers. The network is designed to represent images with a more rich format. The innovation behind the network roots from biology, and to be more precise from the visual cortex of mammals [3]. CNNs have provided excellent performance and are widely used in different applications, such as computer vision, speech recognition and text classification [7, 8, 1], due to their ability to extract high-level features from high-dimensional input data. When learning patterns from an image, the network does not pass on information about single pixels. Instead, it passes information about the patterns it has been able recognize. In addition, when pattern recognition is combined with the concept of layer stacking, the network becomes even more powerful. First layers of the CNN will recognize simple patterns such as edges and lines. But on the higher layers the patterns start to be objects such as faces, cars and buildings [26]. Figure 1 illustrates this behavior. The higher level information can then be used instead of the raw pixels. CNNs are a powerful tool for the agent to understand and recognize the environment around it in a reinforcement learning setup, which will be explained later in detail.

2.2 Reinforcement learning

Reinforcement learning (RL) is a setup, where the agent takes actions in an environment and receives rewards based on those actions, as illustrated in Figure 2. The problem that the RL tries to solve can be described using the Markov Decision Process (MDP). For any problem to be effectively solvable using the RL methods, they must also comply with the Markov property. The property defines that the probability of future states of the process is only affected by the current state, the process is not required to memorize preceded states for that.

MDP constitutes from a set of states S , a set of actions A , states $s_i \in$

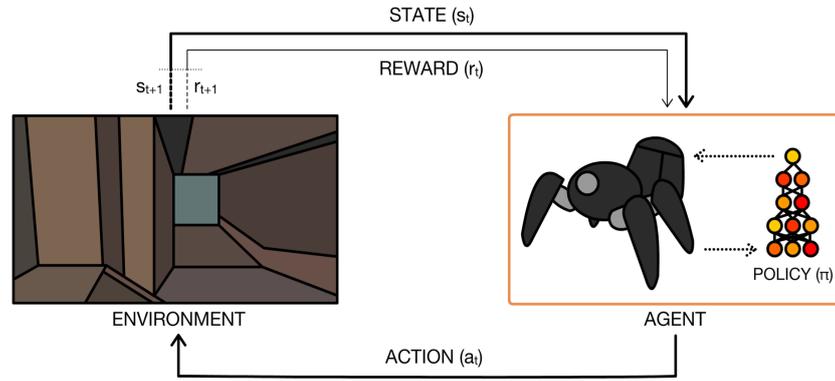


Figure 2. An agent-environment loop in reinforcement learning, where agents takes actions and receives both new state and reward from the environment. Source: Adapted from [2].

S , actions $a_j \in A$, transition dynamics $\tau(s_{t+1}|s_t, a_t)$, a reward function $\mathcal{R}(s_t, a_t, s_{t+1})$, rewards r_k and a discount factor $\gamma \in [0, 1]$ [2]. In addition to these, distribution of starting states $p(s_0)$ and a policy $\pi : S \rightarrow p(A = a|S)$ are required. In the RL setup, the agent is the main actor that is executing the actions a_t and for any given time it is on some state s_t . The state s_t is an entity that represents a specific and unique composition of the environment. The agent is able to take actions a_t that moves it from the previous state s_t into the next state s_{t+1} . However, taking the action a_t during the state s_t does not always produce the same next state. The list of possible next states s_{t+1} and their corresponding probabilities are determined by the transition dynamics $\tau(s_{t+1}|s_t, a_t)$. The agent chooses the appropriate action using the policy π . This policy π maps all the possible states S into a probability distribution p when some action a_x is taken. At the beginning, probability distribution of starting states $p(s_0)$ is used to define the initial setting. This setting is simply a discrete function, which maps all the initial states into corresponding probabilities.

To formalize the goal of the agent and to be able to give feedback for different states, accumulated reward function $R = \sum_{t=0}^{\infty} \gamma^t r_{t+1}$ is calculated. This reward is the sum of all gathered partial rewards r with discount rate γ . The discount rate that decays with time t has at least two different purposes. First, it is used to control the amount of steps that the policy considers during the decision making. If the discount rate is closer to zero, agent's decisions are heavily influenced by immediate rewards. On the other hand, if the discount rate is closer to 1, agent considers future rewards while taking action. If the discount factor is equal or bigger than 1, the accumulated sum and the overall algorithm will not converge.

Generally, the discount factor is set near 0.9.

Now that the accumulated reward function is defined, the ultimate goal of reinforcement learning can be put in a more formal way [2]. Find the optimal policy that maximizes the expected return from any starting state.

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \mathbb{E}[R|\pi] \quad (1)$$

Reinforcement learning is a very rapidly developing research area, and new approaches are constantly emerging. There are situations where the formal definition presented above is too limited. One example could be a scenario, where the agent might not be able to fully observe the underlying state [2]. However, MDP still works as a very good basis and provides an in-depth understanding of the problem setting.

3 Adversarial Attacks

Adversarial attack is a technique used to deceive machine learning algorithms (particularly deep learning algorithms such as convolutional neural networks). When the attack is properly applied, previously well functioning algorithms starts to produce wrong estimations with high confidence [22]. As the estimations might be utilized by critical applications, such as autonomous driving, the results of incorrect interpretations are destructive. Understanding the assembly process and the inner structures of adversarial attacks are crucial to secure the integrity of machine learning algorithms against these type of attacks.

3.1 Selecting a target, an approach and a goal

When constructing an adversarial attack, it is important to consider the target model. The target models can be classified into two groups: white-box models, where the attacker have full access to the algorithm, and black-box models, where the attacker does not know inner mechanisms of the algorithm or which algorithm is in use [25]. In case of the white-box model, deciding how the algorithm should be deceived is more straightforward, as the attacker has full access to the inner functionalities. Such an approach should be selected, that was especially designed to leverage weaknesses of the underlying deep learning algorithm. On the other hand, when attacking a black-box model, there are less information avail-

able and crafting the adversarial example becomes more expensive. Studies show that a single adversarial attack can be effective on variety of different deep learning models. Furthermore, this is not only limited to different iterations of the same model. Instead, adversarial examples might be transferable between different models when they are trained on a similar dataset distribution [16]. This correlation can arguably be seen as the basis for the black-box attacks. Naturally, the correlation between different algorithms varies and for time being no single answer exists for selecting the best approach for all possible cases.

After the target model has been chosen, the attacker needs to decide the goal. The goal can be divided into four categories: confidence reduction, misclassification, targeted misclassification and source/target misclassification [17]. With the first two options, the attacker simply wants to disrupt the basic behavior. It does not matter what the algorithm predicts as long as the algorithm produces wrong estimations or have lower confidence with the correct ones. With the targeted misclassification, the algorithm is deceived to predict the same target class, which is different than the original class, for all the test data. This approach has similarities with the last option: source/target misclassification. However, the source/target misclassification forces only one input to be classified as the target label. The difference with these last two is that the produced perturbation of the latter one is not expected to work with other images having the same original label.

Once the target, the approach and the goal are clear it is time to craft the actual attack.

3.2 Crafting the attack

Adversarial attack can be constructed using variety of different algorithms such as Fast Gradient Sign Method (FGSM) [4] and Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) [22]. This paper will focus on FGSM, as it seems to be a popular choice and have excellent capabilities such as cheap computational cost [6].

FGSM leverages the weaknesses of deep neural networks (DNNs) by directly exploiting the cost function $J(\theta, x, y)$. Here θ denotes the trainable network parameters, x is the input data and y is the expected label. The cost function calculates the difference between the predicted outcome of the model and the expected outcome (true class). There are different functions to calculate the cost, but a common choice for DNN is a cross-entropy

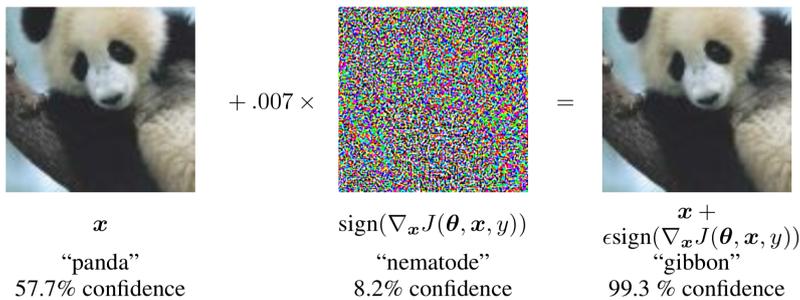


Figure 3. An original image (left), perturbation calculated with FGSM (center) and manipulated image (right). A small perturbation can change the predicted label from true class "panda" to "gibbon". For human observer, two images are still indistinguishable. Source: Adapted from [4].

cost. It represents the difference between predicted distribution of class probabilities and the actual class. Cross entropy cost is convex; therefore, it is differentiable and can be minimized.

Combining the gradient ∇ of the function $J(\theta, x, y)$ w.r.t to the input x , we get a function $\nabla_x J(\theta, x, y)$, that represents the relation between the input and the cost function. Using this method, FGSM can effectively produce noise η that causes high increase in the cost function $J(\theta, x, y)$ when added to the original input x . To have full control over the amount of perturbation, ϵ symbol is introduced. In addition, *sign* method is utilized to find the direction of perturbation. Instead of directly using the values of $\nabla_x J(\theta, x, y)$ we only consider the sign of the function $sign(\nabla_x J(\theta, x, y))$. This is then multiplied with ϵ thus producing the following equation [4].

$$\eta = \epsilon sign(\nabla_x J(\theta, x, y)) \quad (2)$$

This equation is referred as fast gradient sign method [4]. If η were unwrapped, the content might be something like $[[\epsilon, -\epsilon, -\epsilon, \epsilon, \dots], \dots]$. By changing the magnitude of ϵ , the amount of perturbation can be controlled. Selecting the most sufficient value for ϵ depends on the sensitivity of a human observer. However, when undetectability is a factor, an adequate solution could be based on the discrete step between two adjacent values supported by the computer memory [4]. If the input were a 8 bit pixel then the step size would be $1/255 \approx 0.0039$.. and thus ϵ would be something slightly more than that.

The attacker is now able to introduce carefully crafted noise into the network and manipulate the results. Figure 3 shows FGSM in practice.

4 Results and Discussion

Recently, adversarial attacks in reinforcement learning has emerged as a potential research topic, since these attacks can be a serious problem in reinforcement learning applications. Indeed, the studies strongly reports that the generated perturbation, that were effective against the supervised learning models, are also disruptive on the deep reinforcement learning setup [6, 11]. This section will overview recent studies and list the current status of adversarial attacks on different reinforcement learning algorithms. In conclusion, different possibilities for future work are discussed.

4.1 A brief overview of the reinforcement learning algorithms

To solve the problem setting of reinforcement learning, numerous different approaches have been suggested. Many algorithms exists, such as Q-learning [24] and SARSA [18], that does not depend on neural networks. The strength of these algorithms is that given unlimited amount of resources and time they will eventually find the optimal solution. However, in practice the number of possible states are usually too high and too complex to represent. This results in an increased demand for huge memory resources and leads to high time complexity. To solve these limitations, variety of different deep reinforcement learning algorithms have been proposed. Deep Q-learning (DQN) replaces the memory intensive parts of the Q-learning with deep neural networks [15]. This allows a more efficient solution, as the Q-learning agent does not need to store all the possible combinations of the state-action pairs in the memory anymore. Instead, deep neural networks are used to approximate the quality of state-action pairs.

Deep Deterministic Policy Gradient (DDPG) [10] takes the utilization of deep neural networks even further. To achieve better accuracies, DDPG is split into two parts: an actor and a critic. Both of these are deep neural networks with slightly different goals. The actor is still responsible for taking the actions based on the given policy, and the action is selected by the underlying deep neural network. The critic on the other hand evaluates the actor. Given the current state and the received reward, the critic produces a temporal-difference error signal, which is then used to evaluate and improve the estimated policy function. The loss between the action value function computed by the actor and the next state values cal-

culated with the critic is minimized over time for convergence. Previously, the DQN required the action space to be discrete. This led to a situation where the action space became enormous and determining the most relevant action impractical. However, DDPG replaces the determination process with another deep neural network. This network is responsible to approximate the most sufficient action given the current state.

Asynchronous Advantage Actor-Critic (A3C) is another implementation of deep reinforcement learning that utilizes the actor-critic architecture [14]. Similar to DDPG, A3C also relies heavily on deep neural networks. In addition to convolutional layers and fully connected layer, A3C also includes another type of layer called Long Short-Term Memory (LSTM) [5]. LSTM was designed to give the neural networks a memory. This memory allowed the network to learn patterns that could only be recognized by examining the data as a sequence. A3C architecture has also another advantage as it allows asynchronous processing. Asynchronous processing enables multiple agents, called workers, to learn concurrently and update the results into a single shared model. Concurrency of the A3C allows parallel processing and reduces the training time.

Finally, this paper will also give an example of adversarial attacks with Trust Region Policy Optimization (TRPO) [19]. TRPO introduces a new approach to improve the gradients of the policy. This approach limits the amount of a single gradient update by creating a trusted region. The trusted region is then able to protect the gradient updates from overshooting. The next section discusses the current state of these algorithms and their connection with the adversarial attack.

4.2 Adversarial attacks in deep reinforcement learning

The adversarial attacks are designed to deceive any type of machine learning algorithm. The recent studies have investigated if the adversarial attacks in DNN can be applied to deep reinforcement learning field. Therefore, the community did not explore adversarial attacks against Q-learning and SARSA as they do not use neural networks to map states to actions. However, majority of recent algorithms, such as DQN and A3C, relies heavily on deep neural networks. DQN is often selected as the benchmark algorithm because it is the oldest of the deep reinforcement learning algorithms. A3C is the successor of DQN, and the A3C surpasses the DQN in both the training speed and accuracy [14]. The performance of these algorithms can decrease if the state is perturbed with an adversarial noise

Adversarial attacks on reinforcement learning				
The algorithm	Year	Constitutes from neural networks	Vulnerable for adversarial attacks	Defenses against the attacks
Q-Learning [24]	1989	-	?	-
SARSA [18]	1994	-	?	-
TRPO [19]	2015	+	+	-
DQN [15]	2015	+	+	/ *
DDPG [10]	2015	+	+	/ *
A3C [14]	2016	+	+	-

Table 1. Comparison of popular reinforcement learning algorithms. Symbol '+' refers as true, '-' refers as false, '?' refers as unknown and '/' refers as partially true. Adversarial attacks are not tested against Q-Learning and SARSA. *) Some approaches exist, but they still lack the generality [12].

[6]. The reduction in performance is not limited to these two algorithms. Instead, variety of other deep learning algorithms are also vulnerable to adversarial attacks including DDPG and TRPO [12, 6]. As all of these algorithms includes deep neural networks, the negative effect of adversarial inputs are expected.

Majority of the studies only confirms the effectiveness of adversarial attacks on DRL algorithms. On the other hand, only a few suggests defenses against these attacks. Different ways have been proposed to protect and train the network so that it would be more resistant. These include autoencoders, dropout layers and feature squeezing [12]. Even though some of the studies have achieved prominent results, they still lack the generality and have limitations such as lack of defense against an adaptive adversary [12]. Table 1 compares the different status of reinforcement learning algorithms when considering adversarial attacks.

5 Conclusion and Future Work

This paper summarizes different popular approaches utilized in deep reinforcement learning and studies the security concerns with these approaches. Variety of different reinforcement learning algorithms are reviewed such as deep Q-learning and A3C. In addition, a specific type of vulnerability exploiting technique called adversarial attack is discussed. Many of the studies show that deep reinforcement learning algorithms

are vulnerable for adversarial attacks. Summary of these results can be seen on table 1.

Many questions remain still open. One of the most important one is "How can you defend against these attacks?". No single answer exists yet. Still, it is important to keep asking this question as all these deep learning technologies are being rapidly implemented in our everyday life. Furthermore, this paper focused on adversarial attacks. However, this is not the only threat against deep neural networks. One example for future work could be studying the effects of less known threats such as data poisoning. Lastly, can we introduce adversarial attacks into multi-agent setting where agents either collaborates with each other or competes against each other? Can a single agent be deceived in this setting and how this affect other agents? These questions and gaps will motivate future research in this field as well as emphasize the importance of securing artificial intelligence.

References

- [1] U Rajendra Acharya, Shu Lih Oh, Yuki Hagiwara, Jen Hong Tan, and Hojjat Adeli. Deep convolutional neural network for the automated detection and diagnosis of seizure using eeg signals. doi: 10.1016/j.combiomed.2017.09.017. *Computers in biology and medicine*, 100:270–278, 2018.
- [2] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. A brief survey of deep reinforcement learning. *arXiv preprint arXiv:1708.05866*, 2017.
- [3] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. doi: 10.1007/bf00344251. *Biological cybernetics*, 36(4):193–202, 1980.
- [4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. doi: 10.1162/neco.1997.9.8.1735. *Neural computation*, 9(8):1735–1780, 1997.
- [6] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.
- [7] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. doi: 10.3115/v1/p14-1062. *arXiv preprint arXiv:1404.2188*, 2014.
- [8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. doi: 10.1145/3065386. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [9] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. doi: 10.1038/nature14539. *nature*, 521(7553):436, 2015.
- [10] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [11] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*, 2017.
- [12] Yen-Chen Lin, Ming-Yu Liu, Min Sun, and Jia-Bin Huang. Detecting adversarial attacks on neural network policies with visual foresight. *arXiv preprint arXiv:1710.00814*, 2017.
- [13] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghahfoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. doi: 10.1016/j.media.2017.07.005. *Medical image analysis*, 42:60–88, 2017.

- [14] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.
- [15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. doi: 10.1038/nature14236. *Nature*, 518(7540):529, 2015.
- [16] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.
- [17] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. doi: 10.1109/eurosp.2016.36. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387. IEEE, 2016.
- [18] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, England, 1994.
- [19] John Schulman, Sergey Levine, Pieter Abbeel, Michael I Jordan, and Philipp Moritz. Trust region policy optimization. In *Icml*, volume 37, pages 1889–1897, 2015.
- [20] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.
- [21] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. doi: 10.1109/tevc.2019.2890858. *IEEE Transactions on Evolutionary Computation*, 2019.
- [22] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- [23] Sen Wang, Daoyuan Jia, and Kinshuo Weng. Deep Reinforcement Learning for Autonomous Driving. *arXiv e-prints*, page arXiv:1811.11329, November 2018.
- [24] Christopher John Cornish Hellaby Watkins. *Learning from delayed rewards*. PhD thesis, King’s College, Cambridge, 1989.
- [25] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. doi: 10.1109/tnnls.2018.2886017. *IEEE transactions on neural networks and learning systems*, 2019.
- [26] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. doi: 10.1007/978-3-319-10590-1_53. In *European conference on computer vision*, pages 818–833. Springer, 2014.

Data exchange between smartphones using visible light

Ekaterina Shmeleva

ekaterina.shmeleva@aalto.fi

Tutor: Maria Montoya Freire

Abstract

LTE, Wi-Fi, Bluetooth and NFC are the most widespread wireless communication technologies. However, these technologies use radio frequencies, which causes a number of limitations. In that light, visible light communication (VLC) is considered to be a promising alternative to short-range radio-based wireless technologies. However, the great diversity of mobile devices and operational conditions, as well as imperfect frame synchronization, pose challenges that limit the throughput of the screen-camera link. These challenges require novel and flexible solutions. This paper reviews and compares several barcode-based VLC systems that use a number of different methods of improving the throughput of the screen-camera link. It concludes that methods that shift the bottleneck to the receiver have the most potential. One of these methods uses layered visual codes, while another boosts the effective screen frame rate and transmits the data in a looping cycle so that any receiver can eventually decode the data.

KEYWORDS: *visible light communication, short-range wireless communication, barcodes*

1 Introduction

The widespread use of smartphones and tablets has revolutionized the way people communicate with each other and has given them the ability to be constantly connected through their mobile devices. This became possible with the advent of wireless technologies such as Long-Term Evolution (LTE), Wi-Fi, Bluetooth, and Near-Field Communication (NFC).

Despite having a significant role in communication, all these technologies use radio waves, which causes certain limitations. First, in cellular and Wi-Fi networks, signal strength may be affected by a number of factors, including the proximity to the cellular base station or access point, weather, terrain, building layout and materials, and even pedestrian traffic [4]. Second, wireless networks relying on radio frequencies are sensitive to electromagnetic interference (EMI), which may occur due to both natural and anthropogenic causes and result in data corruption or loss [4]. A third limitation is that the radio frequency (RF) spectrum is a limited resource. As a result, a substantial demand for the spectrum gradually leads to its congestion. For this reason, the use of RFs is rigidly regulated [1, 14]. Finally, from a security standpoint, radio-based wireless networks are vulnerable to eavesdropping and jamming attacks [19].

Visible light communication (VLC) could be a promising alternative to traditional short-range RF technologies for wireless communication, such as NFC and Bluetooth. VLC operates in the unregulated visible light portion of the electromagnetic spectrum and, compared to RF technologies, exhibits robustness to EMI, brings unlimited spatial reuse [14], has no dependency on cellular or Wi-Fi infrastructure, and creates fewer privacy and security risks [3, 13, 17, 2]. In certain use cases, VLC has the potential to become more efficient than RF communication in terms of performance and power consumption. The most promising use cases of mobile-to-mobile VLC include device pairing and data exchange [7].

Due to the prevalence of off-the-shelf smartphones and tablets equipped with built-in cameras, visible light communication between mobile devices in most cases requires no additional hardware. Instead, the screen and camera act as a sender and receiver, respectively, and the data transmitted by the sender is encoded in a barcode [9, 3, 5, 6, 2, 17, 13, 16, 18]. However, the throughput of the screen-camera link is significantly affected by external and internal factors. External factors include lighting conditions, perspective distortion, defocus aberration, motion blur [11],

transmission range, sender and receiver hardware [6]. Internal factors are determined by data encoding and decoding techniques. For barcodes, these factors include capacity, error correction level, and inter-symbol interference [10].

The goal of this paper is to provide an overview of the state-of-the-art of VLC systems, summarise and compare methods for increasing the throughput of the screen-camera link. The rest of this paper is organized as follows. Section 2 introduces the background. Section 3 reviews and compares several methods for increasing the throughput. Finally, Section 4 concludes this paper.

2 Background

This section introduces several relevant visual code designs and explains how they can be used to transmit large payloads. A literature review revealed that most recent studies explore the use of a stream of barcodes for VLC over the screen-camera link. By definition, a barcode is a machine-readable optical representation of data. Based on a number of dimensions, barcodes can be divided into one-dimensional, two-dimensional, and multi-dimensional barcodes. This section gives a description and discusses the limitations of each of the three categories.

2.1 One-dimensional barcodes

A one-dimensional (1D) barcode is a pattern composed of a series of parallel bars and spaces of varying width. In general, a 1D barcode is made up of a quiet zone, start and stop characters, data characters, and a mandatory or optional checksum character to ensure integrity [8]. Figure 1 shows some examples of 1D barcodes.



Figure 1. 1D barcodes

Although 1D barcodes are extensively used in retail, wholesale, grocery, and other industries, their very limited data capacity makes them unsuitable for data exchange between mobile devices.

2.2 Two-dimensional barcodes

In order to increase the data capacity of a barcode, a number of two-dimensional (2D) barcode designs have been proposed that are now available for general use. These designs can be divided into two groups: stacked barcodes and matrix barcodes.

2.2.1 Stacked barcodes

Stacked barcodes consist of an arbitrary number of 1D barcodes stacked on top of each other. Their main limitation is that the receiver must be carefully aligned with the barcode [12]. Figure 2 shows an example of a PDF417 barcode. Other examples of stacked barcodes include Code 49, Code 16K and CodaBlock F.



Figure 2. PDF417

2.2.2 Matrix barcodes

Unlike stacked barcodes, matrix barcodes can be scanned from a variety of angles. In general, matrix barcodes feature more sophisticated methods for enhancing robustness and can store a greater amount of data in a single barcode [12]. Due to these properties, matrix barcodes are extensively used by off-the-shelf mobile devices.

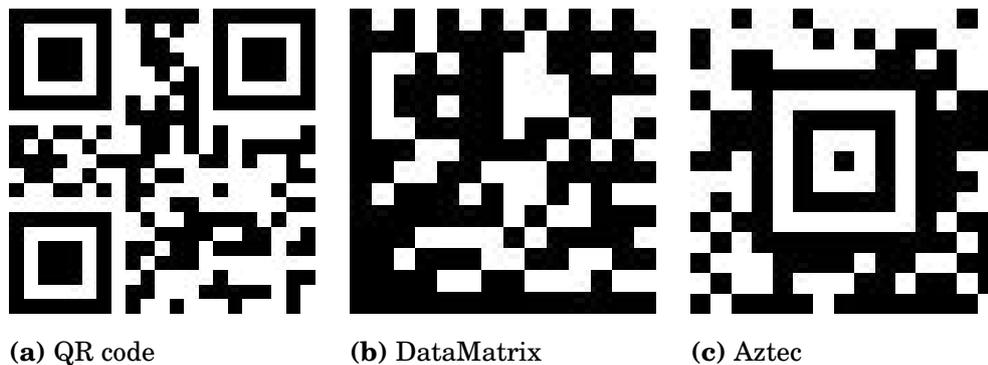


Figure 3. Matrix 2D barcodes

Nevertheless, existing matrix barcode designs (Figure 3) offer limited flexibility and exhibit all-or-nothing behaviour: even with forward error correction, the success of reading depends greatly on the image quality. Furthermore, matrix barcodes only support a small range of distances at which the scanned barcode is sufficiently in focus to be read without error. Hence, research is being conducted to address these limitations.

For example, Hu et al. [6] propose a scalable barcode design, which is further discussed in Section 3.

2.3 Multi-dimensional barcodes

2.3.1 Time dimension

Even though 2D barcodes can encode significantly more information than 1D barcodes, their data capacity is still limited. As a result, 2D barcodes are not seen as an alternative to traditional RF communication technologies, but rather as a complement to them [5]. Nonetheless, a stream of barcodes can be used instead of a single static barcode to achieve virtually unlimited data capacity.

However, the use of a stream of barcodes presents several challenges. First, mobile device cameras exhibit notable differences in the frame capture rate. This may cause a mismatch between the sender frame display rate and receiver frame capture rate and create a synchronization problem [6]. Second, a typical mobile device camera employs a rolling shutter. It scans a barcode line by line, which leads to mixed frames [5]. Section 3 further discusses some approaches to overcoming these limitations.

2.3.2 Colour dimension

Unlike a traditional laser scanner, a smartphone camera is able to detect colours. Therefore, it is possible to use colours other than black and white. In order to improve the throughput of the screen-camera link, COBRA [3], RDCode [15], TETRIS [13], RainBar+ [18] and other works suggest the use of a stream of colour barcodes with increased encoding capacity. A stream of colour barcodes forms a four-dimensional (4D) barcode with width, height, colour, and time dimensions.

3 Methods for increasing the throughput

The first part of this section presents an overview of six barcode-based VLC systems that use a number of different methods for increasing the throughput. It is organized in chronological order to show the evolution of these methods.

The second part presents a summary and comparison of these methods based on the published findings. It also briefly mentions other works that were not included in the overview.

3.1 Overview

3.1.1 *Unsynchronized 4D Barcodes*

In 2007, Langlotz et al. [9] proposed a 4D barcode design for wireless data transmission between the public LCD or plasma display and off-the-shelf mobile phone camera. For perspective, it was the same year that Apple released the first-generation iPhone. Accordingly, the proposed design targets mobile phone cameras with a lower frame rate and smaller resolution compared to those of modern smartphones.

In particular, the paper considers a scenario where other communication technologies cannot be used and, as a result, no immediate synchronization between the sender and receiver is possible. To solve the synchronization problem, the system embeds three redundant 2D barcodes in each frame into red, green and blue colour channels. However, this increases the robustness of the system but not the throughput. To achieve higher throughput, the data is optionally compressed before encoding.

The system shows a throughput of 0.138 kbps with a success rate of 82% and 95% for novel and experienced users accordingly. Due to the computation overhead of corner detection and image rectification [3], the achieved data transmission rate is significantly lower than that of Wi-Fi or Bluetooth. It is not considered to be acceptable for real-time use even in a given application scenario, where there are no alternative communication technologies.

3.1.2 *COBRA*

COlour Barcode stReaming for smArtphones (COBRA) is a barcode-based VLC system proposed by Hao et al. [3]. It encodes the data in a custom 2D colour barcode, which is optimized for off-the-shelf smartphones with relatively small screen size and low-speed camera.

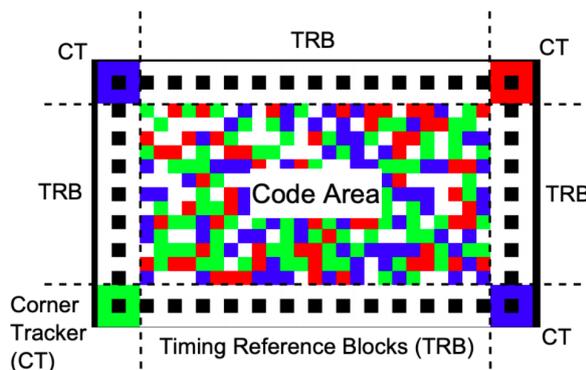


Figure 4. A 2D colour barcode for COBRA [3]

The barcode consists of square colour blocks of equal size. It features corner trackers and timing reference blocks (Figure 4), that significantly improve decoding performance. The four corner trackers are used for quick barcode boundary detection. The timing reference blocks minimize the impact of perspective distortion and are used to locate the actual information carrying colour blocks. The orientation of a barcode is determined by the position of the red and green corner trackers. COBRA also minimizes the length of the edge that blocks of different colour share. This reduces image blurring, which normally occurs across such edges.

COBRA shows the maximum throughput of 225 kbps when using a 6-pixel block size and a refreshing rate of 10 fps. However, the throughput is massively affected by block size and distance between the sender and receiver.

3.1.3 *LightSync*

Hu et al. identify imperfect frame synchronization as a major challenge for smartphone-to-smartphone VLC. In order to achieve frame synchronization, unsynchronized 4D barcodes and COBRA simply decrease the effective screen frame rate. This approach is inefficient because the transmission capacity of the sender is underutilized. To address this problem, Hu et al. suggest LightSync [5] as an alternative to the previous solutions. It achieves faster and more flexible frame synchronization, thus improving the throughput of the screen-camera link. The average throughput of LightSync is up to 2.3 times higher than that of COBRA.

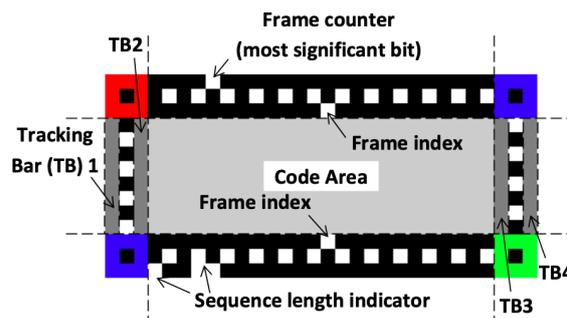


Figure 5. A colour 2D barcode with tracking bars [5]

LightSync offers two mechanisms to ensure that the receiver can decode mixed frames and restore any lost frames. The first one is the in-frame line-based overlap tracking, which is implemented by inserting tracking bars in every frame in the scanning direction (Figure 5). These bars allow to determine the extent of overlap and decode imperfect frames. The second one is the inter-frame erasure coding: each frame encodes data

from multiple original frames for recovery of lost data. The sender plays the entire barcode stream in a looping cycle. As a result, the receiver can eventually decode the data even if its frame capture rate is lower than the screen frame rate.

3.1.4 *Strata*

The quality of the screen-camera channel is generally affected by the distance between the sender and receiver, camera resolution and capture frame rate. Pixel colours may blend together spatially when captured by a low-resolution camera or from a long distance, and temporally when captured at a low frame rate. This issue is commonly addressed by either reducing the encoding capacity of a single barcode and effective screen frame rate for increased robustness under varying conditions, or imposing stringent requirements on the screen-camera link. However, both approaches are not scalable. To address this problem and achieve higher throughput under favourable conditions, Hu et al. propose Strata [6]. Strata is a scalable layered coding scheme for VLC, which encodes the data at different levels of spatial and temporal detail. As a result, the receiver decodes as much data as the screen-camera link quality allows.

Strata encodes the data in a black and white barcode, which tolerates more noise than a colour barcode. A 2D barcode is created by recursively increasing the resolution of blocks at successive spatial layers. It can be used equally as a static image or in a stream. A barcode stream is created by showing the base temporal layers at the lowest frame rate and increasing the frame rate at successive temporal layers.

3.1.5 *TETRIS*

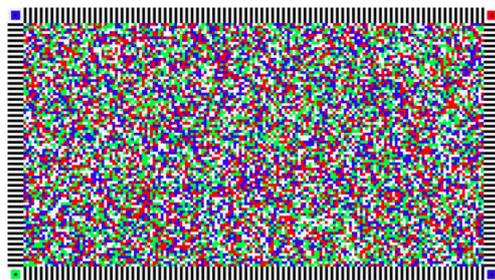


Figure 6. A 2D colour barcode for TETRIS [13]

TEtra- TRansmISSion (TETRIS) is a barcode-based VLC system proposed by Stafford et al. [13]. Much like COBRA, LightSync and RainBar [16], TETRIS uses a 2D colour barcode that features four corner trackers and multiple timing reference strips. However, TETRIS takes advantage

of recent development in the smartphone screen and camera quality by decreasing the size of a block and increasing the frame rate and to achieve higher throughput (Figure 6). In addition, TETRIS adds a margin around each corner tracker to improve the accuracy of corner detection. Nevertheless, the system requires that the user manually marks the screen corners.

TETRIS reportedly shows a throughput of about 311 kbps with 90% accuracy. The throughput is 1.7 times higher than that of COBRA but lower than that of RainBar and SoftLight in the same test. An increase in the number of blocks predictably leads to an increase in the amount of time required to process a single frame.

3.1.6 *RainBar+*

RainBar+ is a barcode-based VLC system with real-time feedback proposed by Zhou et al. [18]. In contrast to other works that focus solely on the use of the screen-camera link, RainBar+ employs the speaker-microphone link, which serves several purposes. First, the use of the speaker-microphone link solves the frame synchronization problem and enables the adaptive barcode configuration. This improves the throughput under favourable conditions when the transmission capacity of the sender can be fully utilized. Second, it allows the receiver to notify the sender if a frame is corrupted or lost. This frame will be then retransmitted to produce high goodput.

However, the speaker-microphone link quality is affected by a number of internal and external factors, most notably by ambient noise. In order to mitigate these factors, Zhou et al. implement a feedback communication system using orthogonal frequency-division multiplexing (OFDM), which is a method of encoding digital data is commonly used in audio broadcasting.

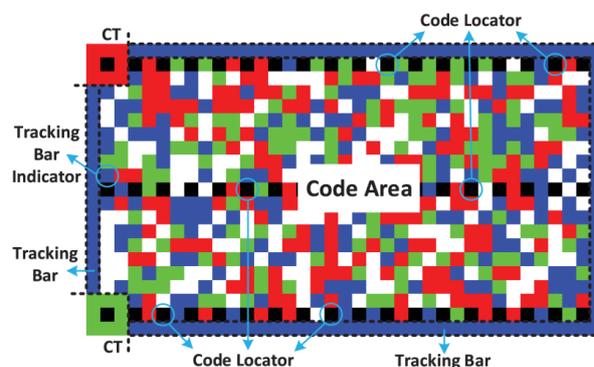


Figure 7. A 2D colour barcode for RainBar+ [18]

RainBar+ increases the encoding capacity of a single barcode in order to produce even higher throughput (Figure 7). The barcode design is based on previous work [16]. As a result of these improvements, RainBar+ exhibits a higher decoding rate and throughput than COBRA.

3.2 Comparison

Unsynchronized 4D barcodes and COBRA laid the foundation for future research on smartphone-based VLC. Many VLC system designs are based on COBRA. TETRIS and RainBar(+) are some examples of such systems. LightSync works independently of any specific barcode design. Nonetheless, the authors also use a slightly modified version of COBRA, since it is optimized for fast code block extraction. However, Strata and RDCode propose novel barcode designs.

Overall, the described VLC systems use one of the following three methods for increasing the throughput of the screen-camera link.

3.2.1 *Transmission frame rate boost*

LightSync improves throughput by pushing the effective screen frame rate. It supports a much higher transmission frame rate than conventional systems. It can also achieve a higher mean rate than Strata, if the receiver supports high frame capture rate [6]. However, LightSync does not scale spatially.

3.2.2 *Barcode encoding capacity boost*

TETRIS attempts to improve the throughput by pushing the encoding capacity of a single barcode. In contrast to COBRA and other reviewed systems, it does not handle blurring, distortions or ambient light. As a result, it is impossible to draw conclusions about how TETRIS would behave in a real-life situation. What is more, this design does not scale. Even though TETRIS exhibits a throughput higher than that of COBRA under normal operational conditions, it will not achieve the same result under challenging operational conditions.

3.2.3 *Scalable streaming*

Strata and RainBar+ are scalable designs that push the frame rate and encoding capacity of a single barcode for high-end smartphones and short capture distances while still supporting low-end devices and long capture distances. However, they approach the frame synchronization problem from different angles. In Strata, the throughput scales independently

from the sender. By contrast, RainBar+ requires that the sender adjusts the throughput based on feedback sent by the receiver. An important disadvantage of Strata is that the encoding capacity of a multi-layer barcode is lower than that of a single-layer barcode, while RainBar+ can fully utilize the area of a barcode. However, the use of the speaker-microphone link creates more potential points of failure. Furthermore, this approach may result in poor user experience. It also puts more limits on operational conditions under which RainBar+ can be used.

4 Conclusion

This paper highlights current challenges that limit the throughput of the screen-camera link, and overviews and compares methods of solving them. Firstly, it gives a brief description of popular barcodes and how they can be used to transmit large payloads. Then, the paper explains how six different barcode-based VLC systems approach the problem of throughput improvement. Finally, it presents a comparative analysis of several methods, the differences between static and scalable methods and between different approaches to visual code scalability.

The comparative analysis shows that multi-layer visual code designs, such as Strata, have the most potential. This is because they shift the throughput bottleneck to the receiver without limiting the effective transmission frame rate. With this approach, the throughput of the screen-camera link between two high-end devices can be boosted. At the same time, other receivers are still supported. LightSync achieves similar results and also has potential. However, it is different in a way that it allows two receivers with a higher and lower frame capture rate to eventually decode the same data from a retransmission loop, while Strata encodes the data in different resolutions. As a result, both methods have great potential but should be used for different applications.

References

- [1] R. Boubezari, H. Le Minh, Z. Ghassemlooy, and A. Bouridane. Smartphone Camera Based Visible Light Communication. *Journal of Lightwave Technology*, 34(17):4121–4127, Sep. 2016.
- [2] W. Du, J. C. Liando, and M. Li. SoftLight: Adaptive visible light communication over screen-camera links. pages 1–9, April 2016.

- [3] T. Hao, R. Zhou, and G. Xing. COBRA: color barcode streaming for smartphone systems. *Proc. Mobisys*, 06 2012.
- [4] M. Harwood. *CompTIA Network+ N10-004 Exam Cram*. Que Publishing Company, 3rd edition, 2009.
- [5] W. Hu, H. Gu, and Q. Pu. LightSync: Unsynchronized Visual Communication over Screen-camera Links. pages 15–26, 2013.
- [6] W. Hu, J. Mao, Z. Huang, Y. Xue, J. She, K. Bian, and G. Shen. Strata: Layered Coding for Scalable Visual Communication. pages 79–90, 2014.
- [7] A. Jovicic, J. Li, and T. Richardson. Visible light communication: opportunities, challenges and the path to market. *IEEE Communications Magazine*, 51(12):26–32, December 2013.
- [8] H. Kato, K.T. Tan, and D. Chai. *Barcodes for Mobile Devices*. Cambridge University Press, 2010.
- [9] T. Langlotz and O. Bimber. Unsynchronized 4D Barcodes. pages 363–374, 2007.
- [10] A. Motahari and M. Adjouadi. Barcode Modulation Method for Data Transmission in Mobile Devices. *IEEE Transactions on Multimedia*, 17(1):118–127, Jan 2015.
- [11] S.D. Perli, N. Ahmed, and D. Katabi. PixNet: Interference-free Wireless Links Using LCD-camera Pairs. pages 137–148, 2010.
- [12] M. Querini, A. Grillo, A. Lentini, and G. Italiano. 2D Color Barcodes for Mobile Phones. *IJCSA*, 8:136–155, 01 2011.
- [13] M. Stafford, A. Rogers, S. Wu, C. Carver, N. S. Artan, and Z. Dong. TETRIS: Smartphone-to-Smartphone Screen-Based Visible Light Communication. pages 570–574, Oct 2017.
- [14] M. Uysal and H. Nouri. Optical wireless communications — An emerging technology. pages 1–7, July 2014.
- [15] A. Wang, S. Ma, C. Hu, J. Huai, C. Peng, and G. Shen. Enhancing Reliability to Boost the Throughput over Screen-camera Links. pages 41–52, 2014.
- [16] Q. Wang, M. Zhou, K. Ren, T. Lei, J. Li, and Z. Wang. Rain Bar: Robust Application-Driven Visual Communication Using Color Barcodes. pages 537–546, June 2015.
- [17] B. Zhang, K. Ren, G. Xing, X. Fu, and C. Wang. SBVLC: Secure Barcode-Based Visible Light Communication for Smartphones. *IEEE Transactions on Mobile Computing*, 15(2):432–446, Feb 2016.
- [18] M. Zhou, Q. Wang, T. Lei, Z. Wang, and K. Ren. Enabling Online Robust Barcode-Based Visible Light Communication With Realtime Feedback. *IEEE Transactions on Wireless Communications*, 17(12):8063–8076, Dec 2018.
- [19] Y. Zou, J. Zhu, X. Wang, and L. Hanzo. A Survey on Wireless Security: Technical Challenges, Recent Advances, and Future Trends. *Proceedings of the IEEE*, 104(9):1727–1765, Sep. 2016.

Video Crowdsourcing in Vehicular Networks: Use Cases and Challenges

Sataponn Phutrakul

sataponn.phutrakul@aalto.fi

Tutor: Chao Zhu

Abstract

In recent years, many drivers have installed cameras on vehicles for their own benefits, and modern vehicles are built with computing and communication capabilities. This enables the rise of a new concept called vehicular video crowdsourcing, which utilizes videos recorded by in-vehicle cameras for solving several issues on a larger scale, including vehicular problems. This article reviews vehicular network paradigms that can be used for video crowdsourcing and presents applications of vehicular video crowdsourcing with brief workflows. This article also summarizes their issues in four aspects: security, privacy, reliability and efficiency.

***KEYWORDS:** video crowdsourcing, vehicular networks, vehicular communications, vehicular fog computing*

1 Introduction

An increasing number of vehicle owners have realized the importance of vehicle cameras, such as dash cameras and rear cameras. A dash camera is installed for recording incidents on the road, which provides strong evidence in the event of an accident. On the other hand, rear cameras as-

sist drivers while driving in reverse gear. In other words, they are mainly used for personal benefits. In fact, these tools can also lead to solutions for various vehicular problems, including the difficulty to search for an available parking spot and unawareness of road conditions and accidents. This is owing to the development of modern vehicles that are equipped with computing and communication devices, which enable the vehicles to interconnect with each other and their surroundings, such as roadside units (RSUs) [3].

Videos from vehicles and communication systems allow the idea of vehicular video crowdsourcing to emerge. Crowdsourcing is a process in which an organization or company outsources tasks to a group of willing participants through an open network instead of performing the tasks by themselves, thereby reducing the cost of task completion and time [12]. The participants may get paid once the task results are evaluated. In vehicular video crowdsourcing, videos are collected from participating vehicles and analyzed with the aim of achieving particular tasks [10].

One way to obtain videos from a connected vehicle is to use a cellular network for uploading the videos to a cloud server. This is called vehicle-to-cloud (V2C) communications [1]. However, videos are relatively large compared to other formats of data, and they will consume a lot of data bandwidth. Additionally, some video crowdsourcing applications require data to be analyzed on a real-time basis, which is impracticable due to unstable mobile network connections depending on the operator, the latency between the vehicle and the server, and the limited computational capabilities of a single vehicle [17].

A proposed solution to these limitations is adopting the vehicular cloud computing (VCC) paradigm and deploying RSUs, which are responsible for collecting data from vehicles and managing data transfer to the cloud [2]. However, it is costly and challenging in real deployment, and the latency issue is still unresolved.

In addition to VCC, there have been researchers who applied fog computing to vehicular crowdsourcing. With vehicular fog computing (VFC), nearby vehicles collectively become an infrastructure, and videos can be processed within the local area by utilizing the computing power of these vehicles [6]. As a result, time-sensitive applications become more feasible.

Having briefly mentioned different approaches to video crowdsourcing implementation in vehicular networks, this article aims to focus on various vehicular scenarios in which video crowdsourcing can be beneficial

(e.g., solving the aforementioned vehicular issues). In addition, the feasibility and challenges of each video crowdsourcing application are discussed along with problems that could arise from these applications.

The remainder of this article is organized as follows. Section 2 provides background information about vehicular networks and video crowdsourcing. Section 3 introduces use cases of vehicular video crowdsourcing. Potential issues regarding security, privacy, reliability and efficiency are analyzed in Section 4. Finally, the conclusion is presented in Section 5.

2 Background

This section introduces vehicular communication technologies and vehicular network paradigms, which serve as building blocks of vehicular video crowdsourcing applications, as well as how vehicular video crowdsourcing works.

2.1 Vehicular Communications

Vehicular communications are categorized based on the kinds of objects with which a vehicle communicates [5]. There are two major types of vehicular communications. One is vehicle-to-vehicle (V2V) communications, which enable vehicles to connect to one another within a reachable range. To put it another way, vehicles in a small area form an ad-hoc network without a controller. V2V communications are suitable for situations in which nearby vehicles need to exchange traffic information in a time-efficient way. Another type of vehicular communications is vehicle-to-infrastructure (V2I) communications, which are also known as vehicle-to-roadside (V2R) communications. V2I aims to address the range limitation of V2V by including connected RSUs in the network. This allows a vehicle to send information to another vehicle located at a great distance.

Recently, the idea of the Internet of Things has gained attention and become one of the main topics in the technology industry. Consequently, with the need for connecting vehicles to a wider variety of devices, researchers have proposed vehicle-to-everything (V2X) communications, which combine V2V, V2I and other types of vehicular communications, such as vehicle-to-pedestrian (V2P) [16]. This invention supports the development of several intelligent vehicular systems.

2.2 Vehicular Networks

A substantial amount of vehicular network paradigms have been introduced to overcome the limitations of the preceding one. The simplest paradigm is vehicular ad-hoc networks (VANETs) [2]. In a VANET, vehicles with on-board units (OBUs), which have the ability to communicate using wireless signals, are connected together with RSUs. This network is enabled by V2V and V2I communications. However, the available resources solely equipped on a vehicle are not sufficient for many applications to operate smoothly and efficiently.

Therefore, another paradigm called vehicular cloud computing (VCC) was developed to solve this issue. With VCC, VANETs are integrated with cloud computing, which allows vehicles to obtain on-demand storage, communication and computing capabilities from cloud services. In most use cases, data from connected vehicles in a VANET are gathered and transmitted to cloud servers by RSUs. Despite the unlimited resources supplied by cloud computing, the Quality of Service (QoS) of applications relies significantly on unpredictable Internet connections, resulting in a high probability of delayed requests and responses between the vehicles and the servers. This prevents most applications from achieving their expected QoS. Furthermore, for extremely popular applications that are used by numerous vehicles, the cloud servers themselves may not be able to serve all incoming workloads. Sometimes it can lead to server failures.

To avoid such problems, researchers recommend a new paradigm named vehicular fog computing (VFC) consisting of geographically distributed fog nodes, which lie between the cloud and vehicles. In other words, it is a decentralized architecture in which cloud services are brought closer to vehicles. Either vehicles with networking and computing capabilities or RSUs can act as fog nodes. A fog node will receive data from vehicles in the local area. Data for low-latency applications and data that are useful only within the area are processed at this fog node; otherwise, they are forwarded to the cloud. This facilitates the development of time-sensitive vehicular applications, and the vehicles are only aware of information relevant to them. Moreover, cloud servers do not have to take responsibility for processing all data.

2.3 Vehicular Video Crowdsourcing

Vehicular video crowdsourcing describes a paradigm in which a large number of vehicles share videos captured by equipped cameras in order to accomplish crowdsourcing tasks [10]. Some crowdsourcing applications also reward participating vehicles for their contributions.

Vehicular video crowdsourcing systems can be implemented in different ways depending on the kind of vehicular networks on which the systems are based. For example, a fog-based vehicular video crowdsourcing system is composed of the cloud, fog nodes, vehicles and customers. Customers, e.g., vehicles and organizations, can benefit from the outcomes of crowdsourcing. The system is divided into three layers: service layer, fog layer and vehicle layer.

In the service layer, a customer assigns a task to the crowdsourcing service, which will forward the task to the fog nodes in the target area. Then, the fog nodes in the fog layer gather relevant videos and upload them to the service for further processing and output delivery to the customer. Lastly, in the vehicle layer, vehicles record videos with the equipped cameras, search for nearby fog nodes, send requests containing metadata of the videos, and transmit the videos to the fog nodes [17]. In some use cases, the fog nodes can also operate as a crowdsourcing service.

3 Applications of Vehicular Video Crowdsourcing

There have been several proposals of vehicular applications that use video crowdsourcing in order to provide convenience to drivers or solve a variety of problems, including vehicular issues. This section provides examples of these applications and discusses their feasibility.

3.1 Driving Assistance

State-of-the-art vehicles are equipped with sensors, cameras and smart systems that assist drivers by keeping them informed about the surrounding environment and notifying them when obstacles are detected. However, there are more incidents beyond the detection range of a vehicle that the driver should be aware of, and the driver has to mainly rely on himself with extreme caution. With a video-crowdsourcing-based driving assistance, the driver will be aware of traffic situations, road conditions

and other factors, such as weather, accidents and construction, on a large scale ahead of time in a real-time manner. Moreover, the assistance can suggest an alternative route to avoid unpleasant conditions.

Other than real-time context detection, a driver will be provided with a streaming video from a dash camera on another vehicle in suitable situations, e.g., changing lanes in order to pass a larger vehicle, thus improving safety of the driver and passengers [11]. Cooperative lane changing aided by the assistance also prevents accidents and inefficient traffic flow.

Many researches have been conducted with the focus on intelligent driving assistance systems. Hu et al. [7] adopted the concept of vehicular crowdsourcing to enhance an ordinary real-time traffic map, which is required in driving assistance systems. In addition to sensor data, they utilized videos recorded by in-vehicle cameras to provide more detailed traffic information to users, e.g., warning drivers when an animal is detected on the road. In their implementation, vehicles uploaded videos to the cloud over 4G LTE for analytics and map generation. They improved system scalability and bandwidth consumption by using edge computing with cloudlets that collected data, analyzed them and transmitted small outputs to the cloud.

Driving assistance systems need to act in response to traffic data in real time; therefore, video processing has to be fast and highly accurate. This is feasible when running state-of-the-art algorithms on existing models of processing units. However, the accuracy also depends on the set of images used for training the detector. Moreover, some objects, such as potholes, are difficult to detect. As a result, it is unlikely to develop a system that achieves ideal performance.

3.2 Parking Navigation

A number of parking lots, especially in shopping malls, have installed sensors and colored lights to indicate the locations and number of available parking spots on each floor. However, the accuracy and reliability of these systems are moderately low, and they are not well-maintained. Additionally, roadside parking spots are not monitored for availability. As a consequence, it is difficult for drivers to find places to park their vehicles, and vehicles looking for parking spots contribute as one of the causes of traffic congestion [17].

Instead of an investment in a parking space monitoring system, which requires additional construction work and an overhead, vehicular video

crowdsourcing serves as a more efficient solution at a lower cost. In this video crowdsourcing application, free parking spots can be recognized from the crowdsourced videos, and drivers are notified and conveniently navigated to the spots.

Such application can be implemented based on VFC in order to provide real-time parking information to drivers [10]. Vehicles in the network record videos with dash cameras and upload them to near fog nodes. Vehicles that are looking for parking spaces act as service customers and send queries to the crowdsourcing service in the cloud. Once the service receives a query, it collects videos from the fog nodes in the destination area, analyzes the videos to identify an available parking spot, and reports the result to the customer.

The performance of this application relies heavily on image processing for detecting available parking spots. It is a challenging task to identify a parking space when the environmental conditions vary [4]. Additionally, with image processing alone, the system cannot verify that the space is a legal parking spot. Other creative approaches can be taken in order to overcome these challenges. For instance, Grassi et al. detected parked vehicles instead and incorporated public parking information to obtain empty legal parking spots. Another problem is that the retrieved GPS location might not be precise enough to locate the detected spot. Nevertheless, this can be solved with a localization algorithm.

3.3 City Infrastructure Improvements

Road and city infrastructure conditions change over time. To maintain their good conditions for the safety of drivers, passengers and pedestrians, they have to be monitored constantly, and this requires an extremely large amount of human resources and time. Vehicular video crowdsourcing can be employed to reduce the cost of employment and avoid human mistakes [17].

In this scenario, in-vehicle dash cameras record videos, which contain surrounding infrastructures, as the vehicles move around the city. Then, these vehicles upload the videos to fog nodes or cloud servers [10]. Organizations or public sectors that are in charge of city infrastructures can send queries to the crowdsourcing service asking for the locations of the areas that need to be investigated, repaired or improved. The cloud server will detect those in question from the videos and respond accordingly. Implementing this system in the real world could be challenging due to varying

and unpredictable patterns of infrastructure damage.

3.4 Accident Reconstruction

Accidents on the road happen every day. Those who have dash cameras installed on their vehicles can use the recorded videos to defend themselves and prove the other parties' fault. On the other hand, in case none of the parties has cameras, they can only explain the situation to the car insurance officer by mouth according to what they see, remember and understand. Drivers with the lack of traffic knowledge may interpret the situation incorrectly, or both parties might try to lie and blame each other. This results in a misjudgement about the cause of the accident.

Videos from other vehicles can be valuable sources of evidence for these accidents [10]. Furthermore, if a driver escapes after causing an accident, these videos can be analyzed to track the vehicle and its owner. However, dash cameras generally do not store videos for a long period of time. This problem can be solved with VFC-based video crowdsourcing. These short-lived videos are uploaded to fog nodes and the cloud afterwards, and the cloud can acquire evidence from the videos recorded in the areas where collisions occur according to queries from insurance officers and police, who will reconstruct accidents for further investigation. In vehicles that can store videos for an extended period, it is possible to enhance network resource utilization by collecting only metadata of videos, e.g., locations, from the vehicles and requesting for the actual videos when needed [13].

Other approaches can be used for applying vehicular video crowdsourcing in this scenario as well. Instead of gathering videos at a cloud server and letting the server filter for relevant videos, Ford Global Technologies LLC included event reporting in their implementation [8]. A crowdsourcing server is notified when an accident happens, and it signals to the connected vehicles in the vicinity of the accident to record the surroundings with their sensors and cameras. These data, including videos, are analyzed for reconstructing the accident. This has proved the feasibility of vehicular video crowdsourcing for accident reconstruction even without the use of VFC.

3.5 Criminal Investigation and Crime Scene Reconstruction

There are a considerable number of criminal cases in which the victims lack witnesses and clear evidence that can be used for investigation and

sentence enforcement. Although surveillance cameras have been installed in most buildings and at intersections, not all of them are working properly. They might be broken when a crime takes place, or the recorded videos are too blurry to identify the criminal. In addition, they can be easily noticed by the criminal, so he can destroy them before committing the crime.

Vehicular video crowdsourcing can be beneficial in this situation since there could be plenty of videos recorded by vehicles passing by the crime scenes from different angles [17]. These videos can be used to reconstruct the crime scenes. The implementation and feasibility of this type of applications are similar to those for accident reconstruction explained in Sec 3.4.

4 Challenges and Issues

Since vehicular video crowdsourcing is gaining interest from researchers and companies, it is important to consider and elaborate its challenges and issues. This section discusses various problems in the aspect of security, privacy, reliability and efficiency.

4.1 Security

In spite of the benefits brought by video crowdsourcing applications, they also pose significant threats [10]. Videos can be found at vehicles, fog nodes and cloud servers, and they may contain sensitive information. These system components could be vulnerable to attacks. Internal or external attackers who successfully gain access to the videos and generated crowdsourcing reports can infer confidential information from these data. Other than attacking on data confidentiality, attackers can also maliciously control the crowdsourcing process and results directly [14]. Encryption and digital signatures can ensure data confidentiality and integrity, but it is not completely feasible because vehicles might not have sufficient capability to perform cryptographic operations.

In addition, crowdsourcing systems without proper authentication mechanisms are not protected against impersonation and Sybil attacks. An attacker might impersonate a large number of different identities and flood the server with forged videos in order to earn rewards and manipulate the result of a crowdsourcing task, which affects the action taken afterwards

by the crowdsourcing customer. This could cause danger or difficulties to the customer. Hence, it is necessary to ensure that data in a crowdsourcing system come from trustworthy and authenticated sources.

4.2 Privacy

When drivers decide to participate in video crowdsourcing systems, it is inevitable that they will lose privacy to some extent. Videos recorded on their vehicles contain places that they visit and routes that they take. This can imply personal information, such as the locations of their houses and their travel behavior [10]. The connection established between a vehicle and a fog node in case VFC is used can also indicate a rough location of the vehicle. Crowdsourcing customers may experience similar problems as well. The details of the tasks that a customer has issued can reveal his sensitive information, such as reasons for issuing the tasks.

There are several solutions involving anonymity techniques and cryptography that preserve privacy and still allow authentication. Ni et al. [9] implemented a privacy-preserving navigation system in which vehicles uploaded data to RSUs with signatures that did not reveal their identities, and customers sent queries to RSUs anonymously. To prevent vehicles from providing false data, the system had a trusted authority (TA), which was aware of the real identities of these vehicles.

4.3 Reliability

A vehicular network consists of both moving and parked vehicles, and the communication in the network is based on wireless connections. Issues regarding the reliability of the vehicular network, which depends on the network paradigm and wireless technologies employed in the system, should be considered and diminished, specifically for time-sensitive applications [14].

A crowdsourcing system will work effectively if the number of participating vehicles is large enough. This causes many wireless connections within small areas, thus leading to potential interference and data loss [17]. Moreover, in VFC, a connection between a vehicle and a fog node can be interrupted as the vehicle moves away from the fog node. In systems that require the use of cellular networks, it is not guaranteed that all areas are under the coverage, especially rural areas. Furthermore, the cellular bandwidth is limited.

4.4 Efficiency

Efficiency of a system depends on the system architecture, network paradigm and wireless technologies as well. In most vehicular video crowdsourcing systems, vehicles have to transmit video files, which are usually large, to fog nodes, RSUs, cloudlets or cloud servers. However, not all of these videos are actually utilized for crowdsourcing tasks. Moreover, crowdsourcing tasks that do not involve other areas are not necessary to be sent to and processed in the cloud. This indicates that the communication, computation and storage resources have been wasted.

[8] and [13] showed useful approaches that can improve resource utilization. In addition to these approaches, VCC, VFC and edge computing can significantly raise system efficiency while still maintaining high QoS. Firstly, it provides a pool of resources from vehicles and other components in the local infrastructure [15]. Secondly, it allows crowdsourcing operations and decision making processes to be performed locally, which saves network bandwidth required for data transmission to the cloud and cloud resources. However, it is important to manage fog nodes or cloudlets efficiently, which is another challenge in vehicular networks.

5 Conclusion

In this article, the usefulness of videos recorded from vehicles are presented along with general background knowledge about vehicular communications, vehicular networks and video crowdsourcing in vehicular networks. This article describes five scenarios in which vehicular video crowdsourcing can be beneficial, difficulties in implementing systems in these scenarios and existing researches. Finally, this article discusses potential security, privacy, reliability and efficiency issues in general, which can vary in different implementations.

There are several more applications of video crowdsourcing in vehicular networks that have not been mentioned in this article. It is worth investigating in detail since this innovation has the potential to be widely deployed in the future.

References

- [1] A. Bazzi and A. Zanella. Position based routing in crowd sensing vehicular networks. *Ad Hoc Networks*, 2015. doi: 10.1016/j.adhoc.2015.06.005.
- [2] G.S.G. Deepika and P.S. Kiran. Vehicular fog computing: The need for a new paradigm and its issues. *International Journal of Engineering & Technology*, 2018. doi: 10.14419/ijet.v7i2.7.10890.
- [3] Center for Advanced Automotive Technology. Connected and automated vehicles. [Online]. Available: http://autocaat.org/Technologies/Automated_and_Connected_Vehicles, 2018. [Accessed: 26-Jan-2019].
- [4] G. Grassi, P. Bahl, K. Jamieson, and G. Pau. Parkmaster: An in-vehicle, edge-based video analytics service for detecting open parking spaces in urban environments. Oct 2017. doi: 10.1145/3132211.3134452.
- [5] S.F. Hasan. Vehicular communication and sensor networks. *IEEE Potentials*, 32, Jul 2013. doi: 10.1109/MPOT.2012.2225172.
- [6] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen. Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE Transactions on Vehicular Technology*, 65, Jun 2016. doi: 10.1109/TVT.2016.2532863.
- [7] W. Hu, Z. Feng, Z. Chen, J. Harkes, P. Pillai, and M. Satyanarayanan. Live synthesis of vehicle-sourced data over 4g lte. Nov 2017. doi: 10.1145/3127540.3127543.
- [8] Ford Global Technologies LLC. Crowdsourced event reporting and reconstruction. United States Patent 0017734, Jan 2017.
- [9] J. Ni, X. Lin, K. Zhang, and X. Shen. Privacy-preserving real-time navigation system using vehicular crowdsourcing. 2016. doi: 10.1109/VTCFall.2016.7881177.
- [10] J. Ni, A. Zhang, X. Lin, and X. Shen. Security, privacy, and fairness in fog-based vehicular crowdsensing. *IEEE Communications Magazine*, 55, Jun 2017. doi: 10.1109/MCOM.2017.1600679.
- [11] D. Sabella, H. Moustafa, P. Kuure, S. Kekki, Z. Zhou, A. Li, C. Thein, E. Fischer, I. Vukovic, J. Cardillo, V. Young, S.J. Tan, V. Park, M. Vanderveen, S. Runeson, and S. Sorrentino. *Toward fully connected vehicles: Edge computing for advanced automotive communications (White Paper)*. 5GAA, 2017.
- [12] P. Whitla. Crowdsourcing and its application in marketing activities. *Contemporary Management Research*, 5, Mar 2009. doi: 10.7903/cm.1145.
- [13] Y. Wu and G. Cao. Videomec: A metadata-enhanced crowdsourcing system for mobile videos. Apr 2017. doi: 10.1145/3055031.3055089.
- [14] Y. Xiao and C. Zhu. Vehicular fog computing: Vision and challenges. 2017. doi: 10.1109/PERCOMW.2017.7917508.
- [15] P. Yang, N. Zhang, Y. Bi, L. Yu, and X. Shen. Catalyzing cloud-fog interoperation in 5g wireless networks: An sdn approach. *IEEE Network*, 31, Sep 2017. doi: 10.1109/MNET.2017.1600078.

- [16] S. Zhang, J. Chen, F. Lyu, N. Cheng, W. Shi, and X. Shen. Vehicular communication networks in automated driving era. *IEEE Communications Magazine*, 56, Sep 2018. doi: 10.1109/MCOM.2018.1701171.
- [17] C. Zhu, G. Pastor, Y. Xiao, and A. Yläjääski. Vehicular fog computing for video crowdsourcing: Applications, feasibility, and challenges. *IEEE Communications Magazine*, Oct 2018. doi: 10.1109/MCOM.2018.1800116.

GPS jamming and spoofing

Jussi Hietanen

jussi.hietanen@aalto.fi

Tutor: Tuomas Aura

Abstract

The Global Positioning System (GPS) has grown to a universally trusted source for navigation and timing data. However, the GPS signals are vulnerable to jamming and spoofing attacks. Therefore, one should never trust or be totally dependent of data received from GPS.

This paper should give the reader an understanding of what the GPS signals consists of, how they are created and how the receiver calculates its own position using the received signals. The paper also describes how this process can be interfered with a jamming attack, and also how the receiver's position can be deceived with fake GPS data broadcast more powerfully than the legitimate signals. In the end we discuss about different countermeasures against these attacks.

KEYWORDS: *GPS, Global, Positioning, System, Jamming, Spoofing*

1 Introduction

The Global Positioning System (GPS) is a globally used radionavigation system, operated by the United States. It provides positioning and time synchronization service for military and commercial use. The positioning service of GPS is used by all sorts of land, sea and air traffic. Timing signals are used in banking and financial industries for money transfers and time locks. In short, GPS is widely trusted to provide real position and time information to the receiver.

Unfortunately, only the military GPS signals are secured by encryption [1]. This makes the unencrypted, commercial GPS signals insecure, as the origin of the signal cannot be verified. The GPS signal is also very weak in the receiving end, which makes it vulnerable to simple jamming attacks.

In the recent years, there have been claims of attacks against GPS services carried out by nations' governments; there have been claims¹ that Russia has executed location spoofing attacks. One of these suspected attacks happened during a major NATO military exercise². Russia has also been suspected to use GPS spoofing as a way to protect its statesman Vladimir Putin³.

As Software Defined Radios (SDRs) have become cheaper and more popular in the last few years, anyone with a basic radio technology knowledge could execute a short-range GPS location (or time synchronization) spoofing attack for under 300 US dollars put into hardware[8, 11].

This paper will gather together the most common attacks against a GPS receiver and their possible countermeasures. In the section 2, we will go through the fundamentals about how the GPS works. Sections 3 and 4 describe the jamming and spoofing attacks against a GPS receiver. In section 5, we will discuss various prevention methods against a GPS attack. We conclude the paper in section 7.

¹<https://maritime-executive.com/editorials/mass-gps-spoofing-attack-in-black-sea>

²<http://www.thedrive.com/the-war-zone/15194/russia-jammed-phones-and-gps-in-northern-europe-during-massive-military-drills>

³<https://nrkbeta.no/2017/09/18/gps-freaking-out-maybe-youre-too-close-to-putin/>

2 The Global Positioning System

The development of GPS originally started from the needs of the United States military for a better and universal positioning system in 1973. It was made for the military first, but was later made available for civilian use as well. Today it is a dual-use service, providing encrypted positioning data for the military and unencrypted for civilian applications. [1]

GPS is a passive (one-way) system, which means that the GPS receiver only receives GPS radio signals. In addition to providing the position, the system also provides time information to the receiver. The GPS receiver requires signal from at least four GPS satellites to determine its position, whereas the time information only requires signal from one GPS satellite. [1]

This section will describe the theory of operation of GPS, focusing on the trilateration and radio communication principles.

2.1 GPS Trilateration

Trilateration determines the receiver's location by using multiple distances to objects with a known position. To detect the position using trilateration in a receiving-only communication channel where the sender (the satellite) only broadcasts its own time and location information, the receiver should also know its own clock offset (receiver's internal clock error). This is why four satellites are needed; the clock offset adds a fourth unknown besides the 3-dimensional coordinates x , y and z . These four unknown quantities must be solved for the precise positioning, as shown in the figure 1.

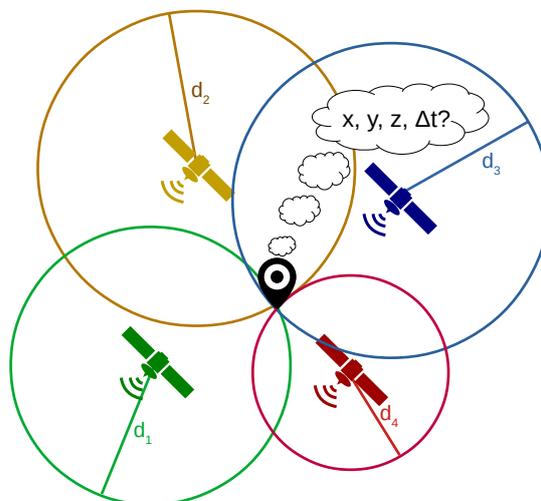


Figure 1. GPS receiver using trilateration to determine its position and clock offset.

Trilateration should not be confused with triangulation, where angles to known points are used to determine the location. Because the receiver in GPS does not know the direction of the received signal in trilateration, the (spoofed) signal could also be broadcast by someone on the ground without receiver's knowledge.

2.2 GPS Signals and Frequencies

The location providing signals of GPS can be divided to legacy signals and modernized signals. This section will only describe the legacy GPS signals in depth, as most of the satellites and GPS receivers do not yet support the modernized signals.

Legacy GPS works by transmitting navigation data and chipping codes. The navigation data message contains the data needed for positioning: the satellite's time and position. It also contains the ephemeris data, which gives the receiver the trajectory of the satellite. The pseudorandom noise (PRN) chipping codes that carry no actual data are called coarse/acquisition (C/A) code and precision (P) code. The encrypted P code is only for military use, whereas the C/A code is meant for civilian use. Both of these PRN codes are generated by linear feedback registers that are unique for each satellite, which makes the GPS system a Code Division Multiple Access (CDMA) network. Because the civilian C/A codes are publicly known, the receiver can decode the civilian signal and knows which satellite it is receiving the signal from.

The navigation signal coding for L1(C/A) and L2 signals happens by first adding the navigation data with a modulo 2 sum (xor) to both PRN codes. The resulting signals are mixed to the carrier wave using Binary Phase Shift Keying (BPSK) modulation and then broadcast as shown in figure 2. [2] Because the L1 and L2 carrier wavelengths are multiples of the other signals, the modulation results in a clean 2-phase BPSK signal with phases of 0 and 180 degrees.

The navigation data message consists of 25 frames with 1500 bits of information in each. Every frame has a 900-bit part of satellite data including the satellite's ephemeris and clock correction data, which is updated in every frame. The remaining 600-bit system data takes 25 frames to transmit. At 50Hz (50 bits per second) this takes 12.5 minutes. Even though the satellite data updates on every frame, each frame still takes 30 seconds to complete. This means that the receiver must predict the position of the satellite and calculate its own position by the C/A or P code,

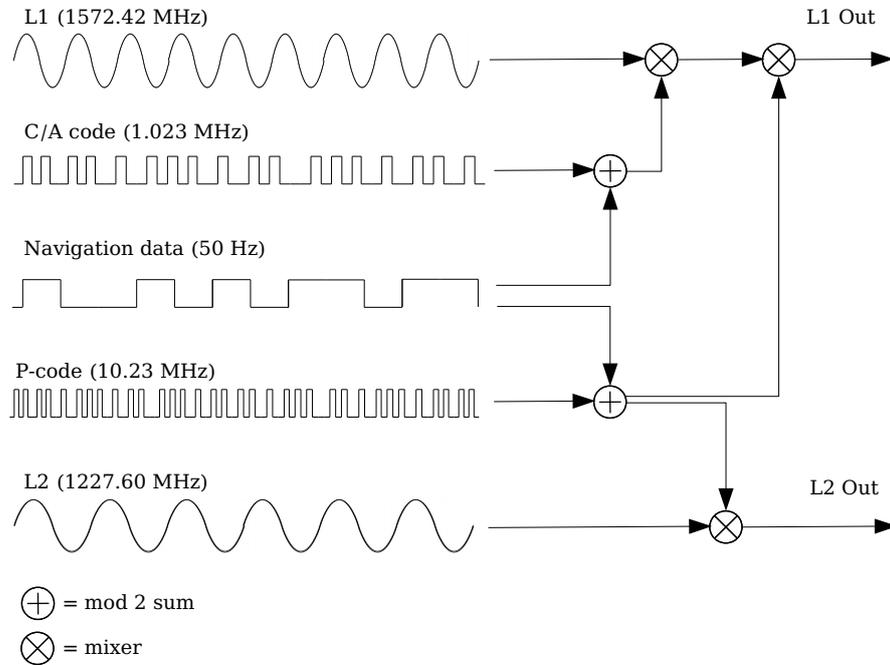


Figure 2. The mixing of L1 and L2 signals. [2]

which is transmitted more frequently.

Because the C/A code is transmitted at 1.023 MHz, the distance between two chips is about 300 meters. With for example 1% error in the measured chip distance at the receiver caused by variations in the signal transmitted in the ionosphere and receiver clock jitter and processing errors, this translates into roughly 3 meters of position error in the GPS position. The military-used P-code is ten times more accurate because of its higher frequency.

A total of five different fixed signal frequencies carry GPS data from the satellites to the users, but currently only two of those frequencies are used for position tracking. The signals L1(C/A) and L2(C) (C meaning civilian) are transmitted on frequencies 1575.43 MHz and 1227.60 MHz and contain the needed information for GPS position services. Other signals include L3 used to broadcast information globally in a case of nuclear detonation events, L4 for studying correction against signal biases caused by the atmosphere, and L5 for future civilian safety-of-life use. L5 is broadcast on a frequency where no other signals than GPS data should ever be sent. All the signals and their frequencies are presented in the table 1.

The modernized positioning signals were implemented after the GPS system had reached its full operational capability in 1995. The L2C was first introduced as a new civilian-use signal with better accuracy and carrier-tracking than the original L1 C/A signal. It could be also used as

Signal	Frequency	Use
L1(C/A))	1575.43 MHz	Legacy: P code for military use, C/A code and navigation data for both Modern: M code for military use, L1C for civilian use
L2(C)	1227.60 MHz	Legacy: P code and navigation data for military use Modern: M code for military use, L2C for civilian use
L3	1381.05 MHz	Informs about nuclear detonation events
L4	1389.913 MHz	Studied for additional correction against solar ionization in atmosphere
L5	1176.45 MHz	Proposed for civilian aviation safety-of-life positioning signal

Table 1. GPS signals and their use. [1]

an assisting signal in case of interference on the L1 frequency. L1C was designed to add interoperability between GPS and Galileo L1 satellites for civilian users. Galileo is a GNSS system operated by EU and ESA. L1C also includes the legacy L1 C/A code for backward compatibility with legacy receivers. The M code was designed for military use to protect the signals from jamming attacks and to make them more secure. It is broadcast on L1 and L2 frequencies, but because of its classified status, not much more of it is known publicly. [1]

3 Jamming Attacks Against GPS

Radio jamming is an attack where the attacker has a radio transmitter broadcasting a signal to block the reception of the real radio signal [9]. This requires the jamming signal strength to be more powerful than the real signal. As the GPS signal is a radio wave with a fixed frequency and its amplitude is low due to the attenuation from long distance, it can be jammed easily [3].

Radio jamming is a less sophisticated attack compared to the spoofing attack. Therefore, it is more likely to be used in combat situation as an attack instrument of electric warfare. In a combat situation, the jamming attack tries to block the information exchange of the enemy. For GPS

jamming, this would most likely mean blocking GPS location acquisition for enemy airplanes, maritime navigation and GPS-guided missiles. [9]

Different types of cheap and illegal GPS signal jammers were studied by Mitch et al. [6] in 2011. The tested jammers' transmitting power varied from a fraction of a watt to a couple of watts. The study found that most of the jammers are specifically for jamming L1(C) and L2(C) signals, which are the only signals carrying the code needed for position acquisition in the legacy GPS. Many of the tested jammers were rather inaccurate, both for frequency and advertised effective range. Commonly the jammers swept over the GPS frequency band, and the sweep bled over to non-GPS frequencies. Many of the jammers also jammed WiFi and cellular phone frequencies. Some of the jammers blocked the GPS signal reception from hundreds of meters away, when the advertised effective jamming range was stated to be only tens of meters. The most powerful jammer was calculated to block GPS position acquisition from almost 9 km distance in perfect environmental conditions.

Military-grade radio jammers have generally much more transmitting power than the cheap civilian jammers or transmitters. The effective range of a GPS jammer was tested in a more concrete study done by Grant et al. [4] in 2009. In the study, a jammer with 1.5 watt transmitting power was able to block acquisition of GPS position from the distance of 30 kilometers. The study also suggested that a more powerful very high frequency (VHF) transmitter could cause errors of multiple kilometers or complete loss of GPS position from hundreds of kilometers away.

4 Spoofing Attacks Against GPS

Jamming attacks against a GPS receiver are not the biggest security risk because the receiver will be aware of the lost GPS signal and use other means to determine its position. The GPS position spoofing attack is more sophisticated and possibly much worse than the jamming attack because the GPS receiver most likely cannot detect it. The spoofing attack requires the attacker to broadcast fake GPS signals so that the receiving victim calculates its location incorrectly. [13] Usually the victim of the attack is first connected to legitimate GPS satellites, but the legitimate signals are taken over by more powerful spoofing signals as show in the figure 3 [10].

Warner et al. [12] conducted a study on how easily civilian GPS receiver (receiving legacy L1 C/A-signal) position could be spoofed. In their study,

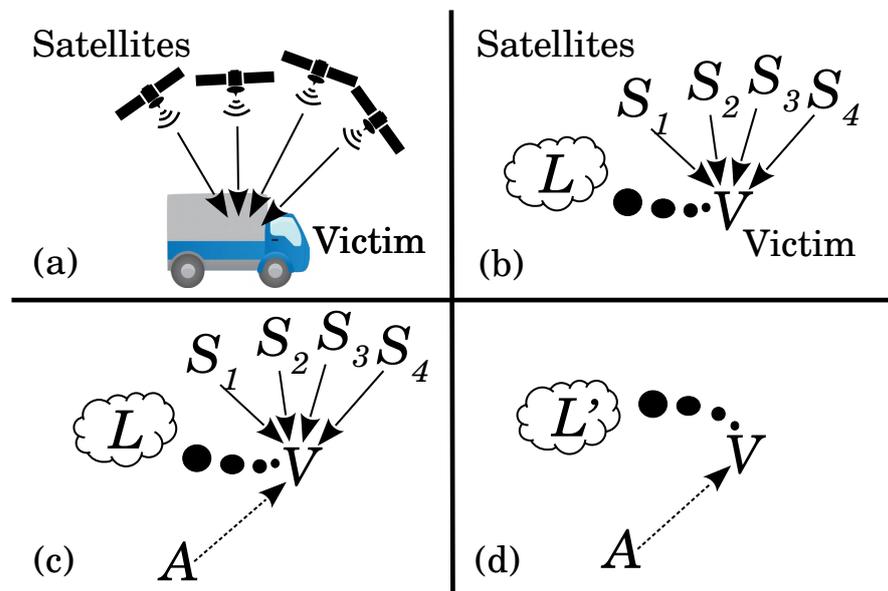


Figure 3. Basic spoofing attack scenario. (a) Visualization of the setup. The victim uses a GPS-based localization system and is synchronized to the legitimate satellites. (b) Abstract representation of the scene. (c) The attacker starts sending own spoofing and jamming signals. (d) The victim synchronizes to the attacker's signals. [10]

they used an industrial-made GPS satellite simulator, which was capable of simulating 10 GPS satellite signals at once. They also had to use a GPS signal amplifier as the simulator was meant to feed the GPS signal only through a cable. They put this hardware setup into the cab of a truck, and they were able to mislead nearby GPS receivers from 10-20 meters of distance with low transmitting power. Attacks against another moving truck's GPS receiver were also carried out successfully. All the hardware for this attack was acquired for a total cost of less than 2000 US dollars. During the tests, they noticed that taking over the legitimate satellite signal lock down took anything between 20 seconds and 3 minutes. This is logical if the spoofed satellite constellation does not match the legitimate constellation, as the spoofing signal needs to update the receiver's ephemeris data as described in the section 2.2. The lock-down delay is also backed by a more theoretical study conducted by Tippenhauer et al. [10]. The suddenly changed satellite constellation does not cause an alarm, because the standard GPS receivers do not check for any data consistency [7].

Spoofing attacks can also be carried out against the authenticated, military-only GPS receivers. The attacker is not able to generate its own GPS signals, but recorded legitimate signals can be replayed. Theoretically, the attacker could capture the signals from known satellites, separate them

and delay each signal so that the time deltas simulate the wanted location for the receiver. This is called *selective-delay attack*. [10]

Software Defined Radios (SDRs) have become more powerful in the last years. They are also very affordable; for example a HackRF SDR board can be purchased for under 300 US dollars at the time of writing of this paper. Wang et al. [11] conducted a study on GPS spoofing with open source code and SDR boards. They used open-source code from a GitHub project⁴ with HackRF and BladeRF boards for successful attacks against different cellphone GPS. They were able to simulate 13 satellites simultaneously, which is greater number than achieved by the industrial GPS simulator used by Warner et al. [12] in 2002.

5 Countermeasures against GPS attacks

The current civilian GPS receivers do not have any built-in detection of GPS spoofing attacks, as all the receivers have very similar GPS receiver circuitry [12]. The simplest place for implementing GPS spoofing detection would be in the software stack that receives the position information from the GPS unit. The software could do some sort of sanity checks if the position of the receiver jumps from one place to another instantly. However, this kind of checks would not detect sophisticated attacks like the one carried out by Warner et al [12].

Using a directional antenna has been a traditional countermeasure against radio jamming and spoofing attacks. Directional antenna would attenuate the attacker's jamming or spoofing signal and raise the signal-to-noise ratio (SNR) of the legitimate radio signal enough so that the legitimate information could be received. This applies to GPS as well because the legitimate GPS signals come from the sky, whereas the spoofing or jamming signals most likely will originate from ground or sea level [5].

Wen et al. [14] listed possible spoofing detection methods, such as monitoring the power of the received signals, relative power of the signals, and Doppler shifts. However, most of the detection methods would need to work in the GPS receiver chip, and since they are not implemented in today's receivers, we are not going to describe them in detail.

Tippenhauer et al. [10] proposed a solution of a static formation of GPS antennas for receiving GPS signals. If the attacker uses only one antenna to transmit the GPS signals, two receiver antennas in known positions

⁴<https://github.com/osqzss/gps-sdr-sim>

relative to each other would detect the spoofing. If the attacker would be using more than one antenna, four receiving antennas in a known formation would make it impossible to spoof the location so that all the receivers would stay in the same formation when the attacker tries to spoof the locations of the receivers.

The modernized GPS signals included the new M code, which is meant to counter spoofing threats on the military-used authenticated GPS. This might mean that a way to achieve continuous authentication was added to the signal. However, the method is not publicly known. The modernized signals also include the new L1C and L2C signals for civil use, but as they are publicly known, they can still be spoofed. [1]

In case of a jamming attack with signal power close to the legitimate GPS signal, the receiver could employ Russian Global Navigation Satellite System (GLONASS) in addition to GPS to provide better receiver performance because of its better SNR performance. The satellite constellation of GLONASS also provides better coverage in some areas, such as the Northern sea. However, if the jamming signal is substantially more powerful than the legitimate signals, using GLONASS would not help because of their close signal frequencies (1602 MHz and 1575.43 MHz). [3]

6 Discussion

The referred studies demonstrated how the GPS receivers did not alert any anomalies regarding GPS positions or satellite constellations suddenly changing during the lock down of the spoofing GPS signal [11, 12, 7]. This is questionable, because for example the mobile phones of today could warn the user about the sudden change in the GPS data. However, we cannot trust that every software developer creating a GPS-dependent application were to implement this kind of anomaly detection in the software. Therefore, detection methods against GPS attacks should be implemented and deployed to GPS receiver chips themselves. This would allow alerting of possible GPS attacks in the operating system level in mobile phones and in system level in other sorts of GPS-positioning applications.

The GPS receiver could also use Assisted GPS (A-GPS) to check the plausibility of the received signals from the satellites [1]. A-GPS broadcasts the navigation data of the GPS satellites over the cell phone network. If the data from A-GPS and GPS start to differ enough, the receiver should alert the user about the anomaly.

None of the above methods would work against a sophisticated GPS attack, where the signal timing is only simulated using always the legitimate navigation data and the GPS satellite positions. Detecting a sophisticated attack would require implementation of Doppler shift or signal amplitude changing detection in the GPS receiving chip. Most of the civilian GPS receivers do not do this kind of checking. Therefore, the civilian GPS signals should never be completely trusted.

In one of the studies, the GPS signal jammers were found to be more powerful than they were advertised [6]. This may cause unintentional signal interference; if a citizen acquires a GPS signal jammer to protect their own privacy against GPS tracking devices, it could jam other people's GPS receivers from hundreds of meters away. One could also intentionally jam any GPS receiver from kilometers away with a more powerful civil GPS signal jammer. As there are no many countermeasures against a jamming attack, one should never depend on GPS positioning alone.

7 Conclusion

In this paper we first introduced how the GPS signals work and how the receiver determines its location. We concentrated on the older legacy GPS signals, as they are still used by most of the GPS receivers. We also briefly described the modernized GPS signals.

The main focus of this paper was to present the most common attacks against GPS: jamming and spoofing. We described both of those attacks and defined the requirements and some of the possible real life use cases for both of them. In a conclusion, the GPS signals can be both jammed and spoofed with little effort and with low cost.

In the end of the paper we listed possible countermeasures against jamming and spoofing attacks against GPS receivers. We also discussed why the GPS receivers should not be trusted or relied on, and why the future GPS receivers should be better protected against the spoofing attacks.

References

- [1] Ahmed El-Rabbany. *Introduction to GPS: the global positioning system*. Artech house, 2002.
- [2] Global Positioning Systems Directorate. GPS Interface Specification IS-GPS-200H. Technical report, September 2013. <https://www.gps.gov/technical/icwg/IS-GPS-200H.pdf>.
- [3] Oeystein Glomsvoll. Jamming of GPS GLONASS signals. Master's thesis, University of Nottingham, September 2014.
- [4] Alan Grant, Paul Williams, Nick Ward, and Sally Basker. GPS jamming and the impact on maritime navigation. *The Journal of Navigation*, 62(2):173–187, 2009.
- [5] Sherman Lo, David De Lorenzo, Per Enge, Dennis Akos, and Paul Bradley. Signal authentication: A secure civil GNSS for today. *Inside GNSS*, 4(5):30–39, 2009.
- [6] Ryan H Mitch, Ryan C Dougherty, Mark L Psiaki, Steven P Powell, and Brady W O'Hanlon. Signal characteristics of civil GPS jammers. In *Radiation Navigation Laboratory Conference Proceedings*, 2011.
- [7] Navigation Center, U.S. Department of Home Security. Global Positioning System, Standard Positioning Service: Signal Specification. Technical report, June 1995. <https://www.gps.gov/technical/ps/1995-SPS-signal-specification.pdf>.
- [8] Tyler Nighswander, Brent Ledvina, Jonathan Diamond, Robert Brumley, and David Brumley. GPS software attacks. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 450–461. ACM, 2012.
- [9] Richard Poisel. *Modern Communications Jamming: Principles and Techniques*. Artech House, 2011.
- [10] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. On the requirements for successful GPS spoofing attacks. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 75–86. ACM, 2011.
- [11] Kang Wang, Shuhua Chen, and Aimin Pan. Time and position spoofing with open source projects. *Black Hat Europe*, 148, 2015.
- [12] Jon S Warner and Roger G Johnston. A simple demonstration that the global positioning system (GPS) is vulnerable to spoofing. *Journal of Security Administration*, 25(2):19–27, 2002.
- [13] Jon S Warner and Roger G Johnston. GPS spoofing countermeasures. *Homeland Security Journal*, 25(2):19–27, 2003.
- [14] Hengqing Wen, Peter Yih-Ru Huang, John Dyer, Andy Archinal, and John Fagan. Countermeasures for GPS signal spoofing. In *ION GNSS*, volume 5, pages 13–16, 2005.

IPv6 Honeypots for IoT Devices

Mikko Pohjalainen

mikko.pohjalainen@aalto.fi

Tutor: Amit Tambe

Abstract

IoT devices already make up a large portion of devices which are connected to the internet and this amount can only be expected to rise in the future. Furthermore, security issues are a growing concern with many exploits targeting specifically IoT devices. Honeypots are one possible solution to capture and analyse malware binaries before large scale exploitations occur. IoT honeypots, however, are a special case since usually the attacker expects certain interactivity with them. Afterall, a honeypot should mimic the exact behaviour of a real device and its services and not reveal its true intentions to the attacker. This paper will take a look at existing solutions for this design problem. Additionally, the gradual movement from IPv4 to IPv6 protocol further creates challenges for the design of honeypots, namely the huge address space from which the attacker should be able to find the honeypot. This paper will also examine techniques which can be used to effectively narrow down search spaces and find hosts in the IPv6 network.

KEYWORDS: honeypot, IPv6, IoT

1 Introduction

The number of IoT devices has been rising sharply during the last few years. According to Statista the number of IoT devices in use by the end of 2018 already exceeded 10 billion. Moreover, the amount of these devices is estimated to roughly double by the end of 2020 [1].

The increase in the number of IoT devices also raises concerns about their security. There have been a number of attacks in the last few years exploiting poor security in devices such as IP cameras, baby monitors and printers. For example, at its peak in late 2016 the Mirai botnet had infected around 600,000 devices. This botnet was used to conduct DDoS attacks against DNS provider Dyn and cloud service provider OVH [2]. To prevent attacks towards IoT devices, it is essential to study how attackers exploit these devices. One such method is called a honeypot. Honeypots are intentionally exposed computer systems which aim to lure attackers and examine their actions. Furthermore, honeypots are sometimes used to distract attackers from the real targets by faking important data. Honeypots have been around for a long time already but they have not yet been used extensively in the context of IoT devices.

Naturally, one critical property of a honeypot is that it must not seem like a one. After all, attackers should think they are compromising real computer systems, or else it is unlikely they will use any potential *zero-day exploits*. Currently, honeypots are usually implemented by well-established frameworks such as Honeyd [3] and they work well while disguising as servers or web services. IoT devices, however, are fundamentally different, since they often have some direct link to the physical world (e.g., IP cameras can move and display live video over the internet) [4]. Consequently, it is often necessary to use physical devices or accurate emulation when constructing IoT honeypots.

A Further complicating matter is the gradual movement from the IPv4 to the IPv6 address space [5]. With the expected increase in the number of IoT devices, the remaining IPv4 addresses will likely be exhausted and future IoT devices will be compelled to use IPv6. A crucial problem with IPv6 and the use of honeypots is the vast amount of possible addresses. While IPv4 has a pool of 2^{32} addresses, IPv6 has a much larger pool of 2^{128} addresses. Using a popular network scanning tool such as Masscan, the whole IPv4 address range is scannable within few minutes. The IPv6 address range, on the other hand, is practically impossible to scan by brute

force [6]. This makes it extremely unlikely that an attacker finds a honeypot hosted in the IPv6 network by random.

There are already some proposed solutions to solve problems with IoT honeypots in general as well as problems regarding IPv6 honeypots. This article aims to survey the current state of proposed solutions as well as discuss benefits and drawbacks related to each of these.

2 Background

The concept of honeypots started appearing in books and whitepapers around 1990, but it is likely they were being researched and developed before that. The first publicly available honeypot framework was called "Deception Toolkit", in short, DTK. Its first release was in 1997. DTK simply emulated some known UNIX vulnerabilities and logged the attacker's actions. DTK was rapidly followed by increasingly complex honeypot solutions which could emulate whole networks of computers [7].

In 1999, a group of security researchers founded a group called the HoneyNet Project. This was a non-profit organization which was dedicated to research and sharing findings on network attacks. The HoneyNet Project developed and utilized an advanced honeypot, which was called a HoneyNet. The findings and publications by the HoneyNet Project helped raise general awareness of honeypots in the network security field [8]. In the years 2000 and 2001 honeypots were already widely used to capture and analyse network worms which were an increasing problem then [7].

In the last few decades, the increasing sophistication in honeypots and tools used to expose them has been an ongoing arms race. Originally, most honeypots were low interaction honeypots which only emulated certain services on a computer. However, the increasing power in computing and the adoption of virtualization technologies has enabled a cost-effective way to host multiple high interaction honeypots on one physical machine. High interaction honeypots appear to be full computer systems which are more difficult to identify as honeypots. Many specialized honeypots have also been created for more specific areas, such as social media, spam detection [9] and client-side vulnerabilities [10]. Since attackers generally want to avoid honeypots, it has even been proposed that valuable computing resources would try to disguise themselves as honeypots [11].

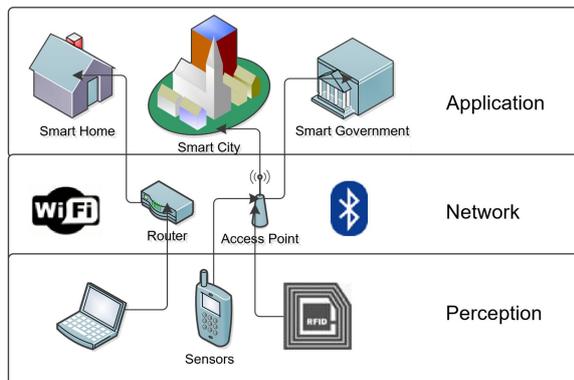


Figure 1. The basic layers of the IOT architecture [14].

3 Internet of Things and IPv6

3.1 Architecture and Security Threats of IoT

The internet of things is a concept of everyday physical objects communicating with each other over a network. According to an article by Gubbi et al., the internet of things is characterized by objects sensing their environment and interacting with the physical world, while utilizing current internet standards [12]. There are many applications for IoT ranging from common items at home to enterprise level machinery and sensors. Due to the high level of interconnectivity, IoT brings with it a lot of conveniences as well as multiple security and privacy concerns.

The basic architecture of IoT consists of three main layers: *Perception*, *Network* and *Application layer* (Fig. 1). The perception layer consists of sensors which collect data from the surrounding environment. The *network layer* is responsible for relaying the collected data to be processed (e.g., to cloud servers). Finally, the application layer is the part interacting with the user. Each of these layers is susceptible to a multitude of different attacks. For example, the *perception layer* is prone to spoofing information (e.g., replay attacks); the network layer can be compromised by a man in the middle attack; the application layer can be attacked with malicious code injections [13].

Since IoT features so many attack vectors, it becomes very difficult to ensure security. A survey of the current devices on the market conducted by Wurm et al. revealed that many of the current consumer and industrial devices have serious vulnerabilities [14]. Honeypots might be one solution for finding many of these vulnerabilities early on. Pa et al. conducted an

IoT honeypot experiment for 39 days and discovered multiple new malware binaries previously unknown to the VirusTotal database [15]. Like most other honeypots, this was public on the IPv4 network and nearly 17 000 unique ip's established a connection to it. This result really shows how easy it was for attackers to find the honeypot by just brute force scanning the whole IPv4 network. As the future of IoT relies undoubtedly on the IPv6 network, it is important to evaluate other methods for finding hosts. In the next section, IPv6 protocol, as well as methods for finding hosts, are discussed in detail.

3.2 General Information on IPv6

The IPv6 specification document, written by Deering et al., defines IPv6 as the direct successor of IPv4. The IPv6 protocol includes multiple enhancements over IPv4, which makes it better suitable for IoT. It features many new addressing capabilities, such as a larger pool of addresses, as well as a simpler way of address auto-configuration. Furthermore, it improves multicast and adds a new *anycast address*, which is used to send packets to any node in a group. It also makes simplifications in the header format as well as increases routing efficiency [16].

The basic structure of an IPv6 address is the following: The first 64 bits of the address is the network part of the address and the last 64 bits is the host part of the address. The network part of the address is issued by a local authority such as an ISP . According to the IPv6 Address Allocation and Assignment Policy document by RIPE defines that ISPs are initially assigned address spaces from /32 to /29. ISPs are required to assign address spaces ranging from /64 to /48 to end sites [17]. The host part of the address can be assigned either manually or configured automatically by one of two methods which will be discussed in the next section.

IPv6 has already gathered significant momentum in adoption. According to Google statistics, the percentage of Google users accessing their services over IPv6 is currently around 25% [18]. Furthermore, the constantly growing number of smart devices can be expected to further increase adoption in the near future.

3.3 Scanning the IPv6 Network

As mentioned before, the IPv6 network consists of a total of 2^{128} unique addresses. Since the effectiveness of honeypots relies heavily in the fact whether they are findable by attackers, it is important to understand how this vast pool of addresses can be narrowed down to a more manageable size. When narrowing down the pool of addresses to scan, the first thing to do is only scan networks which have been assigned. Since the network prefixes issued by ISPs are likely to be mostly public or reasonably easily acquirable information, the pool of addresses can be in many cases narrowed down to 2^{64} per network. However, the remaining address pool is still way too large to scan. In order to further reduce the addresses to be scanned the construction of the host part of the address needs to be understood.

In IPv6 there are two ways of auto-configuring the host part of the address in a network: *Stateless Address Autoconfiguration* (SLAAC) and *Dynamic Host Configuration Protocol for IPv6* (DHCPv6). In addition to these automatic methods it is also possible to manually configure addresses. In their techreport "Network Reconnaissance in IPv6 Networks" Gont et al. discuss their findings on multiple ways on how to narrow down search spaces when trying to find target hosts in an IPv6 network in each of these cases [19].

In SLAAC a host joining the network requests the network configuration from a router using ICMPv6 messages. Among other information about the network, this message unveils the network prefix to the host. The final address is constructed from the received network prefix and appending an interface identifier (IID), which is generated locally on the host. The IID is generated by splitting the MAC address of the network device from where the *Organizationally Unique Identifier* (the OUI is the first 24 bits of a MAC address) ends and adding the constant word 0xfffe in the middle. After this the 7th bit from left to right is inverted. For example, the MAC address a2:e2:cf:e4:37:24 would result into an IID of a0e2:cfff:fee4:3724. Since also the OUIs are public information and the constant (16-bit) word is known, the addresses to be scanned in this case would be narrowed down to $N * 2^{24}$ addresses (N being the amount of OUIs of interest). This is already an easily scannable pool of addresses [19].

The addresses to be scanned can also be often narrowed down in case DHCPv6 is used to assign addresses. Similarly to DHCP in IPv4 net-

works, DHCPv6 assigns addresses to hosts joining the network according to a predefined rule. This often results in very predictable results (such as incrementing addresses) which translate to a pool of 2^8 to 2^{16} addresses. It is worth noting that there exist methods of assigning completely random addresses when using DHCPv6, which would eliminate this predictability [19].

The final way to configure addresses is to manually configure them. This would probably be the most likely case with servers since changing addresses would mostly cause problems. According to Gont et al. system administrators mostly use four patterns while manually configuring addresses:

1. Using low-byte addresses. These are addresses which identify a host with a single number in the least significant bits (e.g., an address ending in `::1`)
2. IPv4 based addresses. These addresses would encode the IPv4 address of the system in the last bits of the IPv6 address (e.g., an address ending `::192.0.2.1`)
3. Service port addresses. These addresses encode the port number of the service running on the host to the last bits of the IPv6 address (e.g., an address ending in `::80`)
4. Wordy addresses. These addresses would encode some words into the last bits of the IPv6 address (e.g., an address ending in `::cafe`).

In these cases the addresses which would need to be scanned could be reduced to anywhere from 2^8 to 2^{24} [19].

4 Existing IoT and IPv6 Honeypot Solutions

While examining current literature there didn't seem to exist honeypots aimed at specifically IoT devices on the IPv6 network. However, several honeypot designs already exist which specialize in some aspects of either IoT or IPv6. There are several important factors when designing these honeypots. In IPv6 it is necessary to cover large address spaces and redirect attackers. It is also beneficial to be able to scale and provide as many

targets as possible. Finally, It is crucial to seem like an authentic service running on a real device. In the following sections several honeypot designs, which aim to solve these design problems, will be introduced.

4.1 IoT POT

Pa et al. created a honeypot design called IoT POT, which emulates telnet services on various IoT devices on the IPv4 network. The IoT POT architecture consists of a low-interaction layer as well as a high-interaction layer. The low-interaction layer immediately responds to known commands and forwards unknown command to be evaluated to the high-interaction layer. Furthermore, the low-interaction layer caches data in device profiles which contain information on authentication, banners and command interactions. The high-interaction layer, which is called IoT BOX emulates unknown commands on various cpu architectures and returns the results to the low-interaction layer to be cached [15].

When attempting to compromise IoT POT, the attacker is greeted with a realistic welcome message which is fetched from device profile. These welcome messages are gathered from the internet by telnet scans. IoT POT supports multiple login modes. It can be configured to: allow any login credentials; accept only certain credentials; reject a number of login attempts and eventually accept. After the attacker has managed to login, the low-interaction layer responds to all known commands issued by the attacker based on existing device profiles [15].

When the attacker executes unknown commands or malware binaries the low-interaction layer interacts with IoT BOX in order to process these actions. While being processed, the commands and malware binaries are analysed on multiple different hardware/os combinations. IoT BOX supports MIPS, MIPS64, PPC, SPARC, ARM, MIPS64, sh4 and X86 cpu architectures, which are emulated by the open source cpu emulator QEMU. An OpenWrt platform was created to run on each of these emulated cpus creating the virtual machines used for emulation [15].

IoT BOX was designed to run on a single physical host machine and thus a virtual bridge network was created in order for the emulated environments to be able to connect via tap interfaces. An access controller was implemented with iptables to manage traffic inputs and outputs of the virtual machines. For every 24 hours of operation, IoT BOX compiles an analysis report containing the results of pcap analysis. This analysis contains timestamps, data rates, packet sizes and victim ips [15].

During the operation period, IoTPOt was able to capture multiple new attack binaries and the team was able to identify distinct intrusion, infection and monetization strategies for these [15]. However, since IoTPOt is implemented through emulation and focuses mainly on the telnet protocol, it is not suitable for a general purpose IoT honeypot. In the next section, a solution which relies on physical devices will be discussed.

4.2 SIPHON

Guarnizo et al. created a honeypot solution incorporating physical IoT devices while attempting to maintain scalability. This was achieved by having cloud virtual machine instances with public IPs tunnel traffic to the physical devices via a forwarder machine using SSH tunneling. Since all traffic goes through the forwarder machine it could be used as a "man in the middle" to capture all traffic with tcpdump and send it to a separate storage and analysis machine. The physical devices themselves were isolated into their own network by using 802.1Q VLANs. This ensured that no unwanted compromises occur [4].

Since the public virtual machines can forward the traffic to same devices the devices visible to the attacker are the maximum number of virtual machine instances multiplied by the number of physical devices. For example, with 1000 virtual machines and 100 physical devices, 100000 services would be visible to the attacker. Naturally, one limiting factor is the ability of the physical devices to handle requests [4].

During the experiment with SIPHON, the team was able to inspect 700MB of daily traffic for two months. Furthermore, the honeypot detection algorithm of Shodan gave SIPHON a very low "honeyscore", suggesting that the honeypot seemed to be an authentic IoT device. SIPHON does, however, require a fairly large number of physical devices and space to set up these devices in order to implement on a large scale. Furthermore, if multiple connections are allowed from different VMs to the same physical device, the honeypot might be revealed [4].

4.3 Honeydv6

The Honeydv6 proposed by Schindler et al. is a honeypot, which was designed specifically for the IPv6 network. Honeydv6 is based on its IPv4 counterpart Honeyd. The key feature of Honeydv6 is that it can be used to simulate huge address spaces on the IPv6 network. This is achieved by

using a method where simulated hosts can be created on demand. Furthermore, IPv6 connections are accepted by random so attackers scanning the network will find a host only after a while of scanning. When a new request arrives, Honeydv6 accepts or rejects it according to a configurable property. If the connection was rejected, the requested address will be blacklisted so further scans won't receive different results. If the connection is accepted, the honeypot responds to requests according to system templates which define information such as open ports and scripts which are assigned to them. Regardless of whether the connection was accepted or not, it is logged so scan patterns can be studied [20].

The Honeydv6 framework can simulate entire networks with virtual routers and low-interaction hosts. In order to make network packet routing seem authentic, honeydv6 implements ICMPv6 packet types *Time Exceeded* as well as *Destination Unreachable*. This ensures that scans via *traceroute6* produce believable outputs. In order to make the honeypot available for the global IPv6 network, the covered prefixes need to be forwarded to the physical host running Honeydv6. In their research, Schindler et al. utilized a /48 network provided by the tunnel broker Hurricane Electric [20].

Since Honeydv6 is a low-interaction honeypot it is only capable of responding according to scripts which are set beforehand. This means that it would likely be fairly easy to identify as a honeypot, especially if it were used to simulate IoT devices. Honeydv6 has been later extended to incorporate high-interaction honeypots. This solution will be introduced in the next section.

4.4 Hyhoneydv6

The Hyhoneydv6 created by Schindler et al. extends the functionality of Honeydv6 by introducing high-interaction honeypots. The high-interaction honeypots are dynamically deployed when the attacker tries to interact with it in ways which are not defined in a system template. The Hyhoneydv6 honeypot architecture is split into two layers: the first layer is the low-interaction layer which responds to simple ping requests; the second layer is the high-interaction layer which handles more sophisticated attacker behaviour (Fig. 2). When a high-interaction honeypot is needed, Honeydv6 creates one dynamically and proxies the traffic to it via the low-interaction honeypot [6].

Hyhoneydv6 utilizes a pool of running virtual machine instances, which

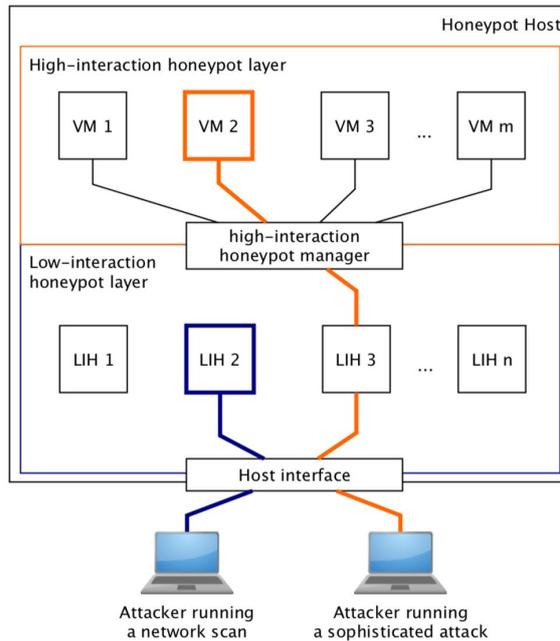


Figure 2. The Hyhoneydv6 honeypot architecture: The low-interaction layer handles responding to network scans and sophisticated attacks are forwarded to the high-interaction layer[6].

are booted already during initialization of the honeypot. This is crucial in order to keep latencies reasonable. Furthermore, the architecture of Hyhoneydv6 allows it to assign the high-interaction honeypots the same IP addresses, which are already assigned to their low-interaction counterparts. While the high-interaction honeypot is still being configured, the low interaction-honeypot buffers all received packets. Once the configuration is complete, the low-interaction honeypot sends the buffered packets to the high-interaction honeypot [6].

Using this combined setup of low and high-interaction honeypots yielded the following results: Initial ssh connections from the attacker to the high-interaction honeypots were established on average in 2.5 seconds. Subsequent commands after an established connection took less than 0.5 seconds. When compared to typical KVM based virtual machine the corresponding latencies were 1.5 and 0.02 seconds respectively. When testing the data throughput with the servload benchmark, the high-interaction honeypot managed to process 200 requests per second. The KVM based virtual machine managed 320 requests per second [6].

Hyhoneydv6 is a complete hybrid honeypot for the IPv6 network. It maintains the great scalability of its low-interaction counterpart while extending functionality. However, the added latency while configuring the high-interaction honeypot might help reveal it to the attacker.

5 Conclusion

For a long time already, honeypots have been a great method of capturing new malware binaries by deceiving attackers in compromising fake targets, while spying on them. Traditionally, honeypots have been mainly used to capture malware and worms targeting certain services on servers. In context of IoT and IPv6, honeypots have not yet been extensively utilized.

At first, it would seem that honeypots on the IPv6 network would be nearly impossible to find. The IPv6 address space is huge and it would be easy to hide hosts in the 64 bits, which is allocated to the host part of the address. However, current findings suggest that the most common cases are addresses assigned by predictable patterns. Attackers are sure to make use of all available information, which can be used to narrow down scans. For example, by using already exposed network prefixes from ISPs as well as public information about OUIs, in many cases scans can be narrowed down to address spaces which range between 2^8 to 2^{24} . Automated network scanning tools and websites such as Shodan are likely to implement these filtering criteria and evolve to be more efficient in discovering hosts in the IPv6 network.

IoT and IPv6 both introduce various challenges to honeypots. While not without drawbacks, there already exist well-working solutions to all the major problems. Honeydv6 has successfully eliminated problems with the huge address spaces of IPv6 by emulating whole /48 networks and providing realistic responses to scans and some commands carried out by attackers. Furthermore, Hyhoneydv6 extends this logic by introducing high-interaction honeypots, which are dynamically deployed to analyse more sophisticated behaviour by attackers. IoTPOt has proved to be successful in capturing and analysing malware binaries targeted at IoT devices by utilizing emulation of different cpu architectures within IoTBOX. SIPHON, on the other hand, could be used to monitor and analyse attackers compromising physical IoT devices, effectively creating honeypots which are very hard to identify as such.

Current literature does not suggest that a honeypot which exposes physical or simulated IoT devices on the IPv6 network exists. However, based on current research, such a honeypot should be a plausible way of assessing threats to IoT devices on the IPv6 network.

References

- [1] “Iot device installed base worldwide 2009-2020.” <https://www.statista.com/statistics/764026/number-of-iot-devices-in-use-worldwide>. Accessed: 2019-01-31.
- [2] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, *et al.*, “Understanding the mirai botnet,” in *USENIX Security Symposium*, pp. 1092–1110, 2017.
- [3] N. Provos *et al.*, “A virtual honeypot framework,” in *USENIX Security Symposium*, vol. 173, pp. 1–14, 2004.
- [4] J. D. Guarnizo, A. Tambe, S. S. Bhunia, M. Ochoa, N. O. Tippenhauer, A. Shabtai, and Y. Elovici, “Siphon: Towards scalable high-interaction physical honeypots,” in *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*, pp. 57–68, ACM, 2017.
- [5] P. Richter, M. Allman, R. Bush, and V. Paxson, “A primer on ipv4 scarcity,” *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 2, pp. 21–31, 2015.
- [6] S. Schindler, B. Schnor, and T. Scheffler, “Hyhoneypot: A hybrid honeypot architecture for ipv6 networks,” *International Journal of Intelligent Computing Research*, vol. 6, 2015.
- [7] L. Spitzner, *Honeypots: tracking hackers*, vol. 1. Addison-Wesley Boston, 2002.
- [8] “About the honeynet project.” <https://www.honeynet.org/about>. Accessed: 2019-02-26.
- [9] K. Lee, J. Caverlee, and S. Webb, “The social honeypot project: protecting online communities from spammers,” in *Proceedings of the 19th international conference on World wide web*, pp. 1139–1140, ACM, 2010.
- [10] Y. Aloseter and O. Rana, “Honeyware: a web-based low interaction client honeypot,” in *2010 Third International Conference on Software Testing, Verification, and Validation Workshops*, pp. 410–417, IEEE, 2010.
- [11] N. C. Rowe, E. J. Custy, and B. T. Duong, “Defending cyberspace with fake honeypots,” 2007.
- [12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (iot): A vision, architectural elements, and future directions,” *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.
- [13] M. U. Farooq, M. Waseem, A. Khairi, and S. Mazhar, “A critical analysis on the security concerns of internet of things (iot),” *International Journal of Computer Applications*, vol. 111, no. 7, 2015.
- [14] J. Wurm, K. Hoang, O. Arias, A.-R. Sadeghi, and Y. Jin, “Security analysis on consumer and industrial iot devices,” in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 519–524, IEEE, 2016.
- [15] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, “Iotpot: analysing the rise of iot compromises,” in *9th {USENIX} Workshop on Offensive Technologies ({WOOT} 15)*, 2015.

- [16] S. Deering and R. Hinden, "Internet protocol, version 6 (ipv6) specification," tech. rep., 2017.
- [17] "Ipv6 address allocation and assignment policy." <https://www.ripe.net/publications/docs/ripe-707>. Accessed: 2019-04-04.
- [18] "Ipv6 adoption." <https://www.google.com/intl/en/ipv6/statistics.html>. Accessed: 2019-02-28.
- [19] F. Gont and T. Chown, "Network reconnaissance in ipv6 networks," tech. rep., 2016.
- [20] S. Schindler, B. Schnor, S. Kiertscher, T. Scheffler, and E. Zack, "Ipv6 network attack detection with honeydv6," in *International Conference on E-Business and Telecommunications*, pp. 252–269, Springer, 2013.

Digital Twins for Industrial Edge 4.0: Concepts and Tools

Adika Bintang Sulaeman
adika.sulaeman@aalto.fi

Tutor: Prof. Hong-Linh Truong

Abstract

Industry 4.0 is expected to be the next big phase in industries. One of the challenges in Industry 4.0 is monitoring and controlling industrial machines in real-time within a virtual platform. Digital Twin (DT), defined as a digital representation of a physical object, supports Industry 4.0 for controlling, monitoring, and providing data for analysis in manufacturing. A literature study has been conducted to review DT in manufacturing. This seminar paper is intended as a review paper for DT developers and gives discussions on an overview of DT, key technologies to implement DT, DT frameworks, DT tools and how they can be combined to build DT systems. From the literature study conducted, DT frameworks and tools rely much on embedded systems, Internet of Things (IoT), cloud services and physical modeling software. Furthermore, this seminar also discusses about the combinations of the existing frameworks and tools to create new frameworks according to the application requirements.

KEYWORDS: *Digital Twins, Industry 4.0, Digital Twin Frameworks, Digital Twin Tools*

1 Introduction

Industry 4.0 is the next big phase in industry. With Industry 4.0, it is possible to gather real-time data from the machines that run in industry and process the data into something meaningful and useful. Industry 4.0 mainly consists of three supporting technologies: IoT, Cyber-Physical Systems (CPS), and Smart Factories [5]. The combination of these technologies builds interconnected devices forming Digital Twins (DT).

A DT models a physical object by creating a digital representation using real-time data [9]. The data is gathered throughout its life-cycle and used as the source to monitor, learn from and enhance decision making. A DT enables engineers to monitor and understand how the machines behave once it is released and run by users. Furthermore, engineers can analyze the data and predict the future performance of the machines.

There are some use cases of DT for Industry 4.0. Consider a Printed Circuit Board (PCB) printer for electronic manufacturers. The PCB printer must be very precise, because the smallest error by the laser cutter may lead to PCB flaws. A DT enables engineers and technicians to monitor and analyze the data to predict the time when the spare parts wear out. Another example would be monitoring the jet engine of airplanes. By analyzing data gathered in real-time, engineers and technicians may predict failures in jet systems, which will lead to the reduction of airplane incidents. Furthermore, DT may give feedback to the engineers who design the machine to help them realize an agile development system.

The main value that the DT delivers is an understanding of product performance [9]. By understanding performance, manufacturers may detect and understand faults better, create an effective maintenance schedule, troubleshoot machines remotely, and decide appropriate add-on services.

The DT has some challenges in its development and implementation. In [1], the author has stated a number of challenges such as data consistency between the real physical assets and the digital representation, as well as connectivity and security concerns of cloud computing for DT. Software architectural aspects such as internal structure, APIs, integration and runtime environment are also critical challenges for DT [8].

1.1 Scope and Goals

This paper aims to review the concept of DT for manufacturing in Industry 4.0 as well as technologies to build the DT system. This seminar paper

is intended as a review paper for DT developers to build DT systems.

1.2 Structure

The rest of this paper is organized as follows. Section 2 discusses the key technologies for DT. Section 3 discusses the existing DT framework approaches and tools to build the DT systems. Section 4 concludes this review paper and discusses future work.

2 Principles of Building Digital Twin Systems

2.1 Requirement Analysis of Digital Twin Systems

There are four requirements for building DT systems [13]. These requirements can be used to determine key technologies for building DT systems.

From the aspect of DT applications, the first requirement is DT systems must be general enough to support various applications such as manufacturing, construction, and health-care.

From the technologies aspect, since the main aim of the DT is to fuse the interaction in CPS, the DT must embrace the integration of new generation information technologies such as IoT, cloud computing, big data, and artificial intelligence (AI) to support its goal.

From the modeling object aspect, the DT must integrate and fuse operational data from physical space and simulated data from virtual space. The DT must also be encapsulated as services which have user-friendly interface and ease the user operations.

From the modeling method aspect, the DT requires high-fidelity virtual modeling. The virtual modeling includes geometrical, physical, behavior, and rules modeling.

2.2 Key Technologies of Digital Twins

The requirements analysis leads to the abstraction of the DT concept model. Fig. 1 shows the DT concept model which is divided into five dimensions, i.e., the Physical Entity (PE), Virtual Entity (VE), Services (Ss), DT Data (DD) and Connection (CN). Each of these components has its own key technologies to build the DT system as a whole as shown in Fig. 2 [13].

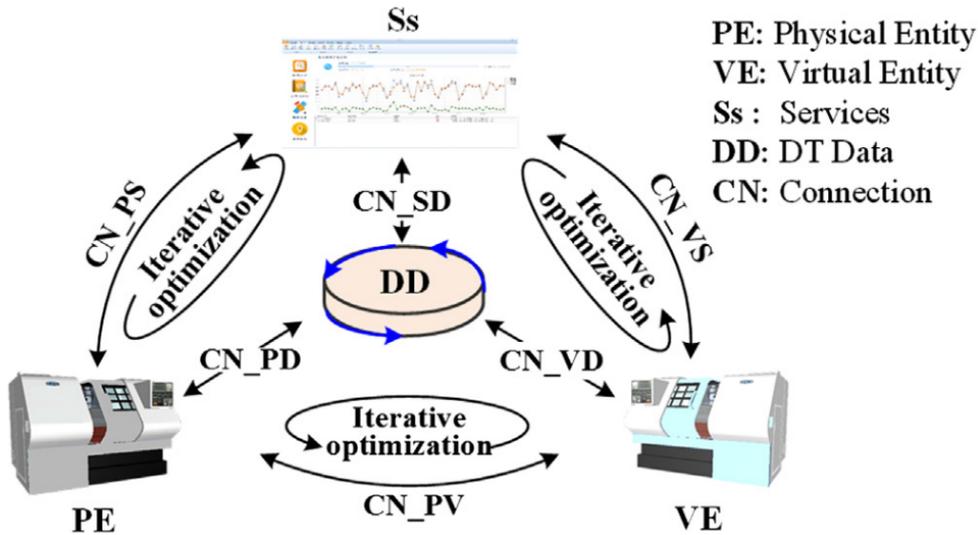


Figure 1. DT concept model [13]

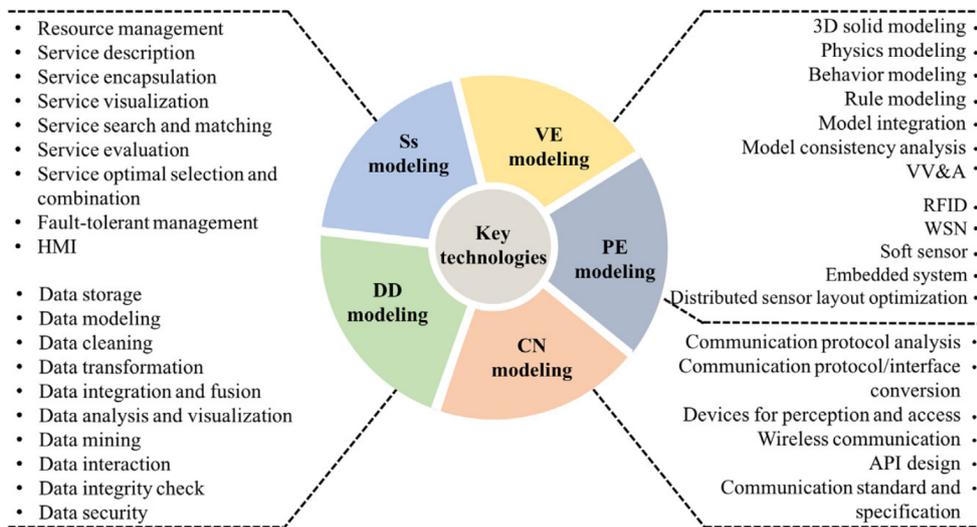


Figure 2. Key technologies for DT [13]

The PE is the real physical entity in the physical space. It contains the real machines with their sensors and actuators. The technologies in the PE includes embedded systems to embed computing capabilities to the entities, RFID to track and identify the entities, Wireless Sensor Network (WSN) to transmit sensors' data and distributed sensor layout optimization to reduce information redundancy.

The VE consists of a set of models representing the PE. Modeling techniques such as three-dimension solid modeling, physics modeling, behavior modeling, and rule modeling are needed to create useful modeling of the PE. To keep PE and VE model consistent, model consistency analysis, verification, validation and accreditation must be implemented.

The Ss of the DT serves other DT dimensions' needs. The services used by PE include monitoring service, state prediction service, and energy consumption optimization service. VE uses the construction service, calibration service and test service for modeling. Some aspects are needed to create and maintain Ss, i.e., resource management, service description, service encapsulation, service visualization, service search and matching, service evaluation, service optimal selection and combination, fault-tolerant management and human-machine interface (HMI).

The DD contains data from physical and virtual space. Therefore, the number of heterogeneous data is high. Technologies for processing and securing data, such as data storage, data modeling, data transformation, data cleaning, data analysis, data mining, data integrity check and data security, are required in helping modeling DT data.

The CN connects different elements of DT. It enables PE, VE, Ss, and DD to communicate with each other. Key technologies of CN includes communication protocol analysis, communication protocol/interface conversion, wireless communication, Application Programming Interface (API) design, as well as communication standard and specification.

3 Implementing Digital Twin Systems

This section starts by discussing the existing DT frameworks, which give guideline or supporting structures for building DT systems. Then, it continues to discuss tools to realize DT framework. Finally, the discussion about the combinations of various DT frameworks and tools is presented.

3.1 Digital Twin Frameworks

Some researchers have proposed frameworks for building DT. While some of them propose holistic frameworks, there are also frameworks focusing on one particular dimension of DT. Table 2 shows a brief summary of the discussed frameworks.

Two examples of holistic frameworks for building DT were proposed by Yu Zheng et al. [15] and Zhuang et al. [16]. The framework proposed by Yu Zheng et al. aims to build a mapping of physical objects, modeling life-cycle of products in a dynamic way and optimizing real-time process. The framework proposed by Zhuang et al. aims to build DT-based smart production management and control framework for product assembly shop-

Framework	Focus	Future works (not implemented yet)
Zheng et al. [15]	Application framework for DT in manufacturing	Data synchronization; mapping methods between physical and digital space; application mode of DT
Zhuang et al. [16]	Framework of DT-based smart production management and control for assembly shop-floor	Construction and optimization of IoT networks in a physical assembly shop floor; construction, optimization, and running of an assembly shop-floor DT; construction of big data management platform; modeling and implementation of smart decision making algorithm
Zhang et al. [14]	DT modeling based on perception data modeled as ontology	Dynamic linkage between physical, digital, 3D; fully functional service system; AR for 3D models
Qi et al. [10]	Combining edge, fog, and cloud to build DT system	Increase the number of and the capabilities of edge nodes; architecture, platform, and standard of fog computing; data filtering to choose which data to process in edge, fog, and cloud; energy consumption; security
Lynn et al. [7]	Modeling and controlling physical objects through DT	Automated plan generation; enhanced trajectory planning
Botkina et al. [2]	Modeling and controlling physical objects through DT	Integrating non-tweeting machines to the system
Ciavotta et al. [3]	Software service for DT based on microservice	IoT protocols such as MQTT for CPS communication; AutomationML for data exchange

Table 1. Existing DT frameworks

floor.

Although both frameworks were proposed to build DT for manufacturing, they had different approaches. The framework proposed by Yu Zheng et al. divided the system into three dimensions, i.e., physical space, virtual space, and information-processing layer. On the other hand, the framework proposed by Zhuang et al. divided the system into four dimensions, i.e., management of the physical space, construction of the virtual space of the shop-floor, DT and big-data driven prediction and production management and control service of the assembly shop-floor.

Zhang et al. proposed another framework focusing on modeling DT workshop based on perception data [14]. The framework consists of three parts, i.e., physical model, ontology-based digital model, and the virtual model. The ontology-based data modeling is what makes this framework different from the others.

The aforementioned frameworks are the examples of the complete DT framework, starting from the physical to the virtual entity. However, they do not thoroughly describe the frameworks for each of the specific DT dimensions.

The service dimension of DT mentioned in section 2 is crucial for DT system because it connects one dimension to the others. There are many ways to build software services.

Ciavotta et al. suggested the use of microservice architecture for building middleware as a service for DT [3]. This middleware supports data collecting, authentication, and accessing DT resource and data. The benefits of using microservice architecture include agility, isolation, and resilience as well as elasticity. However, it also increases the system complexity at the same time.

The PE dimension of DT may come with different messaging formats and protocols. Software service middleware must support a variety of data formats coming from different dimensions of DT. For example, Schroeder et al. proposed a framework for building middleware to accept AutomationML data format and convert them into other data format via REST API [12].

Several frameworks for virtual entity have also been proposed. Schroeder et al. proposed the use of Augmented Reality (AR) to visualize DT [11], leveraging Vuforia Engine to build the AR system. For controlling physical devices through DT in VE, Lynn et al. [7] and Botkina et al. [2] had slightly different approaches. The framework proposed by Lynn et al.

used raw TCP for controlling the device and raw UDP for getting the feedback data from the device, whereas the Botkina et al. approach used the standard Line Information System Architecture (LISA) and ISO 13399.

So far, all the frameworks discussed suggest the use of cloud computing for their processing and storage. However, some applications are latency sensitive. Bringing data to the cloud which can be physically far from the source of data may add significant latency overhead.

Qi Zhang et al. proposed a DT framework based on edge computing, fog computing and cloud computing to improve efficiency and reduce latency [10]. In many cases, the data processing and control of physical objects require very low latency. Therefore, edge computing architecture resides on the unit level to achieve that goal. The unit level entities are connected to the information management systems such as Enterprise Resource Planning (ERP) and Manufacturing Execution System (MES). Fog computing is suitable for connecting CPS to DT to improve efficiency. Finally, the data can also be stored and processed in the cloud at the SoS level for various purposes such as data analysis and long-term storage.

3.2 Tools for Digital Twins

In this subsection, tools or sets of software which equip the DT frameworks are discussed.

The PE needs tools to control, sample the operation and environment data, as well as send them to other dimensions of DT. MachineKit is an open-source platform for controlling machines, enabling users to control servo motors of Computer Numerical Control (CNC) machine based on the commands sent by VE [7].

The CN dimension, which connects PE to other dimensions of DT, represents the IoT in general. While there are many protocols and communication mechanisms for IoT, not all of them fits into industrial applications. OPC Unified Architecture (OPC UA) is a machine-to-machine communication protocol that is used by the framework proposed by Yu Zheng et al. [15]. MTConnect is another mechanism to retrieve data from machine tools. Coronado et al. used MTConnect to retrieve the data from the physical space to their DT system [4].

MES software is commonly used for both collecting data and monitor process and operation of the manufacturing process. MES is generally proprietary and expensive. As a result, small manufacturing enterprise may have difficulties in deploying MES in their system.

Coronado et al. came up with a low-cost MES system based on Android [4]. This tool captures part and tooling information using an Android-based MES. The data from MES is combined with data from device sensors to represent parts, operators, capital equipment and consumable in shop-floor DT. This tool supports the Ss dimension of DT as an HMI and resource management tool.

There are various supporting tools for building the DT services, depending on the system architecture of the service. FIWARE is a middleware for exchanging data that can be used as a service for DT [12]. MAYA Support Infrastructure middleware used Netflix Eureka for service discovery and Elasticsearch-Logstash-Kibana (ELK) for log monitoring [3]. The recent trend in microservice technology is to scale the service horizontally; leveraging containers to deploy the services. For this case, containers orchestration tools such as Docker Swarm and Kubernetes may support the system. However, the containerization is not a silver bullet solution since it may impact the network performance in transfer rate and packet loss negatively [6].

The data from both physical space and virtual space must be stored in a database system. Depending on how the data is structured, several options for database platforms are available. For structured data, ODBC-compliant database such as MySQL can be used to store the data [14]. For unstructured data, NoSQL platform such as Apache Cassandra can be used [3].

There are some tools, open source or proprietary, available to support the DT framework in VE dimension. To control machine tools from the VE, Computer-Aided Manufacturing (CAM) software such as SculptPrint can be used [7]. Vuforia Engine can be used to leverage AR technology in building the VE [11]. To build 3D models of the physical objects, SolidWorks, Pro/E and CATIA can be used [16]. FlexSim can be used as a software simulation for manufacturing process [14]. Table 2 summarizes the tools for building DT systems.

3.3 Combining Appropriate Frameworks and Tools

Several DT frameworks have been discussed to analyze the use cases and limitations. From the limitations and future works of those frameworks, a more complete framework can be built. In principle, the new framework can be formed by combining existing frameworks and adjusted to the application requirements.

Tools	Domain	Roles
MachineKit	PE	Platform for machine control applications
MTCConnect	CN	Manufacturing standard to retrieve information from machines
OPC UA	CN	Machine to machine communication for industrial automation
FIWARE	Ss	Middleware for data exchange
Netflix Eureka	Ss	Service discovery in microservice architecture
Docker/containerd and Docker swarm/Kubernetes	Ss	Containerization and orchestration of deployed services
Relational data storage (MySQL, PostgreSQL)	DD	SQL database
NoSQL data storage (Apache Cassandra)	DD	NoSQL database
FlexSim	VE	Simulation, modeling, visualization and prediction software for manufacturing systems
SculptPrint	VE	Software for generating tool paths for 5-axis machine tools
CATIA	VE	3D physical modeling
SolidWorks	VE	
Pro/E	VE	
Android-based MES [4]	Ss and VE	Low cost and open source based Manufacturing Execution System (MES)

Table 2. Tools for building DT systems

For controlling machines in physical space, the framework proposed by Botkina et al. [2] must have tweeting machines in the shop-floor. While this tweeting machines development is not discussed, Lynn et al. [7] describe the mechanism of sending feedback data from actuators with raw UDP. The use of raw UDP by Lynn et al. might not be applicable if it does not comply with existing standards. This mechanism can be combined with the framework proposed by Yu Zheng et al. [15] to use standard protocol such as OPC UA.

The services in DT is described by the framework proposed by Ciavotta et al. [3]. This service is supposed to be deployed in the cloud. However, it does not cover the latency sensitive applications. The combination of the framework proposed by Ciavotta et al. and Qi et al. [10] might be the solution for case sensitive applications since it offers the use of edge, fog, and cloud computing to deploy the services. Furthermore, the use of containers and orchestration in fog, edge, and cloud computing might be used to ease the deployment, maintenance and horizontal scaling of the services.

For modeling the physical space in the virtual space, framework proposed by Zhuang et al. [16] covers the data flow, Zhang et al. [14] proposed the use of ontology for modeling the data, and Schroeder et al. [11] proposed the use of AR to model the physical space in VE domains. The combinations of these three modeling frameworks might be useful to create a robust, complete, and useful mechanism of modeling DT systems.

In conclusion, the fusion of these frameworks can be combined according to the application requirements, cost and available resources.

4 Conclusion and Future Work

DT represents physical objects such as industrial machines as digital entities. DT can help monitoring machines and understanding product performance.

DT can conceptually be divided into five dimensions, i.e., the Physical Entity (PE), Virtual Entity (VE), Services (Ss), DT Data (DD) and Connection (CN). This seminar provides a review of existing frameworks and tools, as well as how they correlate to the aforementioned DT dimensions.

Seven frameworks have been analyzed and their chosen tools for DT are also mentioned. These frameworks and tools can be combined to form a more complete framework or to create a framework specific to some cer-

tain application requirements. The tools needed to build the DT system to support the frameworks are also discussed. The key technologies used by these frameworks and tools include IoT, embedded systems, software services and APIs, data storage and processing as well as modeling and controlling software.

The research directions or future work for this DT frameworks and tools include the standardization of connection between physical to digital space, adaptive service for modeling different data format, the use of AI to analyze data for various purposes such as for predictive maintenance, benchmarking and analyzing existing protocols and message formats as well as latency impact of DT in control systems applications.

References

- [1] Diethelm Bienhaus. Patterns for the industrial internet/industrie 4.0. In *Proceedings of the 22nd European Conference on Pattern Languages of Programs*, page 17. ACM, 2017.
- [2] Darya Botkina, Mikael Hedlind, Bengt Olsson, Jannik Henser, and Thomas Lundholm. Digital twin of a cutting tool. *Procedia CIRP*, 72:215–218, 2018.
- [3] Michele Ciavotta, Marino Alge, Silvia Menato, Diego Rovere, and Paolo Pedrazzoli. A microservice-based middleware for the digital factory. *Procedia Manufacturing*, 11:931–938, 2017.
- [4] Pedro Daniel Urbina Coronado, Roby Lynn, Wafa Louhichi, Mahmoud Parto, Ethan Wescoat, and Thomas Kurfess. Part data integration in the shop floor digital twin: Mobile and cloud technologies to enable a manufacturing execution system. *Journal of Manufacturing Systems*, 48:25–33, jul 2018.
- [5] Mario Hermann, Tobias Pentek, and Boris Otto. Design principles for industrie 4.0 scenarios. In *System Sciences (HICSS), 2016 49th Hawaii International Conference on*, pages 3928–3937. IEEE, 2016.
- [6] Nane Kratzke. About microservices, containers and their underestimated impact on network performance. *arXiv preprint arXiv:1710.04049*, 2017.
- [7] Roby Lynn, Mukul Sati, Tommy Tucker, Jarek Rossignac, Christopher Saldana, and Thomas Kurfess. Realization of the 5-axis machine tool digital twin using direct servo control from cam. *National Institute of Standards and Technology (NIST) Model-Based Enterprise Summit*, 2018.
- [8] Somayeh Malakuti and Sten Grüner. Architectural aspects of digital twins in iiot systems. In *Proceedings of the 12th European Conference on Software Architecture: Companion Proceedings*, page 12. ACM, 2018.
- [9] Matthew Mikkell and Jen Clark. Cheat sheet: What is digital twin? internet of things blog. <https://www.ibm.com/blogs/internet-of-things/iiot-cheat-sheet-digital-twin/>, January 2018. (Accessed on 01/30/2019).

- [10] Qinglin Qi, Dongming Zhao, T Warren Liao, and Fei Tao. Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing. In *ASME 2018 13th International Manufacturing Science and Engineering Conference*, pages V001T05A018–V001T05A018. American Society of Mechanical Engineers, 2018.
- [11] Greyce Schroeder, Charles Steinmetz, Carlos Eduardo Pereira, Ivan Muller, Natanael Garcia, Danubia Espindola, and Ricardo Rodrigues. Visualising the digital twin using web services and augmented reality. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, pages 522–527. IEEE, 2016.
- [12] Greyce N. Schroeder, Charles Steinmetz, Carlos E. Pereira, and Danubia B. Espindola. Digital twin data modeling with AutomationML and a communication methodology for data exchange. *IFAC-PapersOnLine*, 49(30):12–17, 2016.
- [13] Fei Tao, Meng Zhang, and A.Y.C. Nee. Five-dimension digital twin modeling and its key technologies. In *Digital Twin Driven Smart Manufacturing*, pages 63–81. Elsevier, 2019.
- [14] Qi Zhang, Xiaomei Zhang, Wenjun Xu, Aiming Liu, Zude Zhou, and Duc Truong Pham. Modeling of digital twin workshop based on perception data. In *Intelligent Robotics and Applications*, pages 3–14. Springer International Publishing, 2017.
- [15] Yu Zheng, Sen Yang, and Huanchong Cheng. An application framework of digital twin and its case study. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1141–1153, 2019.
- [16] Cunbo Zhuang, Jianhua Liu, and Hui Xiong. Digital twin-based smart production management and control framework for the complex product assembly shop-floor. *The International Journal of Advanced Manufacturing Technology*, 96(1-4):1149–1163, feb 2018.

Defence Against Authentication Server Breach Attacks: A Review of Multi-Server Identity Providers

Kaspar Papli

kaspar.papli@aalto.fi

Tutor: Siddharth Rao

Abstract

Token-based authentication involving a dedicated identity provider has gained popularity with the rise of free identity providers such as Google, Facebook and Github. However, using a centralized identity provider for many services creates a single point of failure. In case of an identity provider server breach, an attacker might be able to forge arbitrary tokens to impersonate any user to any service or recover hashed passwords to perform offline cracking attacks on them. One solution is to distribute the role of an identity provider between several servers such that a breach to one server would not compromise the whole system. In this review, we outline the most important requirements and features of multi-server identity providers and compare three recently proposed multi-server identity provider systems. The compared systems are Threshold Oblivious Password-Protected Secret Sharing (TOPPSS), Anonymous-Verifying Threshold Password Authentication (AVTPA) and Password-Based Threshold Authentication (PASTA). Following the comparison, we conclude that the PASTA framework proposed by Agrawal et al. provides the strongest security guarantees and sufficient performance.

KEYWORDS: *token-based authentication, identity provider, threshold cryptography, passwords*

1 Introduction

Users are traditionally authenticated over the internet using secret user credentials, such as a username and password or password hash. In order to authenticate to a remote application, the client sends its credentials to the application where they are validated. After validation, the application typically issues a long-term access token or cookie for persisting the authentication status. This approach requires every application that wishes to authenticate users, to store some information about the users' credentials in order to validate them. This stored information typically comprises usernames and the corresponding passwords or password hashes.

Despite being generally considered bad practice, users often use the same passwords for different applications. [10] For this reason, storing passwords or password hashes in each application increases their attack surface and likelihood of compromise. If an attacker manages to compromise just one of the applications and steal its stored passwords, they could use the credentials to also illegitimately access other applications on behalf of the victim.

Recently, an approach has become popular where credential storage and validation is delegated to a dedicated authentication server or *identity provider* (IdP). In order to authenticate to an application, the client exchanges the user's credentials for a token via an IdP and sends this token to the application. The authenticity and purpose of the token can then be validated by the application. The token can be digitally signed by the IdP such that its integrity can be independently verified by the client using the IdP's public key. Alternatively, the token can be opaque to the client. In that case, the client must send the token to the IdP in order to verify its integrity. Figure 1 illustrates a generic authentication flow using an IdP.

In this approach, the sensitive user credentials are handled only by the IdP rather than every authenticating application. Additionally, single sign-on features can be implemented if one IdP provides authentication for several applications. The adoption of central identity management using identity providers is greatly facilitated by open protocols such as OAuth2 [6] and OpenID Connect [9].

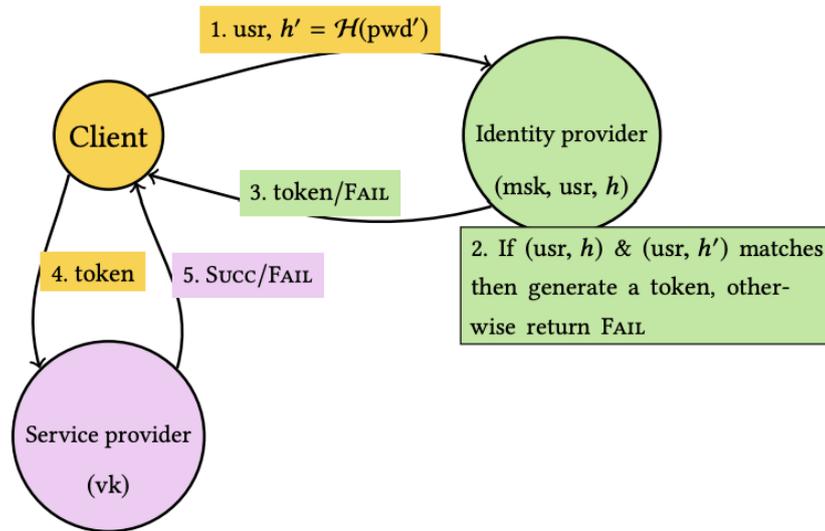


Figure 1. A generic authentication flow involving an identity provider that issues a token and a service provider that validates and consumes the token. [1]

One of the main security concerns related to dedicated identity providers is the risk of server breach wherein an attacker gains unauthorized access to an authentication server. On the compromised server, the attacker may be able to forge arbitrary tokens to impersonate users to applications. For example, this might be achieved by stealing the IdP's private key used to create signatures on tokens.

Additionally, the attacker might be able to retrieve or sniff user credentials, similarly to previously discussed traditional applications that handle user credential storage independently.

Defence against the compromise of user credentials during a server breach is a widely studied topic. Several different remedies have been widely adopted, such as password hashing, salting and employing memory-hard functions [3]. Due to the nature of human-chosen passwords, they often contain low entropy and are vulnerable to dictionary attacks [10]. These traditional remedies make recovering plaintext user credentials more difficult or expensive for the attacker, but do not solve the underlying problem: the trust put into one authentication server is fragile.

Recently, several methods have been introduced that utilize more than one authentication server such that compromising a certain subset of the servers will not enable the attacker to recover user credentials or forge authentication tokens.

The aim of this paper is to review and compare three recently proposed protocols for multi-server identity providers and to provide directions for

further research in the area. The reviewed protocols are Threshold Oblivious Password-Protected Secret Sharing (TOPPSS) [7], Anonymous-Verifying Threshold Password Authentication (AVTPA) [5] and Password-Based Threshold Authentication (PASTA) [1]. Additionally, we outline the most important security features for multi-server identity providers.

We start by reviewing and summarizing the three protocols in Section 2. In Section 3 we list required and desirable properties for the protocols and criteria for comparison. In Section 4 we present a comparison of the protocols in terms of the listed criteria. We conclude with suggestions for further research in Section 5.

2 Overview of the Protocols

In this section, we present an overview of each of the reviewed protocols. We concentrate on the high-level architecture, requirements and limitations of these protocols.

2.1 Threshold Oblivious Password-Protected Secret Sharing (TOPPSS)

Generally, Password-Protected Secret Sharing (PPSS) schemes allow a user to share some secret s between n trustees (servers) using a password pw such that s is recoverable only by running a reconstruction protocol between a predetermined threshold of t trustees using the correct password pw . While t uncorrupted trustees are needed during reconstruction of the secret, the corruption of any $t - 1$ trustees does not affect the security of s or pw . [2]

TOPPSS describes a system using n (typically $n > 1$) trustee servers communicating with one client. The secret, known to the client, is shared between all n trustees such that any subset of predefined size $t \leq n$ of trustees are able to reconstruct the secret using a client-provided password.

The system consists of two protocols: one for secret initialization and one for secret reconstruction. Both protocols require communication only between the client and trustee servers - no server-to-server communication. [7]

In the secret initialization phase, a public key infrastructure (PKI) or similar is required to ensure that the secret is shared with intended servers

(not the attacker). If no PKI exists then an attacker could impersonate trustees by performing a man-in-the-middle attack against the client and thus gain control over any number of trustees. However, in the secret reconstruction phase, no PKI is needed anymore. [7]

Jarecki et al. [7] do not specifically consider how to construct an identity provider system using TOPPSS. However, a straightforward implementation could be constructed as follows.

In the secret initialization phase, a random secret is generated for each user and secret-shared between the trustees using the client-provided password that is used for authentication. This random secret can be stored on each trustee server since it cannot be used to derive the password and thus leaking it during a server breach does not compromise the password itself.

During the secret reconstruction (i.e. client authentication) phase, the secret is constructed using a client-provided password on any of the t servers and checked against the stored secret. In case of a match, the same server establishes an authentication session, i.e. issues a token to the client.

Alternatively, the final secret validation and session establishment could always be done by one (or any other predetermined subset) of the trustee servers, called a *login server*. In this case, the account-specific secrets could be stored on only the login server. More importantly, only the login server would need the capability of establishing authentication sessions and would need to store private signing keys or symmetric encryption keys that would be compromised in case of a server breach. The only disadvantage to this approach is that the dedicated login server must always participate in the authentication phase and the system cannot operate if the login server is unavailable.

2.2 Anonymous-Verifying Threshold Password Authentication (AVTPA)

AVTPA uses a hierarchical server model that consists of a single *login server* and n backend servers. The backend servers are constrained to a separated private network and can only communicate with each other and the login server. The login server is responsible for all communication with the client. If the client needs to communicate with a backend server, the login server acts as a proxy between them. [5]

Similarly to Password-Protected Secret Sharing schemes, AVTPA is de-

signed to remain secure and functional even after the corruption of $t - 1$ backend servers, where $t \leq n$ is a predefined threshold. However, the system does not tolerate a login server breach. [5]

AVTPA defines two phases: the registration phase and the authentication phase.

In the registration phase, the client distributes key shares, created from the user's password, to randomly selected t servers. The backend servers then calculate verification shares and send them to the login server where they are aggregated and stored.

In the authentication phase, the client again distributes the respective key shares calculated from the input password to the previously selected backend servers. The backend servers calculate their verification shares and deliver them to the login server. The login server aggregates the calculated shares and checks the resulting verification value against the value stored during registration. In case of a match, the login server establishes an authentication session i.e. issues a token to the client.

2.3 Password-Based Threshold Authentication (PASTA)

PASTA, similarly to TOPPSS and AVTPA, distributes the role of an identity provider between n servers. It uses a flat server model where all servers share the same responsibilities and are functionally identical. The servers communicate only with the clients, there is no server-to-server communication. [1]

The system enables any t servers to verify passwords and establish authentication sessions in a distributed manner, where $t \leq n$ is a predefined threshold. It remains secure under the corruption of any $t - 1$ servers. [1]

PASTA uses two phases: the registration phase and the authentication phase.

In the registration phase, the client first calculates a threshold-oblivious pseudorandom function [4] on the user's password between all n servers which we denote h . It then divides h into shares h_i for each server i and distributes the shares. The shares are then stored on the servers.

The authentication phase is illustrated in Figure 2. The client chooses t servers and sends them an authentication request. Each server generates a token share, encrypts it with its corresponding h_i and sends the ciphertext to the client. The client now calculates h and each respective h_i . If the user's input password is correct, then the client can decrypt the received token shares and combine them to get a full token. At this point,

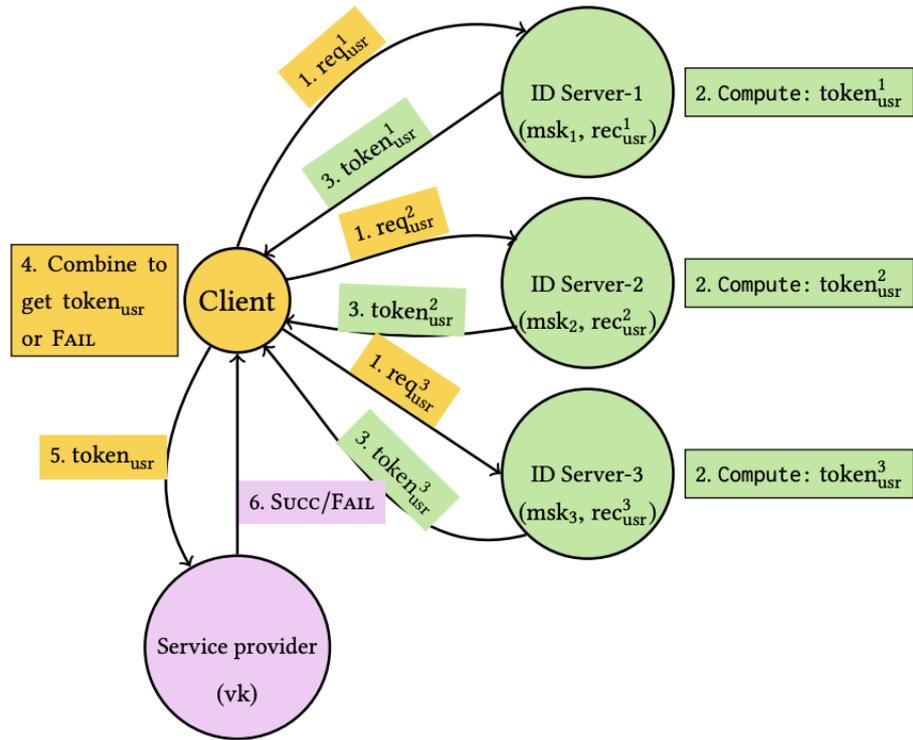


Figure 2. Authentication phase in the PASTA framework. [1]

the authentication session has effectively been established. If the user input password is incorrect, the decryption of the token shares fails and the token cannot be retrieved. Note that the authentication servers have no way of knowing whether the authentication was successful i.e. the client was able to construct the token.

3 Requirements and Criteria

In this section, we outline and explain the most important features of multi-server identity providers. We focus on the security properties and cost of using the systems. These criteria form the basis of our comparison in the next section.

3.1 Security Properties

We consider the systems' resistance to two threats for authentication servers: offline password cracking and authentication token forging.

Offline Password Cracking

When using a single-server identity provider, the server must store some verifying data about the users' credentials in order to validate them. To make recovering plain user credentials more difficult in case of server breach, this verifying information is traditionally a salted hash of the user's password for every user. Thus, in case of server breach, the attacker gains access to hashes of the users' passwords. This enables her to perform offline cracking attacks, such as dictionary attacks, on the passwords.

In a multi-server setting, the compromise of one server or a certain subset of servers should not enable the attacker to mount offline cracking attacks on the passwords using the recovered verifying data. More strongly, the verifying data should not reveal any information about the password, even if combined with verifying data recovered from other servers within the subset of compromised servers.

Token Forging

A single-server identity provider issues authentication tokens in response to valid user credentials. The tokens can contain identifying information about the user and client and are usually accompanied by a digital signature or a message authentication code (MAC). [9] The digital signature is constructed using the authentication server's private key and can be verified by any client or user using the server's public key. In case of a MAC, the authentication server and client have a pre-shared symmetric key that is used by the server to create the MAC and by the client to verify it.

In both cases, the authentication server must contain some secret information, which, if recovered by the attacker, can be used to forge arbitrary authentication tokens. This enables the attacker to impersonate any user to any application.

In a multi-server identity provider, the compromise of a certain subset of servers should not enable the attacker to forge tokens.

3.2 Performance and Cost

In all reviewed multi-server identity provider protocols, there is a performance penalty when compared to their single-server counterparts. In some cases, this penalty is inherent – when more steps or more round-trips must be completed. The penalty might also arise from worst-case

dependency. For example, when a client makes requests to n servers in parallel and needs to wait for all of them to complete, then the average waiting time is shortest when $n = 1$ and longer if n is larger. The performance is measured and compared in terms of client-server communication rounds and required calculations by both parties.

One factor that increases cost for multi-server IdPs compared to single-server implementations is server allocation and maintenance. If a method requires more authentication servers then the cost of allocating the necessary hardware and maintaining the servers is higher.

For businesses, the ultimate cost of having customers use a multi-server IdP might also be influenced by the performance of the system. If the response time of the system is slower or reliability is lower then this could negatively impact their customers' behavior.

Another aspect that affects the cost of using a system is the ease of deployability and integration. If a system is easier to deploy, configure and integrate with other systems, then this leads to a lower barrier to entry, especially for small-scale users.

4 Comparison and Discussion

In this section, we present a comparison of the three multi-server identity provider systems in terms of the criteria defined in the previous section.

4.1 Security Properties

All three systems retain their intended security properties under the corruption of at most $t - 1$ servers, where t is some predefined threshold, similarly to threshold cryptosystems. However, the intended security properties of the systems differ.

Table 1 lists the considered security properties of all systems. The systems are assumed to be run on n servers and have a fixed threshold parameter $t \leq n$. They are evaluated both in the case of having less than t corrupted server and having t or more corrupted servers.

TOPPSS and PASTA resist offline password cracking attacks if less than t servers are compromised. AVTPA is secure against offline cracking if less than t backend servers are breached but becomes vulnerable if the dedicated login server is compromised. If t or more servers are compromised, no system remains secure.

	T-1	A-1	P-1	T-t	A-t	P-t
Offline cracking	+	+	+	-	-	-
Token forgery	-	-	+	-	-	-

Table 1. Security properties of TOPPSS (T), AVTPA (A) and PASTA (P). All systems are considered with less than t compromised server (T-1, A-1, P-1) and t or more compromised servers (T-t, A-t, P-t). "+" means that a system is secure against that kind of attack, "-" means that it is not secure.

However, only PASTA is secure against token forgery attacks when less than t servers are corrupted. PASTA is also the only system to address this attack. In both TOPPSS and AVTPA, the task of token generation and delivery (i.e. persisting an authentication session) in a distributed setting remains unaddressed and unsolved.

This is facilitated by the authentication algorithm, where separate token shares are generated on each of the t servers and sent encrypted to the client. Thus the whole token can be reconstructed only by the client after decrypting the token shares using the user's password. However, if t servers have been compromised, the attacker can spoof all t token shares and create arbitrary tokens.

In TOPPSS, if using a system without a dedicated login server (as described in Sec. 2.1), then a compromise of any server in the system enables an attacker to forge arbitrary tokens. However, if using a login server, only the compromise of the login server would enable token forgery.

AVTPA, similarly to TOPPSS, is vulnerable to token forgery only if the dedicated login server is breached since all tokens are issued from there.

In the AVTPA hierarchical server model, the login server is the only server that is accessible from the internet. All backend servers are constrained to a private network and can only communicate between themselves and with the login server. This makes compromising backend servers more difficult and unlikely. However, this does not impact our reviewed security properties since breaches to the login server alone are more dangerous than to backend servers.

4.2 Performance and Cost

TOPPSS is currently the most efficient PPSS scheme developed. In the secret reconstruction (authentication) phase, it requires one client-server communication round (one request and one response per server), two exponentiations on the client side and one exponentiation on each server. [7]

In the authentication phase, AVTPA uses two communication rounds between the client and login server and it also requires two additional communication round between the login server and backend servers. The client must compute $2t + 4$ exponentiations, where t is the threshold parameter. The login server computes $2t + 4$ exponentiations and the backend servers compute $t + 6$ exponentiations in total, or $1 + \frac{6}{t}$ per server. [5]

PASTA can be instantiated with different threshold token generation (TTG) schemes and threshold-oblivious pseudorandom functions (TOPRF), and its performance depends solely on them. For example, if using 2Hash-TDH proposed by Jarecki et al. [7] as the TOPRF and the TTG scheme by Naor et al. [8], the client must perform $t + 2$ exponentiations and two exponentiations must be performed on each server. [1]

AVTPA and PASTA have published implementations of their systems but neither has been integrated into existing identity provider software. Both are deployable as independent applications. There is no concrete specification or implementation of the IdP system using solely TOPPSS.

All systems require roughly the same number of servers to achieve their intended security properties. However, PASTA is moderately more efficient in regards to resource usage, because it does not use a dedicated login server.

5 Conclusion

The purpose of this review was to outline the most important security properties of multi-server identity providers in terms of server breach and compare three recently proposed multi-server IdP systems.

The two most important security features we identified are the resistance to offline password cracking attacks and token forging attacks. Firstly, when an attacker manages to compromise an authentication server, they should not be able to recover any such information about the users' passwords on which they could perform offline cracking attacks, for example dictionary attacks. Secondly, an attacker should not be able to forge arbitrary authentication tokens (i.e. create authentication sessions) on a compromised server.

We reviewed three systems – TOPPSS, AVTPA and PASTA – in terms of these security properties and their performance. PASTA provides the strongest security properties, it is secure against both offline password cracking and token forgery attacks when less than t (a predefined thresh-

old parameter) servers have been corrupted. TOPPSS and AVTPA both lack resistance against token forgery attacks. A simple IdP system using TOPPSS is the most efficient in terms of communication rounds and calculations.

This review has some limitations that should be taken into consideration when generalizing the findings. Firstly, we only considered security properties that are unique to multi-server identity providers during a server breach. We did not examine other important security features, such as resistance to denial-of-service attacks, protocol implementation correctness or web security features. Secondly, we do not consider the reliability or usability of the systems, which are important in a real-life security setting involving real users.

We believe more research into the topic of multi-server identity providers is needed. The main research directions are developing new multi-server identity provider systems with strong security properties or performing security and performance analyses of existing systems. Further work is also needed to integrate existing systems, such as PASTA, into open identity provider software. In addition, we urge public identity providers, such as Google and Facebook, to contribute to the research of multi-server IdP systems.

References

- [1] Shashank Agrawal, Peihan Miao, Payman Mohassel, and Pratyay Mukherjee. Pasta: Password-based threshold authentication. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS '18*, pages 2042–2059, New York, NY, USA, 2018. ACM.
- [2] Ali Bagherzandi, Stanislaw Jarecki, Nitesh Saxena, and Yanbin Lu. Password-protected secret sharing. In *Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS '11*, pages 433–444, New York, NY, USA, 2011. ACM.
- [3] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. Argon2: new generation of memory-hard functions for password hashing and other applications. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 292–302. IEEE, 2016.
- [4] Michael J Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Key-word search and oblivious pseudorandom functions. In *Theory of Cryptography Conference*, pages 303–324. Springer, 2005.
- [5] Mengxiang Guan, Jiaying Song, and Weidong Liu. A threshold multi-server protocol for password-based authentication. In *2016 IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud)*, pages

108–118. IEEE, 2016.

- [6] D. Hardt. The OAuth 2.0 Authorization Framework. RFC 6749, The Internet Engineering Task Force, October 2012. <https://tools.ietf.org/html/rfc6749>.
- [7] Stanisław Jarecki, Aggelos Kiayias, Hugo Krawczyk, and Jiayu Xu. Topps: Cost-minimal password-protected secret sharing based on threshold oprf. In *International Conference on Applied Cryptography and Network Security*, pages 39–58. Springer, 2017.
- [8] Moni Naor, Benny Pinkas, and Omer Reingold. Distributed pseudo-random functions and kdcs. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 327–346, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.
- [9] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros, and C. Mortimore. OpenID Connect Core 1.0 incorporating errata set 1. Technical report, The OpenID Foundation, November 2014. https://openid.net/specs/openid-connect-core-1_0.html.
- [10] Ding Wang and Ping Wang. Offline dictionary attack on password authentication schemes using smart cards. In *Information security*, pages 221–237. Springer, 2015.

Formal Verification of EAP-TLS with mCRL2

Giacomo Mariani

giacomo.mariani@aalto.fi

Tutor: Aleksi Peltonen

Abstract

Modern information and communication technology (ICT) systems and protocols have increased in complexity, which makes the correct verification of their properties more difficult. The use of formal verification, and in particular model-based verification, is useful to check that a protocol design responds to the desired properties given by its specification. This paper presents an example of a model-based verification of the EAP Transport Layer Security (EAP-TLS) protocol, using the micro common representation language 2 (mCRL2). The contribution of this paper is in giving the reader an understanding of what is formal verification in general and model-based verification in specific, it also explains EAP and EAP-TLS, and using a simplified model, describes how to carry a model-based verification of the desired protocol.

KEYWORDS: formal verification, model checking, EAP, mCRL2

1 Introduction

Due to the great complexity of modern information and communication technology (ICT) systems, it is no longer sufficient to verify their correct-

ness only with software and hardware testing. The amount of test cases that need to be checked is often so large, that verifying all of them becomes extremely challenging [1]. Formal verification is a group of methods used to establish the correctness of a system by applying mathematics to model and analyze the system [2]. This type of verification is usually applied on a protocol or another system, before it is implemented and before software testing, in order to establish that it behaves as stated by its specifications. Model-based verification is a formal verification method where a model of the system to be analyzed is created and then systematically checked, to ensure that it satisfies the desired properties [2].

Previous researches have used model-based verification to analyze Extensible Authentication Protocol (EAP) methods built on top of the EAP framework, and the tools used to model and analyze were SPIN [3, 4], Casper-CSP and FDR [5], improved authentication test model [6] and $\text{Mur}\varphi$ [7]. Using the model-based verification the researchers found vulnerabilities in EAP-MD5 and EAP-GPSK. In the verification of EAP, Hegde et al. [3] established that the final state of the model is reachable from the initial state by all the possible paths.

The scope of this paper is to present an example of model-based verification of EAP Transport Layer Security (EAP-TLS) done using the micro common representation language 2 (mCRL2) tool [8]. The goal is to familiarize the reader with formal verification and model checking, and to give a demonstration of how these verification techniques can be used with mCRL2 formalism to verify the EAP-TLS protocol. Section 2 introduces the concepts of formal verification and model-based verification. Section 3 discusses EAP, EAP-TLS and the steps involved in the authentication process. Section 4 is about the modelling and verification of EAP-TLS with mCRL2, and finally Section 5 presents the conclusions of this paper.

2 Formal verification

In order to establish that a product or design works correctly, and therefore possesses the properties described by its specifications, different types of system verification can be applied during the design, development and implementation process [2], including software testing, hardware testing and formal verification. The use of formal methods involves producing a mathematical proof to demonstrate that the system satisfies the required properties [9]. In order to reach a mathematical proof, it is necessary to

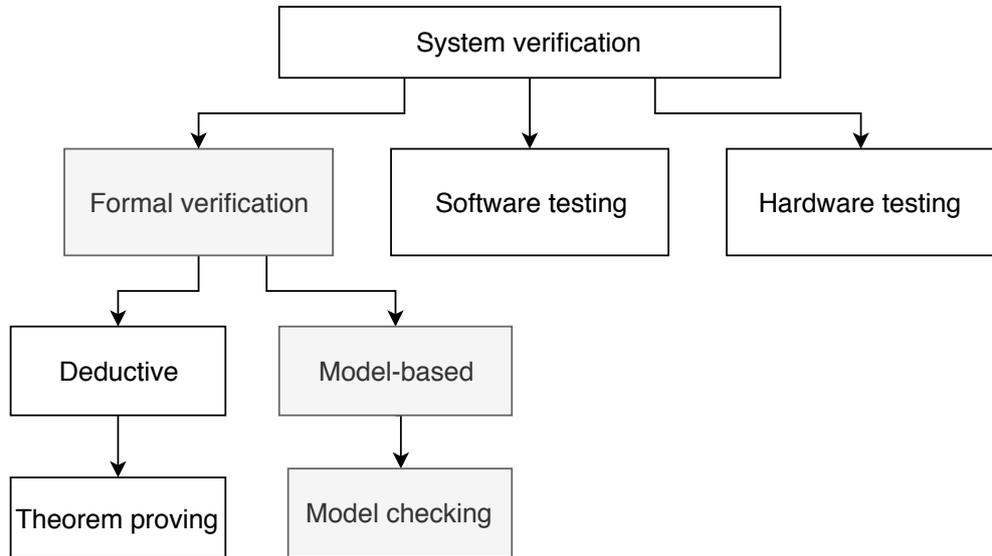


Figure 1. Types of System Verification

model the system under examination using a mathematical structure and then declare the system properties using mathematical theorems [9]. In this way it will be possible to mathematically verify if the created model, representing the system, satisfies the theorems, representing the desired properties.

The differences between various formal verification techniques lie mostly in the formalism chosen to engage with the mathematical reasoning about the system to be verified [9]. As shown in Figure 1, formal methods of verification can be distinguished into two main categories: deductive verification, based on theorem proving, and model checking, based on the creation of formal models describing when a system can be said to satisfy a certain formula [9].

2.1 Model-based verification

In model-based verification the model created describes the system's behaviour in an explicit and clear manner. With the use of a software tool the model and all its possible states are then checked systematically to verify that they all satisfy the desired properties [2]. One of the challenges of model-based verification, is that the models tend to become very large, especially if they include data and parallel behaviours. This is known as the *state space explosion problem* [1]. It is thus important to apply some restraints when modelling the behaviour of a system, but at the same time if the model is not detailed enough, we run the risk of producing inaccurate results.

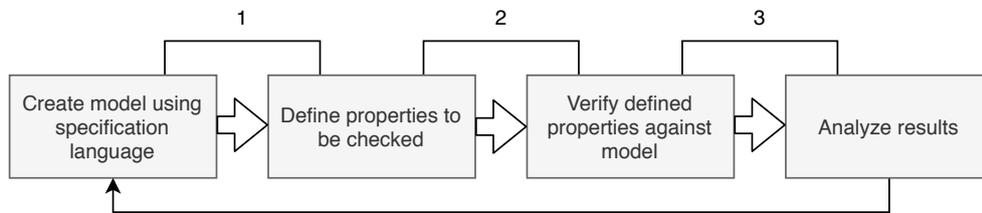


Figure 2. Model-Based Verification Phases. Adapted from Baier and Katoen (2008) [2]

Model-based verification can be divided in three stages, as shown in Figure 2: (1) the modelling phase, (2) the running phase and (3) the analysis phase. In the modelling phase, using the language of choice, the model of the system to be verified is constructed and undergoes the first assessments. In this phase also the properties to be checked in the later stages are formalized. In the running phase the ready model is executed and the defined properties are checked. In the analysis phase the outcome of the simulation is analyzed, including violated properties and flaws in the design of the model, which might require going back to the modelling phase. Violated properties will have to be checked to verify if the violation is due to the system itself or to a flaw in the model. [2]

3 Extensible Authentication Protocol

The Extensible Authentication Protocol (EAP) [10] is an authentication framework used by many protocols, because it offers reliable transport of authentication data and supports a wide range of authentication methods, both on wired and wireless links. In its first specification [11] EAP was defined as an authentication extension for the Point-to-Point Protocol (PPP), but it later became part of other protocols as well, such as EAP over LAN (EAPOL) and the Protected Extensible Authentication Protocol (PEAP). Each protocol implementing EAP is responsible for the encapsulation of the EAP messages.

EAP supports many authentication methods and beside EAP-TLS, which we describe in Section 3.1, some other examples of authentication methods applied to EAP are: EAP Protected One-Time Password Protocol (EAP-POTP) [12], designed for the use of One-Time Password tokens, EAP Pre-Shared Key (EAP-PSK) [13], intended for mutual authentication and session key derivation with Pre-Shared Key, and EAP Subscriber Identity Module (EAP-SIM) [14], for session key distribution and authentication

using the Subscriber Identity Module (SIM) of the Global System for Mobile Communication (GSM).

With EAP the communication between peer and authenticator is initiated by the latter, after which, the necessary information is requested from the peer, and the desired type of authentication method is selected. All available authentication methods do not necessarily need to be supported by the authenticator, which can instead act as a pass through for the back-end authentication server.

The EAP standard [10] specifies the following steps in the authentication process:

1. The server (authenticator) sends a request to authenticate the client (peer) and specifies the data being request with a type field (for example Identity or MD5 challenge).
2. The peer sends a response packet containing a type filed corresponding to the one being request.
3. The authenticator sends another request packet, to which the peer replies with a response.
4. The sequence of request and responses continues until the authenticator can either determine that a successful authentication has occurred, or it cannot authenticate the peer.

3.1 EAP Transport Layer Security

EAP Transport Layer Security (EAP-TLS) [15] was created to give support for certificate-based mutual authentication and key-derivation capabilities, which are not supported by the original EAP specification [10]. This was done by adding to EAP the capabilities of the TLS protocol [16]. As previously described in Section 3, in the EAP protocol the authenticator initializes the conversation by requesting the peer identity, to which the client responds with the required information. After this first exchange the EAP protocol defines that the communication continues as long as needed. In EAP-TLS, after the first identification exchange has occurred, the authenticator starts the EAP-TLS communication, where each request is an EAP-Request of type EAP-TLS, and each response is

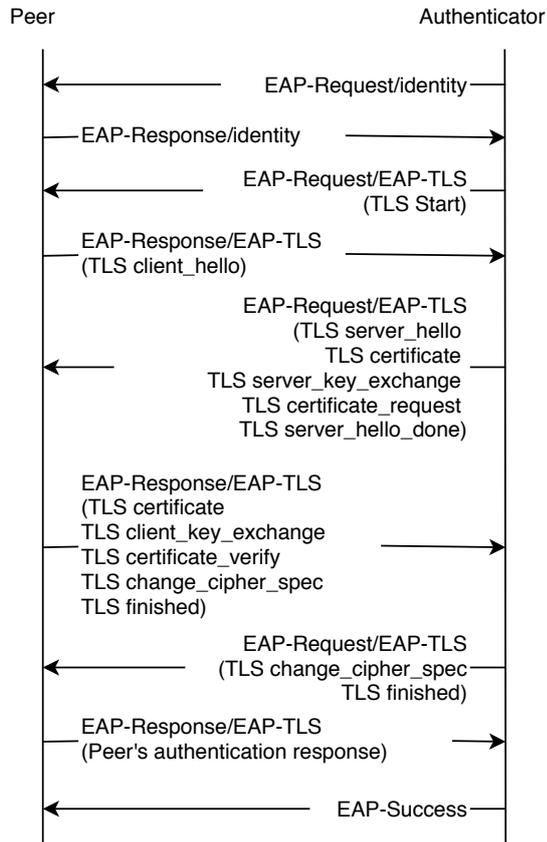


Figure 3. EAP-TLS Exchange. Adapted from the EAP-TLS standard [15]

an EAP-Response of type EAP-TLS. These EAP packets contain the information required by the TLS protocol. The communication between authenticator and peer is synthesized in Figure 3.

If the authentication of the peer is unsuccessful, the authenticator should send the peer a TLS Alert message containing the authentication response from the EAP server. Only after receiving a response from the peer, the authenticator can end the conversation sending an EAP-Failure packet. If the server authentication is unsuccessful, the peer may send a TLS Alert message and then wait for the server's EAP-Failure packet. In this paper the cases of session resumption and peer re-sending the TLS client_hello message are not covered.

4 Modelling and Verification

In this paper, we model and verify EAP-TLS using mCRL2 [8], a formalism based on the algebra of communicating processes, which is a way to describe concurrent systems in algebraic terms. The mCRL2 formalism is also extended with features on data, time and multi-actions. Using

mCRL2, the model is constructed based on the system specifications, using atomic actions and interactions composed with algebraic operators. The system requirements, including the allowed and forbidden behaviors, are described by using the modal mu-calculus with data and time. Finally the behaviour of the model can be simulated, tested against the requirements and visualized [1].

When constructing a model with mCRL2, we have to define the actions present in the model, the processes interacting in the system, which in our case would be the peer and the authenticator, and which actions each process takes. For example we could model a system with process A and process B, and after declaring the range of existing actions we can define that process A takes *action1a* followed by *action2a* or *action3a*, while process B takes *action1b* followed by *action2b* or *action3b*. Finally we define which sequence of actions triggers a state transition. In the mCRL2 syntax the concatenation of actions is defined with a "." (dot) sign, and the "or" with a "+" (plus) sign.

4.1 Modelling of EAP-TLS

We created the model of EAP-TLS on the bases of the communication between peer and authenticator, as described in the EAP-TLS standard [15] and as shown in Figure 3. From the EAP-TLS model we excluded the cases in which the client is re-initiating a previous sessions or re-sending the *TLS client_hello* message. Figure 4 shows how the peer/authenticator communication from Figure 3 was mapped into actions, states and transitions for the mCRL2 model. The mCRL2 code for modelling the EAP-TLS protocol is shown in Appendix A. The code also includes the actions and transitions representing error handling when the peer can not authenticate the server or the server can not authenticate the peer.

We declared some of the expected properties of the EAP-TLS protocol according to the EAP-TLS specification [15]. These properties specify that: (1) an EAP response of type Identity is only sent in response to an EAP request of type Identity, (2) if the peer sends a TLS Alert packet, the authenticator responds with an EAP Failure, (3) if the authenticator hello message is valid, the peer responds with the required data, otherwise with a TLS Alert message. The code with the specified properties used for the verification is show in Figure 5.

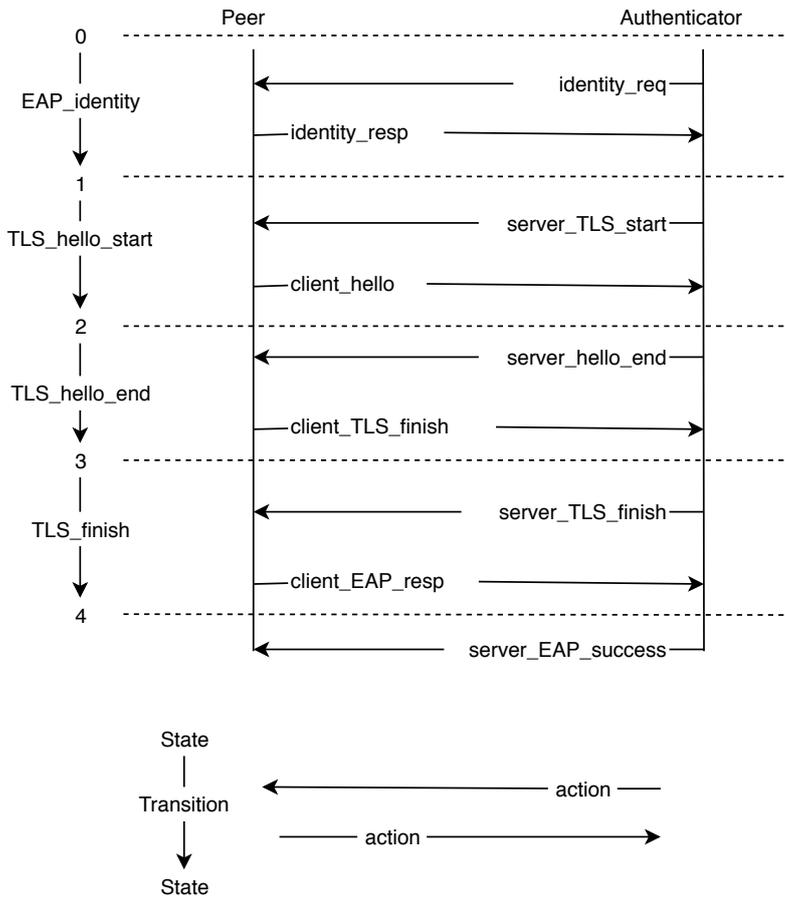


Figure 4. EAP-TLS States and Transitions

```

1  % An EAP-Response/Type=Identity is only sent in response to an
   % EAP-Request/Type=Identity
2  [!identity_req . identity_resp] false
  
```

```

1  % If the peer sends a TLS Alert packet, the authenticator has to respond
   % with an EAP-Failure
2  [client_TLS_alert . server_EAP_failure_user_disconnected] true
  
```

```

1  % If the authenticator server hello is valid, the client responds with
   % the required data, otherwise with a TLS Alert
2  [server_hello_end . !client_TLS_finish] false && [invalid_server_msg .
   client_TLS_alert] false
  
```

Figure 5. EAP-TLS Properties Declaration

4.2 Verification of EAP-TLS

In order to run and verify the model, we compiled the model code to a linear process, then transformed the linear process file into a labeled transition system and finally visualized the outcome, which can be seen in Figure 6. From the visualization of the linear transition system we can see that the process starts at state zero, from which an *EAP_identity* transition causes the state to change to one. The *EAP_identity* transition is the result of the the action *identity_req* followed by the action *identity_resp*, as modelled in Figure 4 and declared in the model code shown in Appendix A at line 41. In the same way each following transition is the result of two actions and takes the system into a new state. Because the model we constructed is relatively simple, the majority of the states have only one outgoing transition. State two and state five have instead two outgoing transitions. The system can continue from state two to state three and finish the TLS handshake, or can go to state four if the peer finds that the message sent by the authenticator is invalid. State five is reached if the handshake was successful, and from here the system goes back to the initial state if the client was authenticated successfully, otherwise it goes to state six and then to the initial state after the authenticator has informed the client of the failure.

To verify that the model respects the properties we declared in Figure 5, we run the properties files against the model linear process file and checked the outcome. For all the three properties the outcome was true, so we could verify that: (1) an EAP-Response of type Identity can only be sent in response to an EAP-Request of type identity, (2) if the peer sends a TLS Alert packet, the authenticator has to respond with an EAP-Failure, (3) if the authenticator server_hello is valid, the client responds with the required data, otherwise with a TLS Alert.

5 Conclusion

Formal model-based verification is an essential tool to verify that the properties of a protocol respond to its specifications, and with the use of formal model-based verification, it is possible to detect errors and security flaws in a protocol before it's implementation. This paper introduced formal model-based verification, the EAP and EAP-TLS protocols and demonstrated how formal model-based verification can be applied to

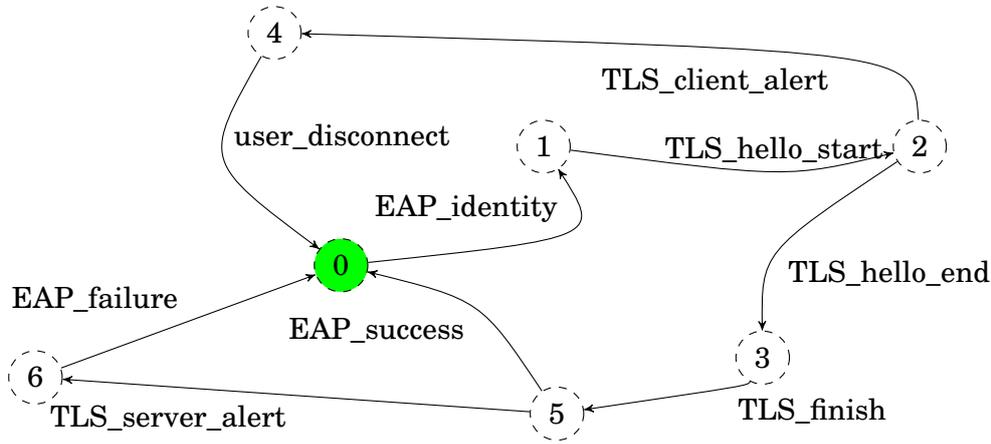


Figure 6. Labeled transition system of EAP-TLS

these protocols using mCRL2 formalism. The given demonstration is not a complete verification of the EAP-TLS protocol, but rather a start for a further, more complete and in depth application of mCRL2 formalism to verify the EAP-TLS or other protocols that use EAP as a framework. Our results show that mCRL2 formalism can be used to model the EAP-TLS protocol, that it's properties can be declared based on the protocol specification and then checked against the created model. The contribution of this paper is that it can be used as a reference by those who want to verify the correctness of EAP-TLS and other EAP methods or are interested in formal model-based verification and mCRL2 formalism.

References

- [1] J. F. Groote and M. R. Mousavi, *Modeling and Analysis of Communicating Systems*. MIT Press, Aug. 2014.
- [2] C. Baier and J.-P. Katoen, *Principles of Model Checking*. MIT Press, Apr. 2008.
- [3] M. S. Hegde, H. K. Jnanamurthy, and S. Singh, “Modelling and Verification of Extensible Authentication Protocol using SPIN Model Checker,” *International Journal of Network Security & Its Applications (IJNSA)*, vol. 4, pp. 81–98, 2012.
- [4] H. B. Ali, M. R. Karim, M. Ashraf, and D. M. W. Powers, “Modeling and verification of Extensible Authentication Protocol for Transport layer Security in Wireless LAN environment,” in *2010 2nd International Conference on Software Technology and Engineering*, vol. 2, pp. V2–41–V2–45, Oct. 2010.
- [5] I.-G. Kim and J.-Y. Choi, “Formal verification of PAP and EAP-MD5 protocols in wireless networks: FDR model checking,” in *18th International Conference on Advanced Information Networking and Applications, 2004. AINA 2004.*, vol. 2, pp. 264–269 Vol.2, Mar. 2004.
- [6] X. Li, L. Hao, S. Yang, and J. Li, “Formal Verification of EAP-AKA with Improved Authentication Tests,” in *2006 International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–4, Sept. 2006.
- [7] J. C. Mitchell, A. Roy, P. Rowe, and A. Scedrov, “Analysis of EAP-GPSK Authentication Protocol,” in *Applied Cryptography and Network Security* (S. M. Bellovin, R. Gennaro, A. Keromytis, and M. Yung, eds.), Lecture Notes in Computer Science, pp. 309–327, Springer Berlin Heidelberg, 2008.
- [8] J. F. Groote, A. Mathijssen, M. Reniers, Y. Usenko, and M. van Weerdenburg, “The formal specification language mcrl2,” in *Methods for Modelling Software Systems (MMOSS)* (E. Brinksma, D. Harel, A. Mader, P. Stevens, and R. Wieringa, eds.), no. 06351 in Dagstuhl Seminar Proceedings, (Dagstuhl, Germany), Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.
- [9] S. Ray, *Scalable Techniques for Formal Verification*. Springer Science & Business Media, June 2010.
- [10] J. R. Vollbrecht, B. Aboba, L. J. Blunk, H. Levkowitz, and J. Carlson, “Extensible Authentication Protocol (EAP),” June 2004.
- [11] L. Blunk and J. Vollbrecht, “PPP Extensible Authentication Protocol (EAP),” Mar. 1998.
- [12] M. Nystroem, “The EAP Protected One-Time Password Protocol (EAP-POTP),” Feb. 2007.
- [13] F. Bersani and H. Tschofenig, “The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method,” Jan. 2007.
- [14] H. Haverinen and J. Salowey, “Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM),” Jan. 2006.

- [15] D. Simon, B. Aboba, and R. Hurst, "The EAP TLS Authentication Protocol," Mar. 2008.
- [16] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1," Apr. 2006.

Appendix A

Model Code

```
1 % Here we define the actions and transitions names
2 act
3     identity_req, identity_resp, EAP_identity;
4     server_TLS_start, client_hello, TLS_hello_start;
5     server_hello_end, client_TLS_finish, TLS_hello_end;
6     server_TLS_finish, client_EAP_resp, TLS_finish;
7     server_EAP_success, client_no_resp, EAP_success;
8     server_EAP_failure, client_failure_no_resp, EAP_failure;
9     server_TLS_alert, client_EAP_alert_resp, TLS_server_alert;
10    invalid_server_msg, client_TLS_alert, TLS_client_alert;
11    server_EAP_failure_user_disconnected, client_disconnected_no_resp,
        user_disconnect;
12
13 % Here we define the two processes and their possible action sequences
14 % dot (.) for concatenation and a plus (+) for "or"
15 proc
16     Auth =
17         identity_req.server_TLS_start.(
18             (invalid_server_msg.server_EAP_failure_user_disconnected) +
19             (server_hello_end.server_TLS_finish.(
20                 (server_TLS_alert.server_EAP_failure) +
21                 server_EAP_success)
22             )
23         ).Auth;
24
25     Peer =
26         identity_resp.client_hello.(
27             (client_TLS_alert.client_disconnected_no_resp) +
28             (client_TLS_finish.client_EAP_resp.(
29                 (client_EAP_alert_resp.client_failure_no_resp) +
30                 client_no_resp)
31             )
32         ).Peer;
33
34
35 % We specifies which are allowed state transitions and which sequence of two
        actions causes a state transition
36 init
37 allow(
38     {EAP_identity, TLS_hello_start, TLS_hello_end, TLS_finish,
        TLS_server_alert, TLS_client_alert, user_disconnect, EAP_success,
```

```
        EAP_failure},
39  comm(
40  {
41  identity_req|identity_resp -> EAP_identity,
42  server_TLS_start|client_hello -> TLS_hello_start,
43  server_hello_end|client_TLS_finish -> TLS_hello_end,
44  server_TLS_finish|client_EAP_resp -> TLS_finish,
45  server_TLS_alert|client_EAP_alert_resp -> TLS_server_alert,
46  invalid_server_msg|client_TLS_alert -> TLS_client_alert,
47  server_EAP_success|client_no_resp -> EAP_success,
48  server_EAP_failure|client_failure_no_resp -> EAP_failure,
49  server_EAP_failure_user_disconnected|client_disconnected_no_resp ->
        user_disconnect
50  },
51  Auth || Peer
52  )
53  );
```

Privacy-Preserving Machine Learning using trusted hardware

Yuxi Xia

yuxi.xia@aalto.fi

Tutor: Samuel Marchal

Abstract

Machine learning (ML) is applied in sensitive and security-critical domains, thus, the need for integrity and privacy guarantees for ML computation as well as security for ML models is growing rapidly. In this paper, we surveyed the solutions that used trusted hardware to provide trusted execution environment (TEE) for ML computations and ML models. Firstly, we list typical security and privacy issues in machine learning applications. Then we summarize the current work for privacy-preserving in ML to several different categories, and identify the strengths and drawbacks individually. To conclusion, we identify the possible novel directions to explore in this field.

KEYWORDS: *security, privacy, machine learning, hardware, TEEs*

1 Introduction

Machine learning (ML) models are becoming more and more popular because of their fast, accurate predictions.

In order to achieve an optimal outcome, ML models require a large amount of training data, and to use these models, clients should provide

their own personal data for prediction. Three parties are involved in this process which are the owner of the ML model, the owner of training data for implementing ML model, and the clients who provide their own data for prediction. These parties require a security and trusted environment to guarantee their personal data or intellectual property (ML model) will not be revealed to public. Thus, privacy-preserving in ML to achieve secure computation, data protection becomes a requirement of these parties as well as a challenge for researchers. One way to realize this goal is to execute ML models in isolated hardware components created by trusted execution environment (TEE). TEE is well known as an isolated processing environment in which applications can be securely executed irrespective of the rest of the system [17]. The set of features intended to enable trusted execution are the following: secure provisioning, trusted path, remote attestation, isolated execution and secure storage [21]. The TEEs improve the security of machine learning applications with its powerful features. They can be used to protect the privacy of clients' input data and the confidentiality of the ML model as well as the training data used for training the model.

Although machine learning achieved marvellous success in many fields, the technical community's understanding about the weaknesses inherent to the design of systems built on ML and the solutions to defend against them are not mature [15]. Therefore, advancing the science of the security and privacy in ML is essential and significant in future research. In this paper, we survey the exiting papers using TEEs for privacy-preserving problems in machine learning and summarize the approaches that have been proposed. We clarify existing work into two categories, some works mainly enhanced the confidentiality of the ML models and clients' data, the left focus on the privacy of the training data. Then, we evaluate the highlights and drawbacks of these approaches individually to identify the possible novel directions to explore in this field.

2 Privacy-preserving problems in Machine Learning

Deep learning (DL) is a class of techniques which makes major advances in solving problems that have resisted the best attempts of the artificial intelligence (AI) community [12]. The breakthroughs of DL makes it possible to achieve human-performance applications of computer vision, speech recognition and game playing. For instance, ML systems can be

used to identify objects in images, transform speech to text and select relevant results of search. In a normal machine learning application, AI cloud service providers offer two independent DL services which are inference and training. By feeding training services with clients' own private training data, they can build individual DL models. After that, the DL models can be uploaded which are in the form of hyperparameters and weights of Deep Neural networks (DNNs) to inference services, this means the data of the clients might be hosted by different AI service providers used for training services [4]. As AI services such as Google Cloud AI, Amazon AI become more and more popular for the public, the data security of the user raised more attention.

We categorized security and privacy problems in Machine Learning in the following.

2.1 Confidentiality and integrity of ML models

Machine learning are applied to sensitive data to make decisions as it becomes more and more popular. Machine learning as a service (MLaaS) offers convenience for service providers to deploy ML models and clients to try models in different applications [20]. The limitation of MLaaS are obvious, since it requires data transfer over the networks, there are high volume communication and the chance of being attacked will be increased [13].

In many cases, cloud services are responsible for ML computations, but if the cloud provider is not honest, it can save resources by running a weaker model [3], use unfair methods for different users, or tamper the result of computations on purpose. Take a medical health DNN-based model for example, this model does not have its own cloud services which means it will need to rely on a third-party cloud provider thus the model can be available to clients. While a compromised cloud services could steal the code and data even the training model, this would be a heavy loss to the service provider.

2.2 confidentiality of clients' data

Service providers offer clients convenience with various ML models for different needs. Although clients generally trust the service providers, they can not ensure the data security of their inputs to the AI service.

Take smart-home device Amazon Alexa for example, voice data is trans-

ferred to cloud services for automated speech recognition through this device. The security and privacy risks here is significant because voice data is highly sensitive, it contains the spoken words of the user and biometric information [2]. If the voice data is not well protected, it can be abused such as impersonation attacks and lead to great loss to the user.

The confidentiality of clients' data can be compromised by accidental disclosures of sensitive user data due to exploited vulnerabilities of the system, misoperations by negligent system administrators or data thefts [4].

2.3 Privacy of training data

Machine Learning models and applications are based on a large amount of diverse training data. Users could benefit from gathering their private datasets to training a more precise and efficient ML model. For instance, if multiple hospitals can reach an agreement to share the patients' data for training a ML model which can help them to diagnose a disease, it would be much better than just use the training data from one hospital. Moreover, they all can benefit from the data sharing. The issue we have to consider is that the owner of the data might like to keep their data private. While the multi-party machine learning system would have more probability to leak a large amount of data because of the exploitation of any side channels induced by network, memory, and disk access patterns [14]. In addition, when we use traditional ML services, the user is revealing the data that can give them a competitive advantage [8]. However, the training data might be sensitive and private which means the owners will not expect the revealing their data to these ML service providers.

At present, data privacy requirements are a challenge for collecting high-quality datasets. Although some attempts have been done to create data marketplaces and distributed AI for some public datasets [11]. For example, smart contracts, which is well-known to public as blockchains, can achieve reaching agreements on the result of computations, but its verification of correctness needs public disclosure of contract state and inputs which would cause a problem for data marketplaces because the data and models will not be secure any more to the user of blockchain.

3 Current work using TEEs for privacy-preserving

There is an increasing need for us to find proper solutions for every possible scenario. In order to provide a security guarantee for the three parties involved in the ML computation process, some researches have been done by using trusted hardware. We summarized some of these researches and categorized what kinds of problems they can solve.

3.1 Ways for protecting confidentiality of clients' data and ML models

ML models are intellectual property of service providers. Finding solutions to protect ML models is essential in a long term and significant for the motivation of creations. In addition, many information-leaking incidents happened in recent years, which motivate us to find more efficient and proper solutions for protecting the confidentiality of clients' data. The good protection of clients' data can also help the service providers earn more confidence from clients.

Slalom [20] is a framework that outsources execution of all linear layers in a Deep Neural Networks (DNNs) from any trusted environment (e.g., ARM TrustZone [1]) to a faster co-located device. This is a novel approach to the challenge: "how do we efficiently leverage TEEs for secure machine learning computations?" By partitioning computation between trusted and untrusted devices, we can learn the high performance execution of DNNs.

There are three steps in Slalom for outsourcing ML from a trusted but slower device to an untrusted but faster device (Figure 1). The first step is quantization which is to quantize weights and inputs and embedded in a field F . Then, we have to verify the integrity of linear layers that have been outsourced to an untrusted device. The last step is the input privacy, we can hide the inputs of linear layers with random masks, thus, we can promise the ML's input is secure.

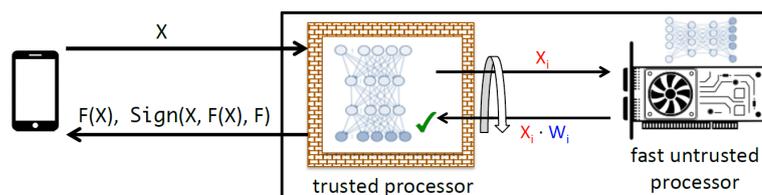


Figure 1. Outsourcing ML to and from trusted hardware[20].

Slalom improved the efficiency of evaluating a DNN in a TEE to guarantee the integrity and privacy of a ML computation, it is also an admirable solution for protecting the confidentiality of ML model and clients' data during the prediction time.

MLCapsule [6] is a client-side solution for off-line computation. Assume the client has a platform which can access a TEE, MLCapsule uses TEE to provide a secure enclave to run the ML model for classification inference. The main idea of MLCapsule is to make full use of the advantages of TEE, i.e., provide an isolated environment for running the codes. First of all, the client needs to verify that the code is running in an isolated environment before it is secreted by the service provider. After that, the client can do prediction task on the enclave. These can achieve the goal that the client can not get more information about the ML model than API access to a server-side model. MLCapsule guarantees the isolation provided by the hardware, the client gains no additional benefits in stealing the ML models comparing to access to the ML model by service-side APIs.

MLCapsule is a novel mechanism for ML models, it protects the models as well as the user data as it never leaves the client.

PRIVADO [19] is a system designed for secure inference-as-service, which is input-oblivious, fully-automated and with a low task control block. Input-oblivious means any DNN framework written in C/C++ will be transformed to be free of input-dependent access pattern. It does not need any developer effort which is fully-automated, low task control is because it can generate compact C code for a ML model that can run within Intel SGX-enclaves [16]. PRIVADO also has low performance overhead.

The main idea of PRIVADO is that deep learning (DL) models have very few data-dependent accesses but a lot data-independent accesses. It uses the PRIVADO-Generator which takes as input models represented in the popular ONNX format which is an open format to represent DL models and provide convenience for AI developers to move models, the generator can automatically create a minimal set of enclave-specific code and encrypted parameters for the model.

PRIVADO is the first system to solve challenges such as enclave memory is susceptible to leakage via data access patterns [18], the trouble to develop custom applications for enclaves and high performance overheads.

DeepEnclave [4] is a privacy-enhancing system developed for protecting confidentiality of clients' input data during a ML computation. DeepEnclave mitigates sensitive information exposed in DL inference pipelines.

Clients are allowed to submit encrypted FrontNet and encrypted inputs to the system. Intel SGX on cloud infrastructures are used to ensure enclave execution of FrontNet and protect the security of clients' input.

The key innovation of DeepEnclave is to partition DL models and make use of secure enclave techniques on cloud infrastructures to cryptographically protect the confidentiality and integrity of user inputs.

VoiceGuard [2] is an architecture which can efficiently protect the speech processing task inside a TEE. By using cryptographic techniques with hardware-enforced code and data isolation, VoiceGuard enables the secure processing of confidential data in a hostile environment.

VoiceGuard works in three stages (Figure 2). In the first stage, user U and vendor V reach an agreement on the code to be running in the enclave. In the second stage, secure channels are established with the enclave by U and V. In the third stage, speech processing can be performed securely in the enclave.

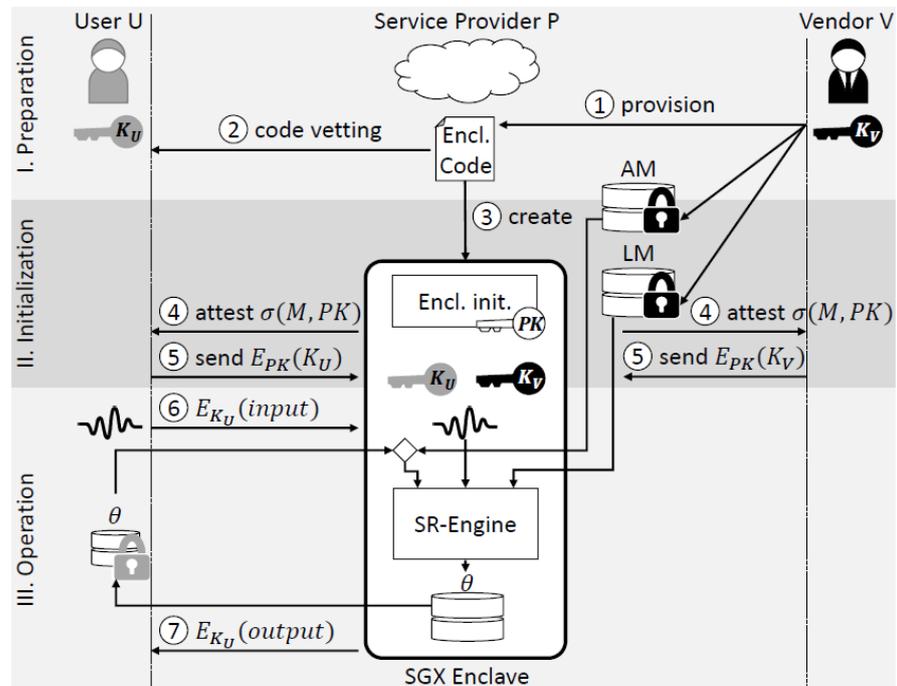


Figure 2. VoiceGuard architecture [2].

VoiceGuard is a novel architecture for privacy-preserving and efficient speech processing, it supports models for different users and can be deployed either in the cloud or on-premise. The demonstration showed that the applicability of VoiceGuard for speech recognition is in real time.

Although these solutions are all practical for the same goal, there are many differences between them. Slalom mainly focus on improving the

efficiency of evaluating a DNN in a TEE to guarantee the integrity and privacy of a ML computation. MLCapsule is a client side solution for off-line computation and it never leaves the client. PRIVADO is first system to against the leakage of enclave memory via data access pattern. The innovation of DeepEnclave is the partition of DL models and the use of secure enclave techniques for protecting users' inputs. While VoiceGuard is a good architecture only for privacy-preserving speech processing.

3.2 Ways for protecting the privacy of ML training data

As a good ML model requires a large amount of high-quantity datasets for training, the protection of training data is another important topic in ML field.

Sterling [11] is a data marketplace for private datasets. It can provide strong security and privacy guarantees for training data as well as ML models by combining blockchain smart contracts, TEEs and differential privacy. Data providers can set constraints to how their data could be used for and that will be enforced according to smart contracts. In addition, smart contracts allow data providers to define their reward and payments. The combination of privacy-preserving smart contracts and TEEs enables all the training data to be private while training the ML models and computing analytics.

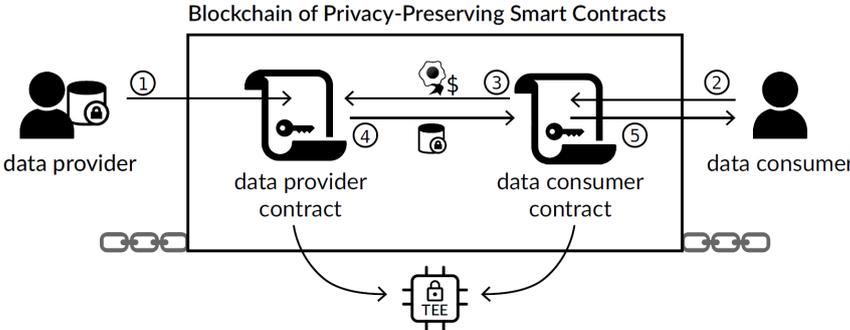


Figure 3. Interaction between data provider and consumers in Sterling marketplace [11].

The first step in the workflow (Figure 3) of Sterling marketplace is that a data provider U1 uploads his/her encrypted data to a centralized storage service and publishes a smart contract C1. Then, if a data consumer U2 is interested in the data, he/she will write a smart contract C2 which can match the constraints of C1. Next, C2 will be invoked and send by U2 with a signed request to attest its identity to U1, U1 verifies C2 and returns a

data decryption key. At last, U2 can use the decrypted data for training computation while U1 can get rewards according to the contrast.

Sterling keep ML models away from leaking their training data by automatic verification of differential privacy. Therefore, it can be well applied to applications like medical diagnosis and credit scoring which involve users' sensitive data.

A privacy-preserving multi-party ML system [14] are proposed based on trusted SGX processors for protecting the training data. In that system, an SGX-enabled data center is responsible for running a joint machine learning task training multiple parties' aggregate data. Although the processor can guarantee in some degree that only the machine learning code inside the enclave can access to the training data, it can not ensure the privacy of the training data. Therefore, the selection, adaptation as well as the implementation of ML algorithms is significant for preventing the leak of an amount of data caused by the exploitation of any side channels. To achieve this, data oblivious ML algorithms have been proposed which are based on new algorithmic techniques, deployment of platform specific hardware features and cautious elimination of data-dependent for support vector machines (SVM), decision trees, k-means clustering, matrix factorization and neural networks. The algorithms prevent exploitation of side channels induced by network accesses, memory and disk.

The combination of the system and algorithms can provide a strong privacy guarantees for the training data. Experiments showed that the system also has a good performance on realistic datasets.

Chiron [8] is a system that can make the data holders to train their ML models on an outsourced service which can prevent the exposure of their training data. The system is designed for privacy-preserving MLaaS, it protects the training data from the service operator then conceal the model structure and training algorithm from the user. In the same time, it only gives black-box access to the trained model.

Chiron makes use of the restriction provided by Ryoan [9] so that the service provider's code can have access to the user's data for training a model, in the meanwhile, prevent it from leaking the training data. One limitation of Chiron is inherited from Ryoan which is regarding covert and side channels.

CalTrain [5] is a TEE-based centralized multi-party collaborative learning system, it can achieve data confidentiality and model accountability for collaborative learning in the same time by using Intel SGX enclaves on

training machines. It ensures the confidentiality and integrity of training data.

The workflow of Caltrain consists of three stages which are training, fingerprinting as well as query. In the training stage, encrypted training data from data providers will be collected and a joint model will be trained on the training server. In the fingerprints stage, the latest trained model will be passed to exact the fingerprints for training examples. In query stage, fingerprints will be linked with their corresponding data providers, this can enable the model users to find the contributors of the "bad" training data which would cause runtime mispredictions by query interface.

The demonstration showed that Caltrain can effectively prevent leakage of private training data and break capacity and performance constraints of secure enclaves.

Myelin [10] is a DL framework designed for establishing a baseline level of performance for privacy machine learning. The demonstration through benchmarks on practical ML models showed the base performance of Myelin. It is a system for data-oblivious and efficient fully-private training of ML models in trusted hardware enclaves.

The system enables that the service provider's model is compiled automatically into a library which can provide high performance by Tensor Virtual Machine (TVM). Then, by using trusted hardware, the model will be privately trained on privately shared data. In addition, Myelin can be deployed on existing commodity hardware and brings convenience for continued exploration of the problem space and production applications. It also has a good performance in accuracy comparing to other related works.

For Comparison, Sterling can be well used in medical diagnosis applications, while Chiron is designed for privacy-preserving MLaaS. Myelin has a better performance in accuracy compare to Chiron [10], Caltrain breaks capacity and performance constraints of secure enclaves.

4 Future work of privacy-preserving machine Learning

Applications based on trusted hardware can efficiently solving private-preserving problems in ML. However, one of the common limitations of these applications is lack of GPU support because they fundamentally rely on hardware.

Myelin is proposed to secure the training process using differential pri-

vacy to achieve obliviousness guarantees. While PRIVADO focuses on supporting end-to-end inference-as-a-service for a given trained model. Comparing to PRIVADO, Myelin cannot execute inference for models that are not trained using the Myelin framework. Chiron does not prevent leakage via access patterns. Similarly, MLCapsule is susceptible to leakage of sensitive inputs via access patterns. Slalom also does not address access-pattern leakage.

Therefore, they are vulnerable to some malicious attacks. Reverse-engineering attacks [7] on convolutional neural network (CNN) models can exploit information leaks by timing side-channels through memory access patterns. The attacks are on a hardware accelerator and protected by secure processor techniques which are similar to the scheme used in Intel SGX. An adversary can reverse engineer the weights and the structure of an encrypted CNN model executing on a hardware accelerator. The main reason is that the CNN feature maps and weights are always too large to hold all of them in the on-chip memory of an accelerator. Thus, feature maps and weights are stored in off-chip memory by the accelerator. Memory access patterns will expose the accessibility of memory locations even if the data is encrypted. The attacks demonstrate that hiding off-chip memory access pattern is important for protecting confidential CNN models.

The future work of privacy-preserving machine Learning can focus on defense of malicious attacks from reverse engineering, membership inference. Distributed data and computation marketplace can be implemented to enable uncoordinated, autonomous execution of ML models on economically valuable, private data [10]. It is also significant to make a progress towards amplifying the value of heretofore unshareable data [11].

5 Conclusion

The paper surveyed up-to-date work in Privacy-preserving machine learning using trusted hardware (Intel SGX and Trusted Zone). In the prediction process, Slalom, MLCapsule, PRIVADO, DeepEnclave and VoiceGuard are summarized for protecting the confidentiality of ML models and clients' data. In the training process, Sterling, multi-party ML system, Chiron, Caltrain and Myelin are summarized for protecting the privacy of training data of the ML model. Some of these solutions still have some vulnerabilities addressed in Section 4. The future direction can force more on resistance to some malicious attacks.

References

- [1] Tiago Alves and Don Felton. Trustzone: Integrated hardware and software security-enabling trusted computing in embedded systems (july 2004), 2014.
- [2] Ferdinand Brasser, Tommaso Frassetto, Korbinian Riedhammer, Ahmad-Reza Sadeghi, Thomas Schneider, and Christian Weinert. Voiceguard: Secure and private speech processing. In *Annual Conference of the International Speech Communication Association (INTERSPEECH18)*, pages 1303–1307, 2018.
- [3] Zahra Ghodsi, Tianyu Gu, and Siddharth Garg. Safetynets: Verifiable execution of deep neural networks on an untrusted cloud. In *Advances in Neural Information Processing Systems*, pages 4672–4681, 2017.
- [4] Zhongshu Gu, Heqing Huang, Jialong Zhang, Dong Su, Ankita Lamba, Dimitrios Pendarakis, and Ian Molloy. Securing input data of deep learning inference systems via partitioned enclave execution. *arXiv preprint arXiv:1807.00969*, 2018.
- [5] Zhongshu Gu, Hani Jamjoom, Dong Su, Heqing Huang, Jialong Zhang, Tengfei Ma, Dimitrios Pendarakis, and Ian Molloy. Reaching data confidentiality and model accountability on the caltrain. *arXiv preprint arXiv:1812.03230*, 2018.
- [6] Lucjan Hanzlik, Yang Zhang, Kathrin Grosse, Ahmed Salem, Max Augustin, Michael Backes, and Mario Fritz. Mlcapsule: Guarded offline deployment of machine learning as a service. *arXiv preprint arXiv:1808.00590*, 2018.
- [7] Weizhe Hua, Zhiru Zhang, and G Edward Suh. Reverse engineering convolutional neural networks through side-channel information leaks. In *Proceedings of the 55th Annual Design Automation Conference*, page 4. ACM, 2018.
- [8] Tyler Hunt, Congzheng Song, Reza Shokri, Vitaly Shmatikov, and Emmett Witchel. Chiron: Privacy-preserving machine learning as a service. *arXiv preprint arXiv:1803.05961*, 2018.
- [9] Tyler Hunt, Zhiting Zhu, Yuanzhong Xu, Simon Peter, and Emmett Witchel. Ryoan: A distributed sandbox for untrusted computation on secret data. *ACM Transactions on Computer Systems (TOCS)*, 35(4):13, 2018.
- [10] Nick Hynes, Raymond Cheng, and Dawn Song. Efficient deep learning on multi-source private data. *arXiv preprint arXiv:1807.06689*, 2018.
- [11] Nick Hynes, David Dao, David Yan, Raymond Cheng, and Dawn Song. A demonstration of sterling: a privacy-preserving data marketplace. *Proceedings of the VLDB Endowment*, 11(12):2086–2089, 2018.
- [12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [13] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Inference attacks against collaborative learning. *arXiv preprint arXiv:1805.04049*, 2018.

- [14] Olga Ohrimenko, Felix Schuster, Cédric Fournet, Aastha Mehta, Sebastian Nowozin, Kapil Vaswani, and Manuel Costa. Oblivious multi-party machine learning on trusted processors. In *USENIX Security Symposium*, pages 619–636, 2016.
- [15] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016.
- [16] Nelly Porter, Jason Garms, and Sergey Simakov. Introducing asylo: An open-source framework for confidential computing, 2018.
- [17] Mohamed Sabt, Mohammed Achemlal, and Abdelmadjid Bouabdallah. Trusted execution environment: what it is, and what it is not. In *2015 IEEE Trust-com/BigDataSE/ISPA*, volume 1, pages 57–64. IEEE, 2015.
- [18] Shweta Shinde, Zheng Leong Chua, Viswesh Narayanan, and Prateek Saxena. Preventing page faults from telling your secrets. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 317–328. ACM, 2016.
- [19] Shruti Tople, Karan Grover, Shweta Shinde, Ranjita Bhagwan, and Ramachandran Ramjee. Privado: Practical and secure dnn inference. *arXiv preprint arXiv:1810.00602*, 2018.
- [20] Florian Tramèr and Dan Boneh. Slalom: Fast, verifiable and private execution of neural networks in trusted hardware. *arXiv preprint arXiv:1806.03287*, 2018.
- [21] Amit Vasudevan, Jonathan M McCune, and James Newsome. *Trustworthy execution on mobile devices*. Springer, 2014.

Beyond the Google's BeyondCorp

Lukas Klein

lukas.klein@aalto.fi

Tutor: Mohit Sethi

Abstract

Traditionally, companies rely on the concept of perimeter security to secure their networks. In perimeter security, most security measures that are taken enforce separation between the internal network and the public internet. This concept has been proven problematic because it is prone to insider attacks. Various countermeasures have been proposed to cope with insider threats and to maintain the security of enterprise networks. One of those proposed countermeasures is BeyondCorp. In this paper, we look at BeyondCorp in detail and understand its applicability and migration requirements.

KEYWORDS: *beyondcorp, perimeter security*

1 Introduction

Traditionally, companies keep their internal network shielded from the public internet. This is called perimeter security [8]. In perimeter security, most security measures that are taken enforce separation between the internal network and the public internet. Access to enterprise services is only granted while employees are within the company facilities. This approach might work for smaller businesses; however, it is rather limiting. Nowadays, it is common practice to allow employees to work from home or any other place they want. For this, access to internal services might be required. This applies to global companies in particular. Not only do they need to exchange information between different company locations but also do they rely on a mobile workforce that needs access to internal services and resources. Typically, Virtual Private Network (VPN) connections are used to establish these connections to internal resources. However, with VPN connections, the employees bypass the perimeter security that has been built around the network. The employees are now part of a network that implicitly trusts them to not act maliciously. Research has shown that insider threats are a critical issue and that this complete trust should not be granted [9, 12]. Various countermeasures have been proposed to cope with insider threats and to maintain the security of enterprise networks. One of those proposed countermeasures is BeyondCorp. BeyondCorp aims to remove the implicit trust from the internal network infrastructure.

In this paper, we look at BeyondCorp in detail and understand its applicability and migration requirements. The paper is organized as follows. Section 2 is giving an introduction to the concept of BeyondCorp. Section 3 elaborates on the benefits of a migration to BeyondCorp. Section 4 elaborates which requirements have to be fulfilled for a successful migration. Section 5 lays out a potential strategy for the migration.

2 BeyondCorp Overview

BeyondCorp aims to remove all implicit trust from the enterprise network [16, 13]. This is achieved by externalizing the entire internal network infrastructure and by making it available from the public internet. One key component of BeyondCorp is an Access Proxy (AP). Similar to the reference monitor in an operating system that enforces user privileges over files and directories, the AP enforces user privileges over the services and resources of the enterprise. In order to act as a reference monitor, the AP must be positioned between the user and the back-end service the user wants to interact with. When users initiate a connection with an enterprise service, they are not directly communicating with the service. Instead, the AP intercepts the connection and acts as a Man-in-the-Middle (MitM). Once the AP decides that access should be granted to the desired service, it forwards the traffic to the back-end. Figure 1 illustrates the data flows. In its MitM position, the AP could also provide load balancing and protection against Distributed Denial of Service (DDoS) attacks.

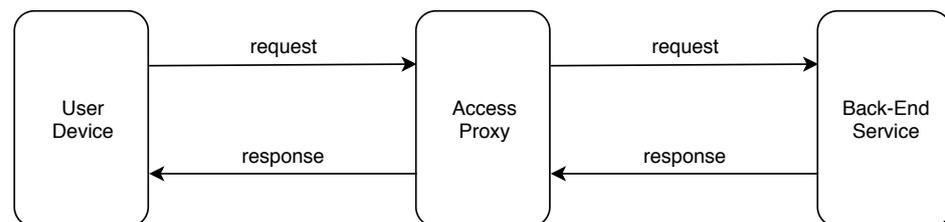


Figure 1. Data flow in BeyondCorp

In order to position the AP, BeyondCorp makes use of the canonical name (CNAME) record of the Domain Name System (DNS) protocol. Instead of the Internet Protocol (IP) addressing the services, the enterprise services have CNAME records that point to the domain name of the access proxy. If a user initiates a connection to an enterprise service, it, therefore, resolves to the IP address of the AP. BeyondCorp implements access control on a per-device level. While developers should have access to the bug tracker with their work device, they should not be able to access it with their personal device(s). This per-device access control, therefore, requires a form of device authentication. Devices that initiate a connection to company resources need a way to prove their identity to the AP. For this purpose, BeyondCorp uses the Transport Layer Security (TLS) protocol and its client authentication feature [11, 2]. Every device has its own

asymmetric key pair and a certificate that belongs to it. Once a connection to an enterprise service is initiated, the AP verifies that the certificate is valid and that the device has access to its private key. Further, the AP verifies that the device should have access to the desired enterprise service. If all those criteria are met, access is granted. Since the proof of identity is based on the private key, it is crucial to the concept of device authentication that the private key is not accessible by any means. This is achieved by storing it in a Trusted Platform Module (TPM).

While device authentication is sufficient for stationary devices, like servers communicating with each other, it is not sufficient for portable user devices, like laptops or mobile phones. The mobility of those devices makes them prone to theft and loss [14, 15]. Access to company resources should not be granted to a stolen device. Company resources should not be accessible from stolen devices. BeyondCorp prevents this by enforcing user authentication, in addition to device authentication, on mobile devices. More precisely, a Single-Sign-On (SSO) system handles user authentication by requesting login credentials. Security can be further enhanced through the complementary use of Two Factor Authentication (2FA).

Before users enter their login credentials, the authenticity of the server should be verified. Modern browsers perform this verification automatically and refuse a connection if the server cannot be authenticated. The browsers ensure that the server's certificate is valid and belongs to the requested domain. Since the AP acts as a MitM, it impersonates all back-end services that are requested. Considering this, the AP needs access to the certificates and private keys for all back-end service domains. Instead of using different certificates, it is reasonable to use one single certificate that matches all back-end domains. This can be achieved by using subdomain wildcards for the certificates.

So far, we have discussed the connection between user devices and the AP. However, even though the AP and the back-end service are operated by the company, the connection between those two components should not be unprotected. The Snowden revelations have shown that state adversaries even tap communication between enterprise back-end components [7]. From those revelations, we conclude that the connection between AP and back-end service should be mutually authenticated and encrypted. BeyondCorp does not specify a protocol that should be used. However, it highlights that the security of the connection is crucial to the proposed

security concept. TLS with its client and server authentication features is one protocol that provides the required security. It might come with a certain overhead at the same time.

3 Advantages

BeyondCorp relies on an SSO system for access control management. Therefore the advantages of SSO systems also apply to BeyondCorp. Those include advantages for all parties involved. Users do not need to remember credentials for every different service. Instead, it is sufficient to remember the credentials to the SSO. This centralized access control has the additional benefit that user credentials are not spread all over the internal network infrastructure. Therefore, it is easy to disable access to the network if an employee leaves the company. Without a centralized access control, it would be necessary to locate all accounts on the different services and to disable them separately. The major benefit for service developers is that they do not have to implement access control on their services. Implementing access control and keeping it secure is major overhead service developers do not have to deal with in BeyondCorp.

However, this does not imply that service developers cannot require more fine-grained access control. In fact, BeyondCorp provides a sustainable and efficient way to implement this. Since the access proxy acts as an HTTP proxy, it can add additional information in the form of HTTP headers to incoming requests from employee devices. The back-end service can then use this information for additional more fine-grained access control. BeyondCorp mentions a trust level as an example. The trust level can depend on information as the type of the device or the software version used. Employees might have a lower trust level when they connect from their mobile phones or if they use deprecated software.

Snowden revealed that state adversaries tap even the communication between parts of the internal network structure. Those revelations placed the security of internal communication in the focus. In fact, when the company infrastructure or parts of it have been migrated to the cloud, it is inevitable that communication is routed through the public internet. In order to secure the communication against tampering and eavesdropping, BeyondCorp enforces authentication and confidentiality of the communication.

4 Migration Requirements

The concept of device identification and authentication is crucial to BeyondCorp. However, secure device authentication can only be achieved, if the device's private key is not accessible by any means. This requires the devices to have TPMs or similar cryptographic hardware. In order to keep the private keys inaccessible to the user, the device is required to be managed by the company. This contradicts the concept of Bring Your Own Device (BYOD) which gained popularity during recent years [4].

Due to its MitM position, the access proxy needs to process all the traffic that users produce. A few things should be considered to ensure that the AP does not turn into a bottleneck. Since BeyondCorp is about securing company networks and companies do not require their employees to work all day, the AP probably does not have to cope with a constant load. Instead, there probably is phases of low load, during the night, and phases of high load, during the day. It's crucial that the AP can cope with the high load phases. If the AP goes down, no communication with enterprise infrastructure is possible. Another issue that could cause latency spikes is that the AP has to maintain one TLS connection between the user and the AP and another one between the AP and the back-end service.

BeyondCorp was designed with an HTTP-based cloud infrastructure in mind. While BeyondCorp architecture might work well with most browser-based traffic, there is some programs and protocols that do not work without workarounds. The following issues have been discussed in [13, 1]:

Some companies might use legacy software or console applications that communicate over HTTP, but that do not support TLS. However, the device authentication of BeyondCorp requires TLS. In fact, all non-authenticated requests are denied by the Access Proxy. This issue can be tackled by using a local proxy server on the client device. Figure 2 illustrates the dataflow with such a local proxy. Once a naive HTTP client initiates a connection to a company server, the local proxy intercepts it. The local proxy then establishes a TLS connection with the Access Proxy based on the client device certificate. The local proxy then sends the intercepted HTTP request over TLS to the Access Proxy.

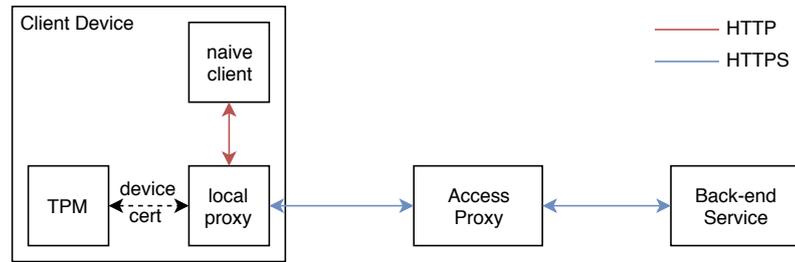


Figure 2. Data flow for local naive HTTP clients

The Access Proxy in BeyondCorp has been designed to enforce device and user authentication on HTTPS traffic. The previous paragraph elaborated client-based adjustments that enable the use of naive HTTP applications. Most companies do not solely rely on HTTP-based traffic. Another widely-used protocol is Secure Shell (SSH). Without additional adjustments, the SSH protocol will not work with BeyondCorp. This is due to two properties of the protocol. First, the SSH protocol is not based on the HTTP protocol. The Access Proxy, however, only supports HTTP-based traffic. Secondly, SSH requires an end-to-end connection. BeyondCorp proposes the use of HTTP tunnelling to make all TCP-based protocols compliant with the Access Proxy. Since HTTP tunnelling happens on the TCP layer, the end-to-end property of SSH is trivially met.

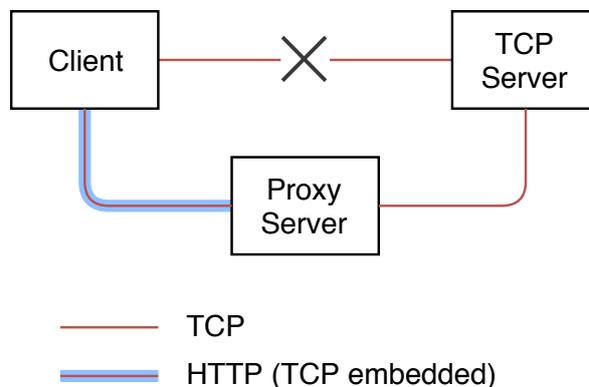


Figure 3. HTTP tunnelling

Figure 3 illustrates the general problem that is solved by HTTP Tunnelling. A client device wants to initiate a connection to an arbitrary TCP server. Due to some network restrictions, the client can only initiate connections to the HTTP(S) standard ports 80 and 443. This is, for example, enforced in some public WiFi hotspots. The client can request an HTTP proxy to open the TCP connection on its behalf. This is typically done via the HTTP CONNECT method [5]. Once the proxy server has established the connection with the TCP server, the client is notified with an HTTP response. The client then wraps all TCP packets into HTTP re-

quests and forwards them to the proxy server. The proxy server removes the HTTP layer and forwards the TCP traffic to its intended destination. Once the proxy server receives TCP packets from the destination server, it wraps those into an HTTP response and returns it back to the client. The connection is kept alive until either the client or the destination server terminates it. Since the proxy server establishes a raw TCP connection, HTTP tunnelling can be used for all TCP-based protocols.

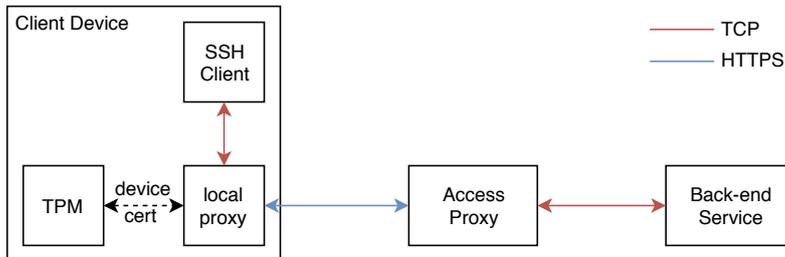


Figure 4. Data flow for end-to-end protocols

The general problem HTTP tunnelling solves is similar to the problem in BeyondCorp. In BeyondCorp, applications on the client device cannot establish non-HTTP connections with a back-end service. However, there is a proxy server that could open the non-HTTP connection on the client's behalf, namely the Access Proxy. Figure 4 illustrates this with SSH as a non-HTTP example protocol. Once an SSH client initiates a TCP connection with a back-end service, a local proxy intercepts it. This local proxy performs an HTTP CONNECT request to the Access Proxy to open a TCP connection on the client's behalf. However, instead of using raw HTTP, the local proxy uses HTTPS based on the client certificate of the device. Through this, the Access Proxy can enforce access control in the same way as it enforces access control in regular HTTPS traffic. If the Access Proxy grants request to the back-end service it initiates the TCP connection and performs the HTTP tunnelling described above.

While HTTP tunnelling technically enables all TCP-based traffic in BeyondCorp, it has a significant downside. Through the tunnelling, the latency drastically increases. This fact renders tunnelling inappropriate for latency-sensitive traffic. BeyondCorp proposes a solution similar to the Kerberos [?] protocol. In this solution, an Authentication Server is introduced as part of the Access Proxy. Instead of intercepting the traffic like the Access Proxy, the Authentication Server is consulted before the latency-sensitive connection is initiated. The Authentication Server then returns some kind of access token to the back-end service. This access to-

ken can be used by the client to authenticate to the back-end server. Unfortunately, the protocol was not explained in detail. Nevertheless, several major downsides can be identified. First, symmetric or asymmetric cryptography between the Authentication Server and the back-end service is required in order to validate the access tokens. Secondly, the server needs a DNS entry that is not intercepted by the Access Proxy. If the back-end service uses the Access Proxy for some applications, a second DNS entry has to be created. Thirdly, the protocol has to be adjusted. Additional software or modifications in the source code of the applications are required to use the Authentication Server.

5 Migration Strategy

In traditional enterprise networks, there are usually two common methods for employees to connect to the enterprise network. While they are within the company facilities, they can connect to the WiFi network of the company. If they are outside of the company facilities, they can use a VPN client to connect to the internal network. Through both methods, employees have access to internal company resources. It is the goal of BeyondCorp to restrict this internal access to the company resources. However, internal access should not be disabled abruptly in order to maintain the productivity of the company. Instead, the migration process should incrementally move employee devices from the privileged network to a virtual network that rejects all internal access to company resources.

Before users are migrated to the unprivileged networks, the services first have to implement BeyondCorp support. In order to do so, the back-end service first needs a domain name. Instead of directly pointing to the back-end server, the domain name has a CNAME entry that points to the domain name of the Access Proxy. Once the DNS is configured, the connection between the Access Proxy and the back-end has to be secured. This is done by setting up TLS with mutual authentication.

Not only the services but also the clients have to implement BeyondCorp support. For clients, this means that device certificates have to be installed on all employee devices. Those device certificates are later used to authenticate the employee devices to the Access Proxy. Furthermore, the client-based local proxy, that has been explained in the previous section, is installed on the employee devices.

In order to automate the migration process, it is necessary to distinguish between users that are ready for the migration and users that have non-compliant workflows. BeyondCorp proposes two tools to achieve this [10, 3, 6]: a client-based network simulator and a log analyzer. The client-based network simulator is installed on the employee devices. It is aware of which back-end services implemented BeyondCorp support. The network simulator has two modes of operation. In the first mode, it analyzes all traffic on the employee device. If there is traffic to internal resources, the simulator applies its information about BeyondCorp-compliant back-end services to the traffic. If a non-compliant workflow has been found, it is logged to a central repository. The log entry contains the IP of the client to identify the user and the IP of the back-end service to identify the service. In the second mode, the network simulator actively rejects all internal traffic to BeyondCorp-compliant back-end services. If the network analyzer finds internal traffic to non-compliant services, the workflow is logged to the central repository.

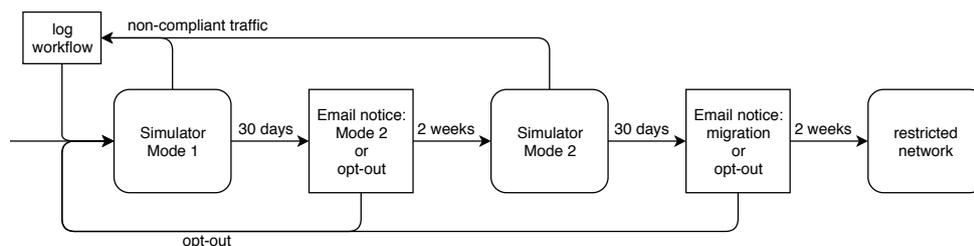


Figure 5. Migration Process

The central logging repository is evaluated by the second tool, the log analyzer. The log analyzer continuously identifies non-compliant services from the log. In addition to that, the log analyzer controls the different stages of the migration process. The different stages are illustrated in figure 5. Initially, the network simulator runs in mode one on all employee devices. If there hasn't been non-compliant traffic for 30 consecutive days, the employee receives an email notification. The email informs the user that stage two is triggered in two weeks unless the user opts out. If a user opts out, the entire migration process is restarted. After 30 days in mode two, the user again receives an email notification with an opt-out link. If the user does not opt out within two weeks the user is assigned to the unprivileged network and the migration is complete.

6 Conclusion

Research has shown that the commonly-applied paradigm of perimeter security is prone to insider attacks. Furthermore, the perimeter security does not work well with cloud computing. BeyondCorp suggests an alternative to perimeter security. In BeyondCorp, all enterprise services are externalized. More precisely, access to enterprise services is granted through the public-facing internet. In order to maintain and improve security, BeyondCorp introduces an Access Proxy. This Access Proxy is placed between the employee device and the back-end service and enforces the access control. A Single-Sign-On system is used to handle user authentication. In addition to that, BeyondCorp introduces device authentication. Device authentication enables more fine-grained access control management; however, it requires Trusted Platform Modules or similar cryptographic hardware. Therefore, managed company devices have to be used in order to access company resources. The Bring-Your-Own-Device paradigm is not supported.

While BeyondCorp was designed to natively support browser-based traffic, only minor adjustments are required to a majority of TCP-based protocols. Namely, a local proxy that runs on the client device can make the majority of TCP-based traffic comply with BeyondCorp. However, BeyondCorp does not propose an applicable solution for UDP-based or real-time protocols. In its proposed form BeyondCorp is not suitable for companies that rely on real-time traffic or UDP protocols.

The proposed migration strategy was designed to be fully automatable. At the same time, productivity is maintained. This is achieved by an incremental migration that identifies employees with unproblematic workflows. Those users are then automatically migrated. Problematic workflows are logged to a central repository. The logs contain the IP address of the user and the IP address of the service. This information can then be used to analyze and resolve non-compliant workflows. Employees that are eligible for the migration have the opportunity to delay it until the migration is complete.

References

- [1] Betsy (Adrienne Elizabeth) Beyer, Colin McCormick Beske, Jeff Peck, and Max Saltonstall. Migrating to beyondcorp: Maintaining productivity while improving security. *Login*, Summer 2017, VOL 42, No 2, 2017. ISSN 1044-6397.
- [2] T. Dierks and E. Rescorla. The transport layer security (tls) protocol version 1.2. RFC 5246, RFC Editor, August 2008.
- [3] Victor Manuel Escobedo, Filip Zyzniewski, Betsy (Adrienne Elizabeth) Beyer, and Max Saltonstall. Beyondcorp: The user experience. *Login*, tbd:tbd, 2017.
- [4] M. Eslahi, M. V. Naseri, H. Hashim, N. M. Tahir, and E. H. M. Saad. Byod: Current state and security challenges. In *2014 IEEE Symposium on Computer Applications and Industrial Electronics (ISCAIE)*, pages 189–192, April 2014.
- [5] R. Fielding and J. Reschke. Hypertext transfer protocol (http/1.1): Semantics and content. RFC 7231, RFC Editor, June 2014.
- [6] Michael Janosko, Hunter King, Betsy (Adrienne Elizabeth) Beyer, and Max Saltonstall. Beyondcorp 6: Building a healthy fleet. *login.*, 43, 2018.
- [7] Newton Lee. *The Afterlife of Total Information Awareness and Edward Snowden's NSA Leaks*, pages 151–182. Springer International Publishing, Cham, 2015.
- [8] Stephen Northcutt, Scott Winters, Karen Frederick, Lenny Zeltser, and Ronald W. Ritchey. *Inside Network Perimeter Security: The Definitive Guide to Firewalls, VPNs, Routers, and Intrusion Detection Systems*. Pearson Education, 2002.
- [9] J. R. C. Nurse, O. Buckley, P. A. Legg, M. Goldsmith, S. Creese, G. R. T. Wright, and M. Whitty. Understanding insider threat: A framework for characterising attacks. In *2014 IEEE Security and Privacy Workshops*, pages 214–228, May 2014.
- [10] Barclay Osborn, Justin McWilliams, Betsy Beyer, and Max Saltonstall. Beyondcorp: Design to deployment at google. *login.*, 41:28–34, 2016.
- [11] E. Rescorla. The transport layer security (tls) protocol version 1.3. RFC 8446, RFC Editor, August 2018.
- [12] Eric D Shaw, Keven G Ruby, and Jerrold M Post. The insider threat to information systems. *Security Awareness Bulletin*, 2(98):1–10, 1998.
- [13] Batz Spear, Betsy (Adrienne Elizabeth) Beyer, Luca Cittadini, and Max Saltonstall. Beyond corp: The access proxy. *Login*, 2016.
- [14] Z. Tu and Y. Yuan. Understanding user's behaviors in coping with security threat of mobile devices loss and theft. In *2012 45th Hawaii International Conference on System Sciences*, pages 1393–1402, Jan 2012.
- [15] Zhiling Tu, Ofir Turel, Yufei Yuan, and Norm Archer. Learning to cope with information security risks regarding mobile device loss or theft: An empirical examination. *52(4):506 – 517*, 2015.

[16] Rory Ward and Betsy Beyer. Beyondcorp: A new approach to enterprise security. *login.*, Vol. 39, No. 6:6–11, 2014.

Service mesh concept and its realizations

Oscar Stigzelius

oscar.stigzelius@aalto.fi

Tutor: Hannu Flinck

Abstract

The era of microservices has evolved into putting services into the cloud. This means that communication happens in the network, which is much more complex and less reliable because of the changing service and traffic mixes between the end point. The network configuration must match the very short life-cycles of microservices. The end-user needs to find service endpoints, have the same API versions and handle messages. End-users also need to monitor, handle call executions by recognizing errors, and retrying failed API calls when necessary. They also need to know the identity of the calling service, and messages must be encrypted and services must follow access control requirements.

Not all of this is new to developers, but what is new is the rapid increase in containers and the number of service calls they are making. These new challenges make network communication very complex. Today, the best way to handle such complexity is to use a service mesh.

KEYWORDS: *Service mesh, Istio, Linkerd, Consul, Envoy, Kubernetes, Data plane, Control plane*

1 Introduction

To truly understand the idea behind service meshes we have to dive into the microservices architectural model and what problems come with it.

The microservices architectural model is very popular today. More and more companies are moving towards distributed microservice systems. Companies that are using microservices today are e.g. Uber, Netflix, Amazon, eBay, Twitter, PayPal, Sound Cloud, and Zalando. [18, 8]

Microservice architecture is described as a style to develop a single application by creating small independent services with specific tasks that communicate, with each other, through lightweight mechanisms, such as HTTP resource API. [10, 8]

Why companies have decided to use microservices are because a lot of benefits do come from a microservice architectural model. One of the main benefits is that it saves a lot of time if you have a growing and changing application. Another benefit is that it gives developers the freedom to choose what technology he wishes to use to implement a service. [8, 19]

Sam Newman has defined seven principals that should be followed to create a project based on the microservice architectural style [14]. Newman's principles can be summarized as follows:

1. **Model Around Business Concepts**, shape your project around your business, and not around technical concepts.
2. **Adopt a Culture of Automation**, automating your project, with automated testing and deployment strategies, will speed up the delivery of your business.
3. **Hide Internal Implementation Details**, Focus on the business logic in your project, not thinking of e.g. communication problems between services, that should be handled by an API or library.
4. **Decentralize All the Things**, allow teams to have full control of their services, meaning no one else then the team itself is needed to e.g. make changes and deploy the changes to their service.

5. **Independently Deployable**, a service should be able to be deployed without causing problems to other services or to millions of users.
6. **Isolate Failure**, expect that failure can happen anywhere and everywhere, don't trust the network. A service should be able to handle failures.
7. **Highly Observable**, have a great monitoring system for your project so it will be easy to find problems in your services.

By following these principals, a project based on the microservice architectural style should work flawlessly. But some of these principals can be hard to implement due to their system wide implications that make them hard/complex to follow. The assumption behind microservices is that developers can handle problems regarding distributed systems. One difficulty is that developers must know how to handle fault tolerance, network latency, different message formats, and load balancing. In addition, developers must implement a service discovery mechanism which enables services to discover the locations, meaning hosts and ports, of other services that a service wants to communicate with. [15, 19, 8]

By implementing a service mesh into your microservice project, the principals 1, 3, 4, 6 and 7 will be much easier to handle. In addition, it will fix the above-mentioned problems.

In this paper, we are going to look deeper into what a service mesh is and what part it plays in a microservice architectural model, what problems a service mesh fixes and what problems come with it. We will also look at some service mesh software/platforms and how good they fill up to the service mesh concept.

2 Service mesh - what is it?

A service mesh is described by Lee Calcote in the following way "Service meshes provide policy-based networking for microservices describing desired behavior of the network in the face of constantly changing conditions and network topology." [19]

The era of microservices has evolved into putting services into the cloud.

This means that communication happens in the network, which is much more complex and less reliable than communicating locally. As Tobias Kunze says it very well, "What used to 'just work' now needs to be spelled out explicitly, for every client and every service". The end-user needs to find service endpoints, have the same API versions and handle messages. End-users also need to monitor, handle call executions by recognizing errors, and retrying failed API calls when necessary. They also need to know the identity of the calling service, and messages must be encrypted and services must follow access control requirements. [13]

Not all of this is new to developers, but what is new is the rapid increase in containers and the number of service calls they are making. These new challenges make network communication very complex. Today, the best way to handle such complexity is to use a service mesh. [13]

A service mesh is a low-latency infrastructure layer that can be configured and designed to handle a vast amount of network-communication-processes among services in an application, using APIs. A service mesh makes sure that communication between services is secure, fast and reliable. Besides the communication between services, the service mesh provides service discovery, encryption, load balancing, traceability, observability, authorization and authentication, and support for circuit breakers. [21]

A service mesh consists of a data plane and a control plane, otherwise, it will not be a service mesh. But what do all these terms mean and what do they do? Let's go through the parts of a service mesh, one by one.

2.1 Service Mesh Terminologies

To give to the reader a better understanding of the different parts of a service mesh, we need to introduce the common service mesh concepts and related terminology.

2.1.1 Container

Docker gives a good explanation about what a container is from their homepage, "A container is a standard unit of software that packages up the code and all its dependencies so the application runs quickly and reliably from one computing environment to another." [9] By putting a service

inside a container, you can get that service to function on any computational environment. A container consists of everything that a service need, to be run.

There are some differences between a container and a virtual machine. One is that containers are more lightweight than virtual machines, another one is that the startup time of an container is much faster than starting up a virtual machine, every virtual machine has its own operating system, while containers share the operating system. [7]

2.1.2 Container orchestration

Containers have become a very popular way of deploying your application. Because of its wide adoption, some common patterns have emerged while developing a containerized application. These are scheduling instances of containers, cluster management, and orchestration. [2] Container orchestration is a software that makes it easy for clients to handle their containerized applications. It handles scheduling, cluster management, orchestration and security issues regarding containers. A service mesh is a technology, which makes communication between microservices easy and manageable. Microservices are usually put in containers and that is why a container orchestration is needed in a service mesh. The most common container orchestration engine is Kubernetes.

2.1.3 Pod

To understand what a pod is, it is easier to tell what a pod consists of. A pod consists of one or multiple containers that all share the same network and storage. Usually, one pod consist of one service. All the containers of a pod have the same IP address and port space. Containers in the same pod can communicate directly with each other through localhost, whereas containers in different pods must communicate through the internet via e.g. APIs'. [5] Pods are units of distribution that the orchestrator software acts on.

2.1.4 Sidecar

A sidecar is an application that runs alongside with another application, inside the same pod as the application is in [16]. From a service mesh perspective, every pod has its own sidecar running alongside with the service(s) inside itself. The purpose of the sidecar, in a service mesh, is to handle all the incoming and outgoing communication to and from the service it serves. This communication is often done over the internet and that is why the sidecar can be called a "Sidecar proxy". A sidecar is one architectural type for a service mesh to handle service-to-service communication. [11]

2.1.5 Node

A node is a place where pods will be placed in. A node can be a physical machine (such as a computer or server) or it can be a virtual machine. A node contains information about its own condition, how many pods can be run inside it, how much CPU and memory is used by the node etc. Nodes can be handled by a container orchestrational software. Kubernetes uses a "Node controller" to control and keep track of nodes.

2.2 Data plane

The data plane is one of the main parts of a service mesh. The data plane is responsible for service discovery, health checking, routing, authentication and authorization, load balancing and observability. In other words, the data plane is the place where all the network traffic is handled for a service instance. [12]

Next, we will go through different types of data planes and explain the most important parts of a data plane.

2.2.1 Data plane types

There are different types of data planes, three types according to Andrew Jenkins. The easiest one to implement is by using library inside every application. These libraries consist of features that a data plane should have. Ribbon or Hystrix, which are made by Netflix, are such libraries.

Another type of data plane architecture is using a node agent. This means that every node has its own "agent", which is responsible for delivering communication from one service to another. This kind of type is used by many service mesh technologies such as Linkerd version 1.

The newest, and the most promising data plane architectural style [1], is sidecar proxies. A sidecar proxy is located inside every pod, in its own container alongside the pod's services. The sidecar proxy is responsible for all outgoing and incoming communication to the pod and its services. This type is used by the current "leader" of service mesh technology, Istio, and by Conduit. [11]

2.2.2 Service discovery

When an instance of a service wants to communicate with another service, the service tries to find an available instance of the other service. This is possible because a container orchestration framework keeps track of service instances that are available. This means that a service, which is communicating with a sidecar, which is managed by the container orchestration framework, can get the information, from the container orchestration framework, about which service instances are available. [21]

2.2.3 Health checking

From a data plane perspective, health checking means checking the health of containers in a containerized application. When using the sidecar architectural type of a data plane, it means that every sidecar is responsible for checking "the health" of the containers that run in the same pod as itself. A health check is as simple as creating a request to the containers inside the pod, periodically, and then waiting for a response back from the container. If the sidecar doesn't get any response, it determines that the container is "not healthy", and will not forward further request to that container. This means of course that a health checking system is needed to be implemented to the services in your project. Kubernetes provide an easy way to implement health checking for your containers. [2, 6]

Circuit breaking is a common term when talking about health checking and service meshes. Health checking is not anything new, but there's a problem that if you have a long set of API calls after each other, and then the last API call is to a service that is "unhealthy" then the call will not be made, and all the API calls before have been unnecessary. This happens again and again if the same first API call is made. Circuit breakers stop this kind of behavior by making a service also "unhealthy" if that service should make an API call but can't because the service it trying to connect to is "unhealthy". In this way all the services, that is in the same line of API calls, get "unhealthy" one by one.

2.2.4 Observability

The word monitoring is used in many different ways nowadays, and have different meanings in different scenarios. Observability is a word for describing all the meanings of monitoring together. How it differs from monitoring, is that observability tells you "why something isn't working", while monitoring tells you "if something is working". It is more about understanding the underlying problem. [2]

2.3 Control plane

The main function of control plane of a service mesh is to configure and set policies for the data plane. It is the software that makes it easy for humans to interact with the data plane. The control plane has some sort of UI or CLI that the user can interact with to configure and set policies to the data plane.

The control plane doesn't directly interfere with the communication between services, that is the data planes task, but the control plane decides what configurations, such as authentication and authorization settings for service communication, load balancer settings, etc. which get pushed to the data plane that handles the communication. A good explanation of the control plane is as Matt Klein puts it, "The control plane turns all of the data planes into a distributed system". [12]

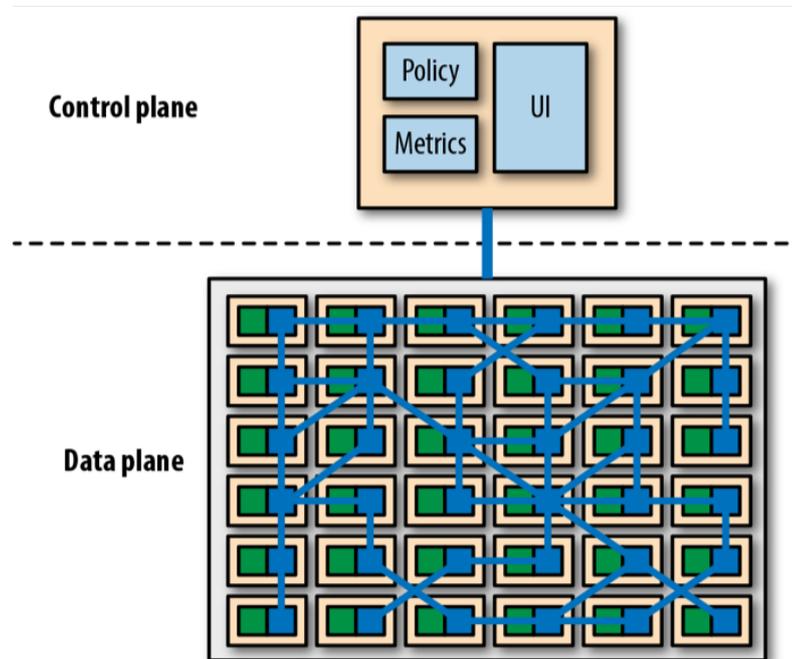


Figure 1. This picture is taken from <https://eksworkshop.com/servicemesh/> and it represents the data plane and the control plane in a service mesh. The data plane in this picture uses the data plane architectural type "Sidecars"

3 State of the art - service meshes

The word "service mesh" has gained a lot of popularity in the last years. Because of the popularity, a lot of different service meshes has emerged, such as AWS app mesh, Tetrade, Istio, Linkerd, and Consul. In this section, we will talk about Kubernetes and the most common service meshes

today.

3.1 Kubernetes

Kubernetes is an open-sourced project that the Cloud Native Computing Foundation (CNCF) is hosting. It is best described by Kubernetes own website, "Kubernetes is an open source container orchestration engine for automating deployment, scaling, and management of containerized applications." [4]. Furthermore, Kubernetes handles security, load balancing and many complex problems that come with containerized applications. [17] Kubernetes plays a big role in service meshes as it handles the main problems of containerized applications. A service mesh uses Kubernetes under the hood to control and get data from microservices, which are inside containers.

3.2 Linkerd

Linkerd is the original service mesh created by Buoyant and are a project by the Cloud Native Computing Foundation. The first appearance of Linkerd was in 2016. Currently, there are two versions of Linkerd, version 1 and version 2.

3.2.1 Linkerd version 1

The first version uses node agents as its data plane type. This version of Linkerd works with different container orchestration engines. This version was written in Scala, which comes with a 110mb of footprint, which is fine for a data plane that is using node agents. But if Linkerd would like to switch to use sidecar proxies instead of node agents, then the footprint would be inside every pod, and not only every node, which would definitely affect the performance on the application. This is why Linkerd version 2 was created. [1, 20]

3.2.2 Linkerd version 2

Linkerd version 2 is written in Rust and Go and is therefore lightweight and doesn't affect the performance of the application as much as Linkerd version 1. Linkerd version 2 works only with Kubernetes and it uses sidecar proxies as a data plane type. At the beginning of the development of the Linkerd version 2, the service mesh was called "Conduit". Later on, when the project showed great potential for future use, they switched from "Conduit" to "Linkerd 2".

The control plane of Linkerd is three components. One which configures all the settings for the data plane, one web component for the dashboard and one metrics component which handles and visualizes data. The data plane of Linkerd is Linkerd's own made proxy.

3.3 Envoy

Envoy is only a service mesh data plane created by Lyft. Together with a control plane, they represent a service mesh. Envoy is also a project of the Cloud Native Computing Foundation.

Envoy is written in C++11 which makes Envoy lightweight and has a high-performance rate. It also works with any programming language, which makes it user-friendly for the developers. It is the sidecar in a data plane and its main task is to make the network transparent for the application. Envoy handles all the things that were described in the data planes section. [3]

3.4 Istio

Istio is currently the most popular service mesh with many features. [1] The making of Istio started in May 2017, by Google, Lyft, and IBM.

Istio consists of 2 components, the data plane, and the control plane. The data plane can be split into two components, Envoy proxies and "Mixer".

The Envoy proxies are the above mentioned Envoy, which is deployed as a sidecar inside every pod, and the "Mixer" is a platform-independent component that collects telemetric data from the sidecars and makes better use of data plane policies and authentications in the service mesh.

The control plane of Istio can be split into 3 parts, the "Pilot", the "Citadel"

and the "Galley".

The "Pilot" provides service discovery for the sidecars, traffic management such as A/B testing and canary rollouts, and resiliency such as how long timeouts should be, how many times should a service call get retried before showing an error message.

The "Citadel" handles the configuration of authentication and authorization for service-to-service communication. It also can encrypt unencrypted traffic in the service mesh. The Citadel is responsible for assigning certificates to services. In this way, services can create a secure connection relying on the service identity, and not on network controls. [20]

The "Galley" is responsible for validating configurations that are made in the control plane. It also makes sure that the configurations, from the underlying container orchestration, gets updated to every other Istio components (Envoy proxies, Mixer, Pilot, and Citadel).

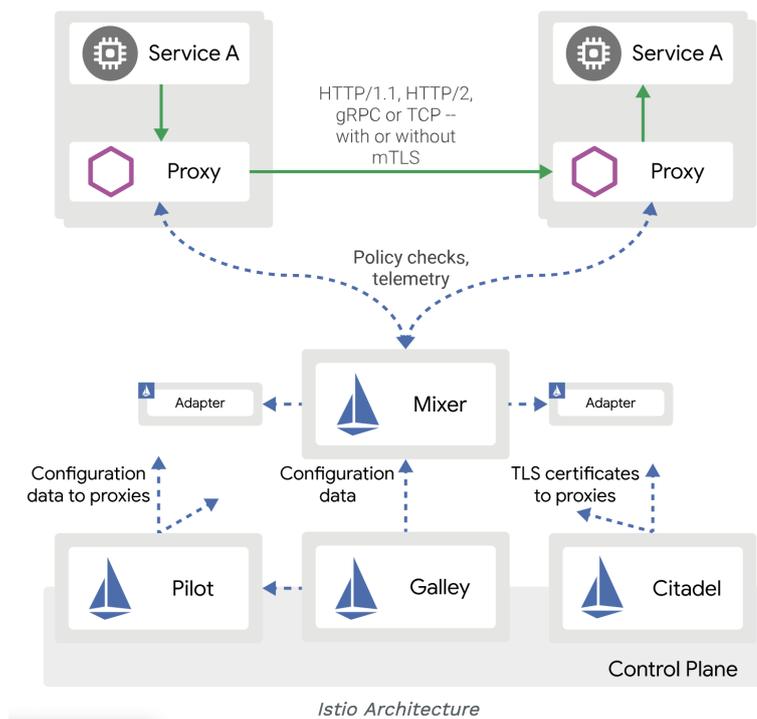


Figure 2. This picture is taken from <https://istio.io/docs/concepts/what-is-istio/>. It shows the different components of the service mesh Istio

3.5 Consul by HashiCorp

Consul is service mesh, which has its own control plane and data plane, but it is possible to switch out the data plane to some other proxies, such as Envoy proxies if wanted. Consul uses node agents as their data plane type.

4 The future for service meshes?

The time of monoliths will end soon as everything is moving towards containerized applications and microservices. Kubernetes has solved many problems regarding containerized applications, but there are some problems that Kubernetes do not fix. "Adopting service mesh is no longer a trend, it's a necessity", writes Twain Taylor in his article [22]

The concept of service mesh is still evolving every day and so are the service mesh technologies. As service meshes become more and more used in ITC industry, the more use cases and data will be caught. These use cases and data will be used to develop the true image of a service mesh and it's technologies.

New service meshes are already evolving and experimenting with new ideas. For example, Envoy is creating its own service mesh. [22] In the future, data collection and monitoring features of service meshes may be complemented with data analytics that is able to dynamically reconfigure the network as demand is changing leading to better automation.

5 Conclusions

Service meshes are a fairly new architectural type that solves problems according to service-to-service communication. The service mesh is made up of two parts, the data plane, and the control plane.

There are three different types of data planes. The data plane main functionality is to handle the communication between services, load balancing, health checking, routing, authentication and authorization, service discovery and observability.

The control planes main task is to configure the data plane.

The most popular service mesh today is Istio, which is using Envoy proxies as sidecars. There are other service meshes which work in a slightly different way, such as Linkerd and Consul.

The importance of this technology is increasing as the number of service are growing and everything moves to cloud environment,

References

- [1] Steven Acreman. Battle of the kubernetes service meshes: Service mesh. Available: <https://kubedex.com/istio-vs-linkerd-vs-linkerd2-vs-consul/>, 2018.
- [2] John Arundel and Justin Domingus. *Cloud Native DevOps with Kubernetes*. O'Reilly Media, Inc., 2019.
- [3] Envoy Project Authors. What is envoy — envoy 1.10.0-dev-a78121 documentation, 2019.
- [4] The Kubernetes Authors. Kubernetes documentation - kubernetes. Available: <https://kubernetes.io/docs/home/>, 2019.
- [5] The Kubernetes Authors. Pods - kubernetes. Available: <https://kubernetes.io/docs/concepts/workloads/pods/pod/>, 2019.
- [6] Ahmet Alp Balkan. Advanced health check patterns in kubernetes. Available: <https://ahmet.im/blog/advanced-kubernetes-health-checks/>, 2018.
- [7] Roderick Bauer. Docker containers vs. vms: Pros and cons of containers and virtual machines. Available: <https://www.backblaze.com/blog/vm-vs-containers/>, 2018.
- [8] Tom Huston. What is microservices architecture? Available: <https://smartbear.com/learn/api-design/what-are-microservices/>, 2019.
- [9] Docker Inc. What is a container? | docker. Available: <https://www.docker.com/resources/what-container>, 2019.
- [10] Martin Fowler James Lewis. Microservices. Available: <https://martinfowler.com/articles/microservices.html>, 2014.
- [11] Andrew Jenkins. What service mesh architecture should i pick? Available: <https://aspenmesh.io/2018/03/service-mesh-architectures/>.
- [12] Matt Klein. Service mesh data plane vs. control plane – envoy proxy. Available: <https://blog.envoyproxy.io/service-mesh-data-plane-vs-control-plane-2774e720f7fc>, October 2017.
- [13] Tobias Kunze. What is a service mesh? | glasnostic blog. Available: <https://glasnostic.com/blog/what-is-a-service-mesh-istio-linkerd-envoy-consul>, 2018.
- [14] Sam Newman. *Building Microservices*. O'Reilly Media, Inc., 2015.
- [15] C. Pautasso, O. Zimmermann, M. Amundsen, J. Lewis, and N. Josuttis. Microservices in practice, part 1: Reality check and service design. *IEEE Software*, 34(1):91–98, Jan 2017.
- [16] Christian Posta. Microservices patterns with envoy sidecar proxy: The series – software blog. Available: <https://blog.christianposta.com/microservices/00-microservices-patterns-with-envoy-proxy-series/>, 2017.
- [17] Inc. Red Hat. What is kubernetes? Available: <https://www.redhat.com/en/topics/containers/what-is-kubernetes>.
- [18] Chris Richardson. Who is using microservices? Available: <https://microservices.io/articles/whoisusingmicroservices.html>, 2018.

- [19] Chris Richardson and Floyd Smith. *Microservices, From Design to Deployment*. NGINX Inc, 2016.
- [20] Marcus Schiesser. Comparing service meshes: Linkerd vs. istio | glasnostic blog. Available: <https://glasnostic.com/blog/comparing-service-meshes-linkerd-vs-istio>, 2019.
- [21] Floyd Smith and Owen Garret. What is a service mesh? - nginx. Available: <https://www.nginx.com/blog/what-is-a-service-mesh/>, 2018.
- [22] Twain Taylor. Kubernetes service mesh market is a lot more than istio. Available: <http://techgenix.com/kubernetes-service-mesh/>, 2019.

Automatic rumour Detection on Social Media: A Survey

Mats Mulder

mats.mulder@aalto.fi

Tutor: Tommi Gröndahl

Abstract

Social media platforms have a major influence on the opinion and decision-making of their users. However, these platforms are sensible for the fast distribution of rumours, which can be spread by everyone and without any certainty to facts. Therefore, there exists an increasing need to automatically detect rumours on social media. This paper aims to provide a survey on the state-of-the-art methods in automatic rumour detection on social media. Two categories of methods are evaluated: traditional machine learning methods and deep learning methods. Overall, the results show a reasonable capability to automatically detect rumour. However, significant improvement has to be made, especially in the early-stage detection capabilities, to utilize automatic rumour detection methods in real-world scenario's. Deep learning methods can be seen as the most promising for future research. These methods both yield the best performances and make time consuming, biased and labor-intensive feature engineering, related to traditional machine learning approaches, redundant.

KEYWORDS: Rumour, Automatic Rumour Detection, Machine Learning, Deep Learning

1 Introduction

Social media has become an integral part of our daily lives. 88% of U.S. adults between 18 and 29 used at least one type of social media in 2018 [16]. Although these percentages decline as age increases —78% among people between 30 and 49, 64% for ages 50 to 64 and 37% among citizens 65 and over—they still reveal a wide usage of social media. In addition to the broad use of social media in general, social media platforms are widely used for the purpose of news distribution. Research on news use across social media platforms in 2018 by the Pew Research Center, states that 68% of American adults say they at least occasionally receive news on social media platforms [15]. 20% of the respondents indicated they receive news on social media often.

These numbers reveal the important role social media plays in the way people obtain information and news. As a result, these platforms have a major influence on the opinion and decision-making of their users. However, in contrast to other media channels, content on social media is also distributed by unverified authors. Recently, researchers at the Leiden Institute of Advanced Computer Science analysed 117,000 Facebook messages between 2013 and 2017 [13]. Their findings show that so-called junk news, news of low journalistic quality distributed by unknown news sites, is being shared and liked more often than news derived from well-known Dutch media authorities.

Because of the growing usage of social media and its ability to quickly spread information, these platforms are susceptible to the propagation of rumour. This, in combination with the popularity of junk news, discloses the need for automatic early-stage detection of rumour on platforms, such as Facebook, Twitter and Sina Weibo. A real-life example of the spread of a rumour on Twitter clearly indicates this need. At the end of April 2013, the official Twitter account of the Associated Press (AP) was hacked and used to distribute a fake tweet about two explosions in the White House that injured the President. Although the news was quickly rectified, the rumour has been associated with a short crash of the stock market [6].

This paper discusses the state-of-the-art methods in automatic rumour detection on social media. The focus of this survey is the automatic detection itself, not the classification of the truthfulness of the rumour. However, some research papers combine the two to obtain better detection results and are therefore included as well. First, the various definitions

of rumour are being discussed in Section 2. Section 3 includes the review of studies divided into two chapters: traditional machine learning approaches and deep learning approaches. The survey ends with the conclusion in Section 4.

2 Definition of rumour

The definition of rumour differs considerably between different studies. Varies psychological research on rumour, including work by DiFonzo and Bordia [5] and Allport and Postman [1], does not define rumour as necessarily false information, but as information which truth value is unverified. From that perspective, information that is eventually found true can also be spread as rumour. However, some studies consider rumour by definition as false information. In addition, a set of additional terms, such as false and true rumour, is being used. To be able to compare the included methods, a general definition of rumour throughout this paper is needed. Therefore, the following definitions of rumour are used:

1. *Rumour*: a story or a statement in general circulation without confirmation or certainty to facts [1].
2. *False rumour*: rumour that is eventually found false.
3. *True rumour*: rumour that is eventually found true.
4. *non-rumour*: information that is not related to rumour.

3 Review of Studies

The review of studies is divided into three subsections. First, traditional machine learning approaches are studied. Secondly, more recent research in deep learning approaches are evaluated. Finally, an overview of the results of both traditional machine learning approaches and deep learning approaches is provided.

3.1 Traditional Machine Learning Approaches

Before the increased popularity of deep learning approaches, traditional machine learning methods were used in the field of automatic rumour detection. Traditional machine learning methods involve three consecutive steps. First, *feature engineering* is used to extract features of both rumour and non-rumour. Secondly, multiple *classifiers* are trained to distinguish rumour from non-rumour using the extracted features. Finally, the classifiers are evaluated using unseen data. The first step, feature engineering, can be seen as the most essential step to conduct reasonable classification. Therefore, successive research using traditional machine learning methods mainly involves the introduction of novel feature combinations.

Features

Features can be divided in the following two categories: content features and social-context features. Content features describe the characteristics of the messages themselves, including both textual and visual characteristics. However, this survey will focus on classification using textual features. Examples of textual features include lexical, syntactic and semantic features, such as the presence of certain words, the total number of words or the meaning of sentences.

Social-context features include user-based, propagation and temporal features. User-based features analyse the characteristics of users, such as the number of followers and the date of registration. Propagation features represent statistics on the spread of the messages among users. For example, some methods model the propagation of messages as trees [17, 11]. Finally, temporal features describe the life cycle patterns of rumours and non-rumours, such as the number of messages per unit of time.

Classifiers

Traditional machine learning approaches compare the results of different classifiers to obtain the most reliable predictor. The following types of classifiers are used among the included methods: *support vector machines* (SVM), *decision trees* and *random forest*.

Overview of Traditional Machine Learning Approaches

Table 1 provides an overview of the included traditional machine learning approaches, including their best-performing classifier and a specific feature that characterizes the method.

Study	Classifier	Distinctive Feature
[3]	Decision Tree	Newsworthy + Credibility classification
[8]	SVM	Focus on conflicting messages
[18]	Decision Tree	Based on enquiring questions
[7]	Random Forest	Introduction of temporal features
[9]	SVM	Introduction of time series modeling
[17]	Hybrid SVM	Propagation graph kernel + feature learning
[11]	SVM	High-order comparison of propagation trees

Table 1. Overview of the best-performing classifier and distinctive feature of all included traditional machine learning approaches.

Review of Traditional Machine Learning Approaches

Castillo et al. [3] can be seen as pioneers in this research field. Their research focuses on the automatic determination of the credibility of content posted on Twitter, using only information available on the platform. First, Castillo et al. compose a reliable dataset of Twitter messages. To do so, a team of experts assesses the messages on newsworthiness and credibility. Subsequently, two classifiers are trained. The first classifier evaluates the messages on being newsworthy information, whereafter the second classifies the newsworthy messages on their level of credibility. After comparing different classification methods, a *J48 decision tree* is selected for both classification steps. The results yield an F_1 measure of 0.891 and 0.860 for the first and second classifier, respectively. Besides, Castillo et al. conclude that newsworthy information tends to have URL-links and deep *propagation trees*. Important features to automatically assess the credibility of Twitter messages include the number of previous tweets by the author, the number of re-posts and the number authors at the origin of the rumour propagation.

Liu et al. [8] expand the set of features of Castillo et al. [3] and apply their method on a different dataset. In their research, rumour is identified as "an event that may comprise of one or more conflicting microblogs" [8]. In other words, their method does not focus on single messages but on collections of multiple conflicting messages. Multiple classifiers are compared, whereafter a SVM yields the best performances. In addition, Liu et al. compare the accuracy of their classifier with the method of Castillo et al. [3]. Their results show an improvement in accuracy from 0.781 to 0.875.

Zhao and Resnick [18] approach the problem from a different angle compared to previous research [3, 8]. Their starting point is the hypothesis that most rumours can be identified by the presence of two types of messages: inquiries of the rumour (verification / confirmation questions) and corrections / disputes of the rumour. Zhao and Resnick propose a five step procedure to automatically detect rumours. First, a set of *regular expressions* is used to identify so-called *signal tweets* that contain skeptical enquiries, for example "really?" and "not true". Secondly, an algorithm, *connected component clustering*, is used to group the signal tweets in *identify signal clusters*. Subsequently, the system determines a single statement that represents each cluster, by extracting the most frequent and continuous substring of the related cluster. During step four, the signal tweets are combined with non-signal tweets based on overlapping content. Finally, the clusters are ranked by a classifier using an input of 13 statistical features. Two different classifiers are used: SVM and decision trees. The precision of the classifiers is evaluated using two datasets. The first includes a collection of tweets during the Boston Marathon bombing incident in 2013. The second dataset is a random sample of tweets with a time-frame of one month. In both cases, the decision tree classifier yields the best performances with a precision of 0.521 on the first dataset and 0.279 on the second dataset. The method yields an average delay of a few hours to detect rumour. Besides, Zhao and Resnick conclude that some expressions, such as "scandal?", are frequent indicators of rumour.

Kwon et al. [7] propose an approach that includes the variation of features over time. This variation is captured in the so-called *temporal features* using a method called *the Periodic External Shocks model* (PES). This model is able to describe the bursty temporal pattern of rumours and non-rumours. For example, rumour tends to have multiple and periodic spikes, a high number of tweets related to a certain topic during a short period of time, whereas non-rumour typically has a single prominent spike [7]. In addition to the temporal features, messages are also classified using content and social-context features. Three different classifiers are trained, including decision trees, random forest and SVM, and evaluated on a Twitter dataset. In addition, the classifiers are compared with the method of Castillo et al. [3]. The results show that the method of Kwon et al. [7] improves the maximum F_1 measure of 0.788 by the work of Castillo et al. [3] to 0.893 using the random forest classifier. Besides, the periodicity of external shocks is evaluated as the feature with the highest

predictive power, thereby supporting the inclusion of temporal features.

Ma et al. [9] include the variation of features over time in a more comprehensive manner. Instead of capturing the variation in a few temporal features, Ma et al. propose a time series modeling technique, called *the Dynamic Series-Time Structure* (DSTS), that captures the variation of a wide spectrum of features over time [9]. Afterwards, a linear SVM classifier is trained and evaluated on two criteria. The first criteria evaluates the ability of the classifier to determine the truthfulness of a rumour, using the complete set of messages related the rumour. The second criteria evaluates the ability to detect rumour early-stage, using only messages published before a certain point in time. Both criteria are evaluated using a Twitter and Weibo dataset and compared with previous work [3, 7]. These previous methods are outperformed by the new proposed method of Ma et al. with margins between 3.3% and 9.5%. This results in an accuracy of 0.896 and 0.846 on the Twitter and Weibo dataset, respectively.

Wu et al. [17] extend the use of propagation structures to automatically detect false rumours. Although previous research [3, 8, 7, 9, 18] included features based on the propagation of messages, the approaches over-simplified the propagation structure by using flat statistics, such as the number of retweets. Wu et al. model the pattern of messages in Weibo as a tree, which reflects both the relation between posts and their authors, and the temporal behavior and sentiment of the posts. A *random walk graph kernel* is proposed to model the similarity of the propagation trees. The kernel is combined with semantic feature engineering, forming a hybrid SVM classifier. To evaluate the proposed method, a random dataset from Weibo is composed. First, a set of known false rumours is collected and enriched using the Weibo API. Secondly, the set is combined with regular Weibo messages to achieve a real-world ratio between regular messages and false rumours. The method is compared with the research of Castillo et al. [3], resulting in an improvement of the accuracy of 6.9% to 0.913. If the method is applied for the purpose of early detection of false rumour, the model yields an accuracy of 0.880, 24 hours after the initial message. Besides, a classifier without the graph kernel is compared to the other classifiers. The results show that the graph-kernel is effective, being the largest single-feature improvement.

Ma et al. [11] continue the research on methods to automatically detect and classify rumour using propagation trees. Ma et al. propose a kernel-based method called *Propagation Tree Kernel* (PTK). This method

is able to capture the high-order propagation patterns of different types of rumour, including linguistic, user and temporal signals, by evaluating the similarities between propagation trees [11]. In addition, an extended model called *context-enriched PTK* (cPTK) is proposed. This method aims to improve PTK by considering different paths from the source tweet to the roots of subtrees. Both methods are evaluated using two Twitter datasets derived from previous research: Twitter15 [8] and Twitter16 [10]. The datasets, however, had to be modified to be usable for the classification of propagation trees. In addition, the methods are compared with previous work [9, 18, 3, 7, 10]. cPTK yields the best results with an accuracy of 0.750, outperforming the older methods with margins between 10.4% and 34.1%. Moreover, the results support the use of propagation trees; A simple version of the proposed method, using only features based on the text of messages, already outperformed all compared research.

3.2 Deep Learning Approaches

Most recent research implemented deep learning methods to automatically detect rumour. Supported by the increasing research in deep learning, the methods deal with a number of problems related to traditional machine learning approaches, including time consuming, biased and labor-intensive feature engineering. Moreover, deep learning methods have proven to outperform traditional machine learning methods. Therefore, these methods are most promising in future research.

Overview of Deep Learning Approaches

Table 2 provides an overview of the included deep learning approaches, including their class and a specific feature that characterizes the method.

Study	Class	Distinctive Feature
[10]	Recurrent	First deep learning approach
[4]	Recurrent	Deep model + attention mechanism
[14]	Recurrent	Fake news detection
[12]	Recursive	Propagation focused using tree structures

Table 2. Overview of the class and distinctive feature of all included deep learning approaches.

Review of Deep Learning Approaches

Ma et al. [10] were the first to implement a method based on deep learning. To improve on especially the intensive preprocessing of traditional machine learning approaches, Ma et al. proposed a method based on *recurrent neural network* (RNN). In their research, the evolution of rumours is seen as a continuous stream of posts. To be able to handle popular events, including tens of thousands of posts, the stream is modelled using variable-length time series, in which posts are grouped and handled as single unit by the RNN. The model is evaluated using both a Twitter and Weibo dataset, being constructed using a combination of foreknown rumour and random non-rumour posts to create a balanced collection. Four different deep structures are compared with previous research [9, 18, 3, 7]: tanh-RNN, single-layer LSTM, single-layer GRU and multi-layer GRU. Although all four deep structures outperform the previous methods which margins between 6.2% and 36.8%, the multi-layer GRU achieves the best results with an accuracy of 0.881 and 0.910 on the Twitter and Weibo dataset, respectively.

Chen et al. [4] further improve the use of recurrent neural network in automatic rumour detection. Like Ma et al. [10], their method is able to handle large amounts of messages by grouping them according to a fixed amount of messages per rumour [4]. Afterwards, a so-called *deep attention model* is trained. This model combines the recurrent deep network with an *attention mechanism* [2], being able to recognize words that strongly indicate rumour. Chen et al. assume that the importance of textual features changes over time, due to different user responses at every stage of the rumour. Therefore, a variable weight parameter in the hidden state is used to measure the importance and contribution of features to the result. The results support the hypothesis of Ma et al.: words not directly related to the topic of the rumour, including doubting or enquiring words, obtained higher weight parameters than most words directly related to the topic. The model is compared with previous research, including Ma et al. [10] listed as ML-GRU. On both the Twitter and Weibo dataset, the new proposed model outperforms the ML-GRU model. ML-GRU achieves a F_1 measure of 0.819 and 0.830 on the Twitter and Weibo dataset respectively, whereas the new model achieves a F_1 measure of 0.871 and 0.867 on the Twitter and Weibo dataset, respectively.

Although Ruchansky et al. [14] focus on the problem of fake news detection, their approach is largely similar to research on automatic detection

of rumour and therefore, worth mentioning. The team proposed a model based on three characteristics of fake news: the text of an article, the user response it receives and the source-users promoting it [14]. The model, called CSI, is composed of three modules: Capture, Score and Integrate. The first module, Capture, uses a Recurrent Neural Network to capture the response to an article, including temporal patterns. The second, Score, is constructed using a recurrent neural network as well. The aim of the module is to capture the characteristics of the user behaviour. Integrate, the third module, constructs a vector from the first two modules and classifies a news article as fake or real. The hybrid model is evaluated using two real-world datasets derived from Twitter and Weibo and compared with previous research, including the multi-layer GRU model of Ma et al. [10]. On the Twitter dataset, the F_1 measure is improved from 0.830 to 0.892 and, on the Weibo dataset, from 0.910 to 0.953.

Previous proposed methods that model the propagation of messages as trees [17] [11] suffer from problems, including extensive preparation and processing overhead due to the pairwise comparison of the tree. However, the concept is promising. Therefore Ma et al. [12] propose a method based on *recursive neural networks* that bridges semantic content and propagation clues [12]. Ma et al. base their method on the following hypothesis: repliers tend to disagree (or question) the support of false rumours or the denial of true rumours. Likewise, they tend to agree with the denial of false rumours or the support of true rumours [12]. To extend the standard model, two variations are introduced: a bottom-up (BU) and top-down (TD) model. Both variations are evaluated using two Twitter datasets, the Twitter15 [8] and Twitter16 [10] dataset, and compared with previous research by Ma et al. [10] listed as GRU-RNN. On both datasets, the new method yields a higher accuracy than GRU-RNN: from 0.641 to 0.723 on Twitter15 and from 0.633 to 0.737 on Twitter16.

3.3 Overview of Results

Table 3 provides an overview of the datasets, measurement method and results of the best performing classifier (bold in table). Moreover, the results of compared methods are included for each study. Filtered or enriched datasets are denoted with the symbols $-$ and $+$, respectively.

Study	Dataset	Measure	Results	[3]	[8]	[18]	[7]	[9]	[17]	[11]	[10]	[4]	[14]	[12]
Traditional														
[3]	T1	F ₁	0.860											
[8]	T2	Accuracy	0.781	0.875										
[18]	T3	Precision				0.521								
	T4	Precision				0.279								
[7]	T5	F ₁	0.788				0.893							
[9]	T1 ⁻	F ₁	0.785				0.848	0.894						
	W1	F ₁	0.800				0.839	0.857						
[17]	W2	Accuracy	0.854						0.913					
[11]	T2 ⁺	Accuracy	0.454		0.409			0.544		0.750		0.646		
	T6 ⁺	Accuracy	0.465		0.414			0.574		0.732		0.633		
Deep														
[10]	T6	F ₁	0.740			0.656	0.801	0.834				0.898		
	W3	F ₁	0.831			0.726	0.864	0.861				0.914		
[4]	T6 ⁺	F ₁				0.672		0.771				0.819	0.872	
	W3 ⁺	F ₁				0.642		0.795				0.830	0.867	
[14]	T6	F ₁	0.702			0.636		0.773				0.830		0.894
	W3	F ₁	0.831			0.726		0.861				0.914		0.954
[12]	T2 ⁺	Accuracy	0.454		0.409			0.544	0.493	0.667	0.641			0.654
	T6 ⁺	Accuracy	0.465		0.414			0.574	0.511	0.662	0.633			0.708

Table 3. Overview of dataset, measurement method and results of all studies. (filtered dataset: -, enriched dataset: +)

4 Conclusion

This paper aimed to provide a survey on the state-of-the-art methods for automatic rumour detection on social media. Early studies [3, 8, 18, 7, 9] mainly focused on novel feature combination to improve the predication capabilities. It is difficult, however, to identify features that are generally strong identifiers for rumour among these studies; The importance of certain types of features differs significantly between various methods and applied on different datasets. Nevertheless, more recent traditional machine learning approaches [17, 11] show significant improvement by focusing on the propagation patterns of rumours. Although deep learning approaches generally outperformed traditional machine learning approaches, the traditional machine learning approach by Ma et al. [11] remarkably outperformed the first deep learning method [10] by using a propagation tree kernel. Moreover, the results of the most recent study by Ma et al. [12], which combines deep learning with propagation structures, support this finding; Their approach is compared with most studies included in this survey [3, 18, 9, 17, 10, 11] and yields the best performances on both the Twitter and Weibo dataset.

Overall, the results of all included studies show a reasonable capability to automatically detect rumour. Liu et al. [8] compared there method with the verification of rumour by human experts. Their method detected 75% of the rumours earlier than the human experts with an accuracy of

at least 0.800. However, the results also reveal the significant room for improvement in this field of study. Regarding the emergent impact of this problem, future studies should be able to identify a larger fraction of distributed rumours and especially improve on the early-stage detecting capabilities. Although most methods claim to detect rumour early stage [8, 9, 18, 11, 17, 10, 12, 4], all methods needed at least a couple of hours to obtain reasonable results. However, the false tweet about the explosions in the White House, being associated with a short crash of the stock market, was only online for four minutes [6]. In general, deep learning approaches can be seen as most promising in future research. These methods both yield the best performances and make time consuming, biased and labor-intensive feature engineering redundant. This could be especially useful, in the fast-changing world of social media.

References

- [1] Gordon W Allport and Leo Postman. The psychology of rumor. 1947.
- [2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [3] Carlos Castillo, Marcelo Mendoza, and Barbara Poblete. Information credibility on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 675–684. ACM, 2011.
- [4] Tong Chen, Xue Li, Hongzhi Yin, and Jun Zhang. Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 40–52. Springer, 2018.
- [5] Nicholas DiFonzo and Prashant Bordia. *rumor psychology: Social and organizational approaches*. American Psychological Association, 2007.
- [6] Patti Domm. False rumor of explosion at white house causes stocks to briefly plunge; ap confirms its twitter feed was hacked. April 2013.
- [7] Sejeong Kwon, Meeyoung Cha, Kyomin Jung, Wei Chen, and Yajun Wang. Prominent features of rumor propagation in online social media. In *2013 IEEE 13th International Conference on Data Mining*, pages 1103–1108. IEEE, 2013.
- [8] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1867–1870. ACM, 2015.
- [9] Jing Ma, Wei Gao, Zhongyu Wei, Yueming Lu, and Kam-Fai Wong. Detect rumors using time series of social context information on microblog-

- ging websites. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1751–1754. ACM, 2015.
- [10] Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J Jansen, Kam-Fai Wong, and Meeyoung Cha. Detecting rumors from microblogs with recurrent neural networks. In *IJCAI*, pages 3818–3824, 2016.
- [11] Jing Ma, Wei Gao, and Kam-Fai Wong. Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 708–717, 2017.
- [12] Jing Ma, Wei Gao, and Kam-Fai Wong. Rumor detection on twitter with tree-structured recursive neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1980–1989, 2018.
- [13] Pekka Nikander. PulpnieuwspaginaTMs bereiken via facebook miljoenen nederlanders. <https://nieuwscheckers.nl/nieuwscheckers/pulpnieuwspaginas-bereiken-via-facebook-miljoenen-nederlanders/>, January 2019. [Online; accessed 24-January-2019].
- [14] Natali Ruchansky, Sungyong Seo, and Yan Liu. Csi: A hybrid deep model for fake news detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 797–806. ACM, 2017.
- [15] Elisa Shearer and Katerina Eva Matsa. News use across social media platforms 2018. September 2018.
- [16] Aaron Smith and Monica Anderson. Social media use in 2018. March 2018.
- [17] Ke Wu, Song Yang, and Kenny Q Zhu. False rumors detection on sina weibo by propagation structures. In *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*, pages 651–662. IEEE, 2015.
- [18] Zhe Zhao and Qiaozhu Resnick, Pauldayand Mei. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1395–1405. International World Wide Web Conferences Steering Committee, 2015.

Study	Dataset	Measure	Results [3]	[8]	[18]	[7]	[9]	[17]	[11]	[10]	[4]	[14]	[12]
Traditional													
[3]	T1	F ₁	0.860										
[8]	T2	Accuracy	0.781	0.875									
[18]	T3	Precision		0.521									
	T4	Precision		0.279									
[7]	T5	F ₁	0.788			0.893							
[9]	T1 ⁻	F ₁	0.785			0.848	0.894						
	W1	F ₁	0.800			0.839	0.857						
[17]	W2	Accuracy	0.854				0.913						
[11]	T2 ⁺	Accuracy	0.454	0.409			0.544		0.750		0.646		
	T6 ⁺	Accuracy	0.465	0.414			0.574		0.732		0.633		
Deep													
[10]	T6	F ₁	0.740	0.656		0.801	0.834			0.898			
	W3	F ₁	0.831	0.726		0.864	0.861			0.914			
[4]	T6 ⁺	F ₁		0.672			0.771			0.819	0.872		
	W3 ⁺	F ₁		0.642			0.795			0.830	0.867		
[14]	T6	F ₁	0.702	0.636			0.773			0.830		0.894	
	W3	F ₁	0.831	0.726			0.861			0.914		0.954	
[12]	T2 ⁺	Accuracy	0.454	0.409			0.544	0.493	0.667	0.641			0.654
	T6 ⁺	Accuracy	0.465	0.414			0.574	0.511	0.662	0.633			0.708

Microservices architecture in I-IoT applications: Motivation and security challenges

Sonika Ujjwal

sonika.ujjwal@aalto.fi

Tutor: Hirvisalo Vesa

Abstract

The traditional Industrial- IoT (I-IoT) application, based on monolithic architecture, provides the security and centralised control over the system. However, it faces challenges in terms of scalability and deployability as business capabilities in the system increases. The distributed nature of the application, network complexity and need for interoperability on different kinds of end-devices demands a distributed, modular and scalable solution. Microservice (MS) architecture provides one such solution. As, I-IoT application processes, aggregates and presents the security and safety sensitive data, it should be protected from external processes or entities. Therefore, security in such systems should be properly addressed. This paper presents the current monolithic architecture along with its scalability and deployment challenges. These challenges act as the motivation to adopt MS architecture for I-IoT applications. Providing a comparative view of both the architectures in terms of scalability and security, this paper outlines the security concerns among communicating microservices. By illustrating one specific framework, Kubernetes, for MS application orchestration, this paper describes the secret information handling for secure inter-service communication in that framework along with other possible solutions for secure application data storage and management.

KEYWORDS: *big data, cloud computing, fog computing, containers, docker, kubernetes, Message Queues, Rule-based access policy, barbican*

1 Introduction

Using new and emerging technological breakthroughs, the fourth industrial revolution (industry 4.0) has blurred the boundaries of the physical, digital and biological spheres.

Industry 4.0 envisages the adoption of Internet-of-things (IoT) for use in manufacturing, and has great potential to improve the productivity, efficiency, safety and intelligence of industrial units [11]. Industrial-IoT system envisions massive deployment and integration of smart computing devices with cutting edge network technologies in industrial settings. Robot-operated warehouses and smart automation of factory processes are some examples of industries utilising the endless capabilities of IoT systems in industrial schemes.

The enormity of the scale on which modern I-IoT systems are deployed necessitates the newer system architecture principles that can resolve challenges such as faster integration, deployment and the scalability of components at application level [29].

Microservice architecture is one such application design principle that facilitates the migration of system architecture from complicated, tightly-coupled monolithics to the loosely-coupled, transaction-less system of microservices. Microservices provide the benefits of rapid development and scalability of software systems, easy debugging of failures, quick deployment and guaranteed availability of system services [9] [21]. However, the migration from centralised monolithic system introduces some security concerns in the I-IoT applications. The presence of security-critical and privacy-sensitive data in I-IoT application systems makes them an attractive target for adversarial attacks. The network complexity and remote end-devices in the architecture further provide increased attack surface for the adversary [14].

This paper aims to provide an overview of I-IoT application, motivation for adopting microservice architecture and the modern solutions that can be used for securely managing applications. This paper would not address the network and embedded system security as those are altogether different domain of research [7] [17]. Section 2 provides insights about I-IoT systems, high-level I-IoT architecture and application layer architectures such as monolithic and microservices based. Section 3 introduces the mi-

crosservice architecture for I-IoT applications and how such architecture deal with security and scalability challenges. Section 4 enlists some of the related work in this domain. For realising the MS architecture in a concrete manner, Section 5 introduces Kubernetes system for I-IoT application orchestration, describing in length its components and secret handling in the system. Section 6 discusses few modern methods of managing secret information at application level.

2 Industrial- IoT: Revolutionising the industries

As an important extension and application of IoT, Industrial-IoT is the combination of IoT, big data, Machine to Machine (M2M) communication, cloud computing, and real-time analysis of data from interconnected smart devices [6]. As an emerging technology, I-IoT has distinct properties and requirements that distinguish it from consumer or mainstream IoT. It interconnects heterogeneous smart devices (sensors, actuators and controllers) with different network technologies in industrial manufacturing and automation systems [29].

2.1 I-IoT architecture

Numerous research and survey papers have defined I-IoT as three or five layered architecture. In case of 3 layer system, the comprising layers are application layer, communication layer, and physical layer while in 5 layer architecture the constituent layers are perception layer(similar to physical layer from 3-layer architecture), network and middleware layer (corresponding to communication layer), and application and business layer (corresponding to application layer) [26] [29].

The physical layer consists of widely deployed end-devices, such as sensors, actuators, manufacturing equipment and automation-related objects. This layer takes care of end-device management, data acquisition and transmission of data through communication layer to central processing system.

Communication layer is defined by the integration of numerous communication networks, such as 5G, M2M, SDN, and so on. It takes care of the secure transmission and interconnection of end-device with central processing system. It supports a number of protocols and data formats to ensure interoperability of heterogeneous devices of physical layer.

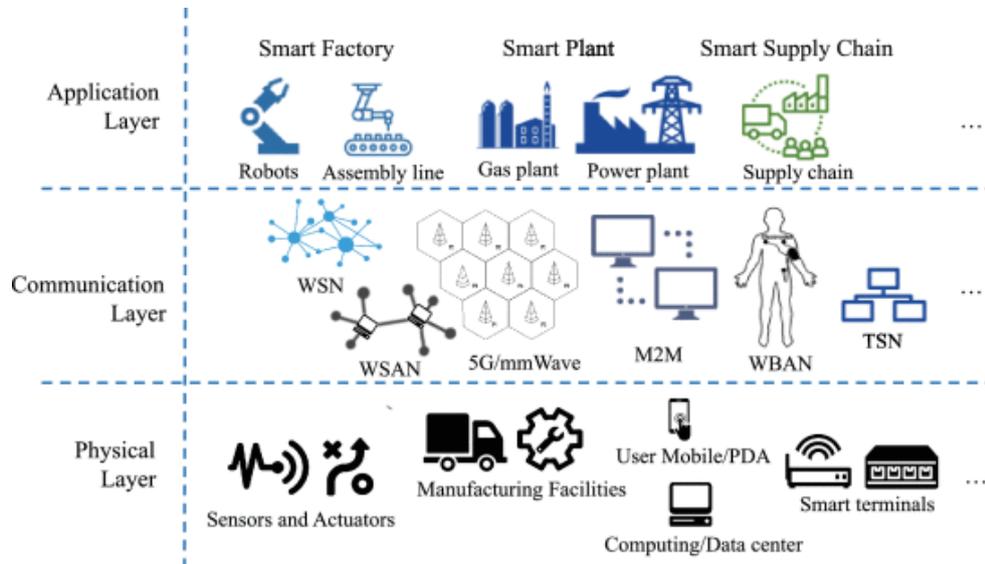


Figure 1. I-IoT Architecture (Courtesy: A Survey on Industrial Internet of Things: A Cyber-Physical Systems Perspective [29])

The application layer provides I-IoT with cloud platforms to process and analyse the received data and provide a scientific basis for decision-making. Various industrial applications, such as smart factories, smart plants and smart supply chains, are deployed in this layer. These smart industrial applications leverage numerous sensors and actuators for the purpose of timely monitoring and decision-making, strict control and efficient management.

2.2 I-IoT applications

I-IoT application is devised on application layer and utilises the data harvested by the numerous physical device. It processes and analyses the information by making use of the cloud platforms, thus providing decision making ability to the system. Similar to Enterprise applications, it deals with the presentation, business logic and database access aspect. Monolithic architecture supports the traditional heavyweight service model of enterprises and can be at odds with realistic I-IoT infrastructures [25].

Monolithic application is a single-tiered, logically modular and single executable set of services. Its is easy to develop, test and deploy. It is considered secure because whole application along with the enterprise-wide database is generally hosted at the same physical place and networking between different components/machines is enabled by Enterprise server bus (ESB). ESBs are fully controlled by enterprises and security is handled by specialist personnel. For load balancing, multiple machines can be used to run replicas of the whole application [25].

2.3 Challenges in I-IoT application: Security and Scalability

The single executable nature combined with the multitude of services makes it difficult for application to develop and integrate new services, continuously deploy and scale efficiently. Monolithic application is scaled by running required instances of whole application replicas on different host machines and load-balancer is used to distribute load on these replicas [16]. In case only a few services needs scaling, this schemes results in inefficient usage of resources. In addition, all stages in application management, such as building, integration and deployment, require considerable amount of time. Development time is also high as all the services are tightly coupled requiring complex coordination between teams working on different services.

Another major challenge is security and privacy of I-IoT system data and resources [12]. Due to the security-critical and privacy-sensitive nature of data available to I-IoT systems, they are prone to numerous adversarial attacks [3] [8] [22]. Every layer in I-IoT architecture has its own vulnerabilities and security problems. Physical and communication layer may suffer from attacks such as Sleep Deprivation Attack on end device, Denial of Service (DoS) Attack, Malicious Code Injection (MIC) and Man-in-the-Middle (MitM) Attack. In Addition to DoS, MIC, application layer may suffer from sniffing and spear-phishing attacks [26].

However, this paper mainly concerns with discussing the optimal architecture of I-IoT applications and the operational and security challenges at application layer.

2.4 Monolithic or Microservice Architecture?

Microservices are an approach to application development in which a large application is built as a set of modular services (i.e. loosely coupled modules/components). Each module supports a specific business goal and uses a simple, well-defined interface to communicate with other sets of services [25].

It adheres to the principle of strong cohesion and weak coupling. A microservice has its own suit of functions, objects and database to support modularity. Intra-MS communications are rich and components (functions/objects) inside an MS are cohesive and tightly coupled while inter-MS communications are infrequent using predefined API that abstracts microservice implementation and functionality details from other MS.

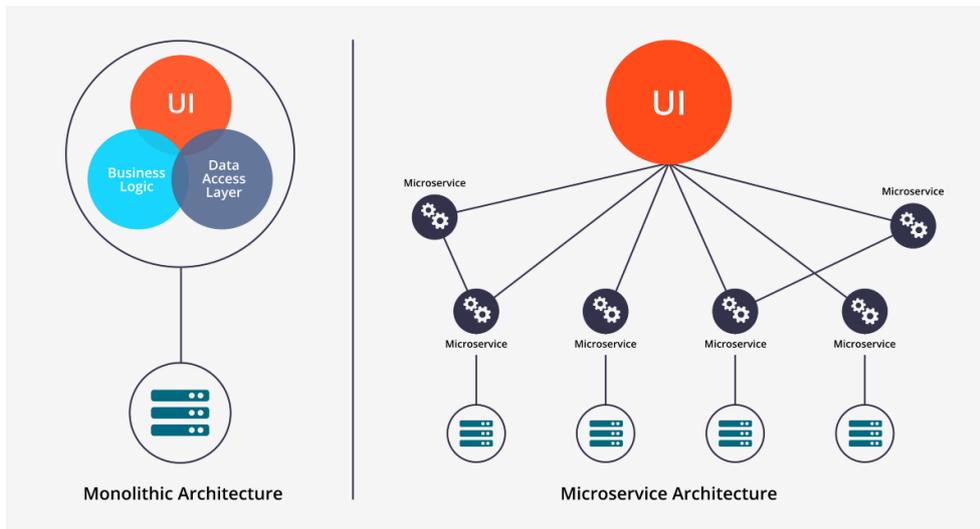


Figure 2. Monolithic vs Microservice architecture (Source: Weave.Works)

For scaling purpose, only the load expecting services can be scaled without scaling the whole set of services in application. The development of an MS is faster due to weak coupling with other services. An MS can be deployed rapidly and independently following the CI/CD pipeline [21]. In case of monolithic application, a malfunctioning component or function can shut-down the whole application, while in case of MS application, a failed MS would not affect the whole system and the application will keep functioning. To ensure availability, MS architecture incorporates additional mechanisms for failure recovery and rapid debugging of failures.

3 MS architecture for I-IoT

An I-IoT application can be constructed as a cluster of numerous fine-grained and self-contained microservices, which are independently developed and deployed to leverage the benefits of MS architecture. MS provides distributed, modular and easily scalable architecture that satisfies key requirements for the execution of I-IoT applications.

These microservices corresponds to the abstraction of a single functionality (in other words, business capability) and are often deployed inside portable, lightweight, execution environment called containers.

3.1 Containers : Tools for microservice deployment

Containers have empowered the usage of microservice architecture by providing least start-up latency, faster integration and deployment, execution isolation and low overhead of management. [15].

Container can be visualised to be running individual microservices of the MS cluster. They allow smooth and efficient deployment of a microservice by packing the microservice along with its dependencies into a single image deployed to the container. They provide an MS with its own virtual environment for execution and work on the principle of isolated execution, making each MS self-contained and independently deployable. [1].

3.2 Microservices

Each microservice can use its own middleware stack and network technology to communicate with physical and communication layer components. Moreover, instead of sharing a single database with other services, each service can have its own database, the so-called polyglot persistence architecture, ensuring loose coupling [25]. As a result, an MS trusted with the messaging or notification service may have Redis DB while an MS managing inventory can have SQL or MongoDB database. Since MS architecture is based on the independence of individual MS, different Microservices may communicate using a protocol they deem suitable, e.g., some MS may use synchronous protocols, such as HTTP/REST, while others may use asynchronous protocols, such as MQTT, CoaP. Microservices expose some public facing REST, RPC or message-based API for communication and consume APIs provided by other services.

3.3 API gateway

API gateway is a dedicated component in MS architecture that works as a proxy and relays the request between MS [27]. It serves as a single point of entry to microservices cluster working on similar lines as a load balancing proxy for web services, or a message broker of MQTT protocol.

API gateway keeps track of all the microservices in the cluster by provisioning, routing and discovering MS. An MS needs to register itself with the API gateway in order to be 'discovered'. This process is called service registration and it creates unique endpoints for an MS discovery. If these MS are hosted on different machines, then these endpoints correspond to the service id of the MS, IP address and port number of the machine on which the MS is running. API gateway maps these endpoints to actual protocol, such as an URL in REST API. In cases where different MS supports different communication protocols, the gateway can be realised as another microservice providing inter-MS translation and making mi-

crosservices in different domain accessible to each other [19].

3.4 Fog/edge computing

One of the key challenges in distributed I-IoT systems is latency sensitivity in data transmission and processing. This can be alleviated by coupling traditional cloud computing with the edge or fog computing paradigm, i.e., moving computation to the edge closer to the data producing nodes. These edge devices send the processed computation to cloud MS, providing much better latency performance as they are usually located close to end-devices. It is easier as well as cheaper to move aggregate computation or summary to the remote cloud microservice than it is to move the whole block of end-device generated data [2].

3.5 Security in communication inside MS architecture

The large number of distributed MS greatly increase the difficulty to monitor the security of the entire application. MS are often designed to completely trust each other, therefore compromise of a single microservice may bring down the entire application. The security of application data, the permission to communicate to and consume the service of another MS and the protection of private resources of an MS demand proper identification, authentication and authorisation policies for access in place.

The API gateway has strategic position in MS architecture and supports the multiple line of defense principle. API gateway is designated with identification of system microservices during registration process. For authentication and authorisation, it could use single sign-on methods, passwords, API token, JWT token for java based API [13]. In addition to that, Role based access policies (RBAC) can be inculcated in API gateway for fine and coarse grain accesses to the application and microservices. These services decide which host or MS can consume or communicate with another MS or with the application's public API [5]. User or machine client are also required to identify themselves in order to interact with the API gateway to access its associated microservices.

This microservice style of highly isolated and distributed components requires new mechanisms for better application security, e.g., through increased diversity or restricting data access to only those services that absolutely need it[30]. As examined above, communications between different MS are necessary and security policies needs to be defined upon

these communications, using principle of least privilege.

4 Related work in I-IoT application domain

For secure MS architecture orchestration, [18] defines a microservice trust-model based on MTLS (Mutual TLS) which is used by all the nodes in a swarm to securely communicate with each other. Some research papers have devised primitives that enable cloud vendors to provide security-as-a-service for cloud applications that are based on microservice architecture [23]. Some papers have extended I-IoT to healthcare sector, describing a cloud-integrated HealthIIoT monitoring framework where security and safety of healthcare data is ensured by watermarking it before sending to the cloud for high-quality health monitoring [10].

5 Kubernetes for I-IoT application orchestration

This paper examines the Kubernetes system to demonstrate the deployment, secure access control and sensitive information handling during application orchestration. MS architecture requires management, coordination, automatic scaling and failure recovery mechanism for numerous containerised distributed microservices. Kubernetes helps in managing these cloud native microservices.

"Kubernetes is a portable, extensible open-source platform for managing, deploying and scaling containerized applications, that facilitates both declarative configuration and automation" [24]. Google Container Engine (GKE) is the most suitable cloud service for running Kubernetes, providing rich features such as direct and virtual management, and streamlined integration. Some of the key components of Kubernetes are:

Pods: Pods are the smallest, independently deployable unit in kubernetes. A pod may consist of one or more containers. All containers in a pod share the same set of resources, such as storage, IP address, port space. All the containers are tightly coupled in a pod.

Master and nodes: Kubernetes components can be categorized as master and node components, together called the cluster. Nodes are worker machines (such as a VM or physical machine) which run the pods and are managed by the master.

API server: It exposes the Kubernetes API which is used to connect to

cluster. Other important components are *controller manager, scheduler, etcd, cloud controller manager and Replication Controller*.

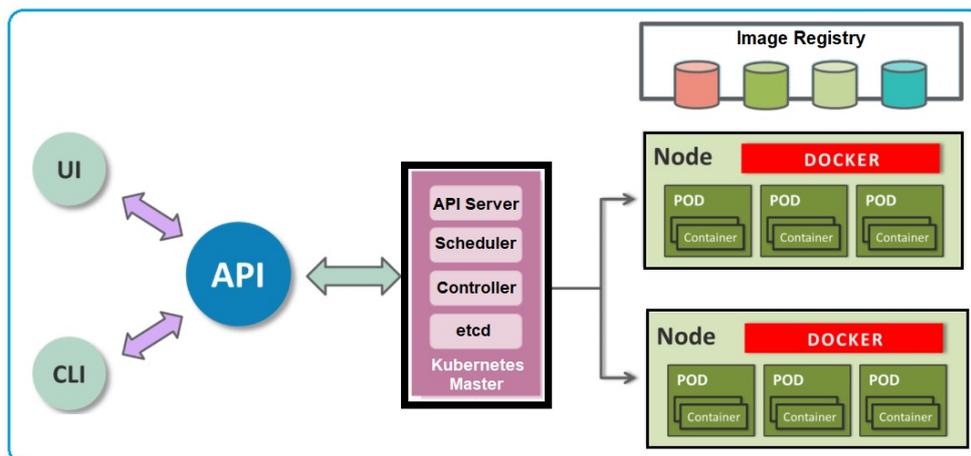


Figure 3. High-Level view of Kubernetes Architecture (Source: dZone.com)

Kubernetes utilizes pod abstraction as service so that it is easier to locate the services using specific endpoints. One or more pods may be required to host a single MS of I-IoT. These services(pods abstracted as service) are registered at service discovery phase. By this means, kubernetes manages the pods that containerise MS of the I-IoT system and define their communications. For containerisation any underlying software virtualisation technology e.g. Docker can be used [24].

5.1 Scalability

Pods are used as the unit of replication in Kubernetes. In case of scaling a load-expecting MS, the pods containing that MS can be scaled automatically by Kubernetes master. Kubernetes can be configured to deploy new replicas of these pods to the cluster as necessary.

5.2 Security

The inter-services communication are of utmost importance and can be restricted using access control policies. These policies are generally based on some passwords, access-token, ssh-keys, TLS certificates. These instruments comprise the sensitive information that needs to be protected in a system. To enable communication between different MS (nodes in kubernetes system) and between different API of a single MS (realised as containers in a single pod), some sensitive information needs to be shared between communicating entities which authenticates and authorises the entities. For e.g. access to a high risk MS or an MS producing Enterprise-

private data should be controlled with some secret keys. Only MS having that secret key will be permitted to access those protected MS.

5.3 Secret handing for I-IoT application

The main challenge is protecting and sharing the secret information necessary for inter-MS communications in I-IoT application. Following are some ways in which secret information is made available to MS running inside a container by Kubernetes system:

- Putting the sensitive information in container image.
- Putting the sensitive information in Pod-specification.
- Storing the sensitive information in special purpose object called secret.

Container image is the immutable and executable file, capable of running an independent service on a containerised platform, such as Docker. Embedding secrets in images is a security risk as anyone with an access to the image will be able to extract and examine secrets[4]. A better approach to handle sensitive information is by using 'secret' objects which is safer and more flexible in terms of utility.

Kubernetes defines the method to create as well as consume a secret on a cluster. Secrets can be created manually or by using Kubernetes CLI and are stored in secret object in a base64 encoded form. Secrets are created automatically by kubernetes system to enable pods to use certain API. They are made available to the pods by kubernetes master either as a volume mounted on the the pod or during initialization of a pod when pulling its image using kubelet, a node-agent component of kubernetes system. Since they are managed by master node and made available to pods during runtime, they are more secure as compared to other options. So, communications among microservices can be secured using secret objects for sharing sensitive information for access control.

6 Modern solutions for application layer security

Kubernetes secret object is still vulnerable in high security-sensitive environment as the secret information is only encoded and not encrypted. The secrets can be made more secure using Public-key cryptography: generating a pair of asymmetric keys and encrypting the secrets with the public key which can be later decrypted by private key using Kubernetes CLI. Due to the maturity of the public-key cryptography, almost all major pro-

programming language has its own library/tools to utilize this technique [28].

On similar lines, google KMS, a cloud-hosted key management service can be used to manage cryptographic keys for cloud services. Other example of open source solution for securing secrets in cloud native complex applications is Barbican. Barbican is an OpenStack Key Manager service. It provides secure storage, provisioning and management of secret data. It supports the protection of its APIs by enforcing rule-based policy. New secret can be created via a POST request only if the request token enlists either the admin or creator role [20].

On similar lines, EdgeX Foundry Project (<https://www.edgexfoundry.org/>) is an open-source initiative to standardise microservices-based I-IoT architecture incorporated with edge-computing. Instead of tightly coupling security guidelines in each MS, Kong (<http://www.konghq.com>), an open-source microservice API gateway, is chosen to secure the EdgeX microservice cluster. MS in EdgeX cluster safely store and retrieve sensitive data such as encryption keys or authentication credentials in Security Secret Store which is implemented using Vault, by Hashicorp (<https://www.vaultproject.io/>). Vault manages and enforces access to secrets and systems based on authorised services and user identity.

7 Conclusion

To leverage the power of cloud computing and big data analysis, Industrial I-IoT systems require rapid and efficient scalability. For scalability, MS architecture is a better fit for I-IoT applications than monolithic architecture. However, the cloud native and distributed nature of microservices necessitates the adoption of security mechanism for inter-MS communication. In addition to that, proper role based policies on basis of authentication and identification of entities need to be in place for ensuring the authorised and secure access to the application. This paper outlines scalability solution and sensitive information handling in kubernetes framework along with some modern methods that can be used to ensure application security.

References

- [1] Charles Anderson. Docker [software engineering]. *IEEE Software*, 32(3):102–c3, 2015.
- [2] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012.
- [3] Eric Byres and Justin Lowe. The myths and facts behind cyber security risks for industrial control systems. In *Proceedings of the VDE Kongress*, volume 116, pages 213–218. Citeseer, 2004.
- [4] Marc Campbell. Don't embed configuration or secrets in docker images. *Medium Corporation [US]*, 2017.
- [5] Ju Chen, Yi Liu, and Yueting Chai. An identity management framework for internet of things. In *2015 IEEE 12th International Conference on e-Business Engineering*, pages 360–364. IEEE, 2015.
- [6] Min Chen, Jiafu Wan, Sergio González, Xiaofei Liao, and Victor CM Leung. A survey of recent developments in home m2m networks. *IEEE Communications Surveys & Tutorials*, 16(1):98–114, 2014.
- [7] Ang Cui and Salvatore J Stolfo. A quantitative analysis of the insecurity of embedded network devices: results of a wide-area scan. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 97–106. ACM, 2010.
- [8] Dacfe Dzong, Martin Naedele, Thomas P Von Hoff, and Mario Crevatin. Security for industrial communication systems. *Proceedings of the IEEE*, 93(6):1152–1177, 2005.
- [9] Wilhelm Hasselbring. Microservices for scalability: keynote talk abstract. In *Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering*, pages 133–134. ACM, 2016.
- [10] M Shamim Hossain and Ghulam Muhammad. Cloud-assisted industrial internet of things (iiot)-enabled framework for health monitoring. *Computer Networks*, 101:192–202, 2016.
- [11] Jay Lee, Behrad Bagheri, and Hung-An Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters*, 3:18–23, 2015.
- [12] Shi-Wan Lin. Industrial internet reference architecture. *www.iiconsortium.org*, 2017.
- [13] Duo Lu, Dijiang Huang, Andrew Walenstein, and Deep Medhi. A secure microservice framework for iot. In *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pages 9–18. IEEE, 2017.
- [14] Pratyusa K Manadhata and Jeannette M Wing. An attack surface metric. *IEEE Transactions on Software Engineering*, 37(3):371–386, 2011.

- [15] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, 2014(239):2, 2014.
- [16] microservices.io. Pattern: Microservice architecture. <https://microservices.io/patterns/>, 2003.
- [17] Bill Miller and Dale C Rowe. A survey scada of and critical infrastructure incidents. *RIIT*, 12:51–56, 2012.
- [18] D Mónica. MTLs in a microservices world, 2016.
- [19] Roberto Morabito, Riccardo Petrolo, Valeria Loscrí, and Nathalie Mitton. Enabling a lightweight edge gateway-as-a-service for the internet of things. In *2016 7th International Conference on the Network of the Future (NOF)*, pages 1–5. IEEE, 2016.
- [20] Official Openstack.org authors. Key manager service. 2018.
- [21] Cesare Pautasso, Olaf Zimmermann, Mike Amundsen, James Lewis, and Nicolai M Josuttis. Microservices in practice, part 1: Reality check and service design. *IEEE Software*, 34(1):91–98, 2017.
- [22] Kevin Poulsen. Slammer worm crashed ohio nuke plant network. <http://www.securityfocus.com/news/6767>, 2003.
- [23] Yuqiong Sun, Susanta Nanda, and Trent Jaeger. Security-as-a-service for microservices-based cloud applications. In *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 50–57. IEEE, 2015.
- [24] Official The Kubernetes Authors. v1.13 documentation. 2018.
- [25] Siraj ul Haq. Introduction to monolithic architecture and microservices architecture. *Medium Corporation [US]*, 2018.
- [26] Shivangi Vashi, Jyotsnamayee Ram, Janit Modi, Saurav Verma, and Chetana Prakash. Internet of things (iot): A vision, architectural elements, and security issues. In *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 492–496. IEEE, 2017.
- [27] Kun Wang, Yihui Wang, Yanfei Sun, Song Guo, and Jinsong Wu. Green industrial internet of things architecture: An energy-efficient perspective. *IEEE Communications Magazine*, 54(12):48–54, 2016.
- [28] Zhimin Wen. Public-key cryptography on kubernetes secret. *Medium US*, 2018.
- [29] Hansong Xu, Wei Yu, David Griffith, and Nada Golmie. A survey on industrial internet of things: A cyber-physical systems perspective. *IEEE Access*, 6:78238–78259, 2018.
- [30] Tetiana Yarygina and Anya Helene Bagge. Overcoming security challenges in microservice architectures. In *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pages 11–20. IEEE, 2018.

Usability problems of email authentication standards with mailing lists

Alexander Gödeke

alexander.godeke@aalto.fi

Tutor: Thanh Bui

Abstract

For sending normal email, mail servers can use the security additions SPF, DKIM, and DMARC to guarantee the integrity and authenticity of the message during transport. However, for mailing lists, these security additions do not always work. Mailing lists often append information to the original message body or the header before forwarding it to the subscribers. This commonly used practice invalidates the checks of these security additions.

In this work, we describe the usability problems with mailing lists. Furthermore, we review existing solutions to solve these problems. The Authenticated Received Chain (ARC) protocol is one of the solutions which was designed to solve all currently known usability problems mentioned in this paper.

KEYWORDS: DMARC, SPF, DKIM, Maling lists

1 Introduction

Since the beginning of the Internet, Electronic Mail (email) is one of the most commonly used communication services. It is often used to send all kind of confidential data, such as contracts or recovery links for forgotten passwords. For sending and relaying email, the Internet Engineering Task Force (IETF) standardized the Simple Mail Transfer Protocol (SMTP) protocol in 1982 [17], which is currently used in the updated version of 2008 [11] for all emails. However, while a large amount of critical data is sent by email, the SMTP protocol was not designed to protect the content or other information of the message, such as the sender address or the subject.

To solve these security issues in SMTP, the IETF published the four additional protocols STARTTLS [6], Sender Policy Framework (SPF) [10], DomainKeys Identified Mail (DKIM) [4], and Domain-based Message Authentication, Reporting, and Conformance (DMARC) [13]. These protocols protect the integrity and authenticity of the messages during transport between the mail servers. Although these additional protocols are known to be secure, the adoption rates are still low. The percentage of mail servers that implemented SPF was 40% in 2015 [5] and 44.9% in 2018 [8]. The adoption rate of DMARC is even lower, which is 1.1% in 2015 [5] and 5.1% in 2018 [8].

One reason for the low adoption rates may be the continuous problems with these protection standards and mailing list. Mailing lists forward incoming messages to multiple subscribers. They often change some content of the original message before forwarding it. These changes break the email delivery, because mail servers that implemented the security additions will reject these messages. As Yahoo implemented a DMARC to reject all emails with a failed check in 2014, sending emails to mailing lists from a Yahoo email address was not possible anymore if the receiving mail server also implemented DMARC [9, 19]. Since this incident, mailing list providers are searching for different solutions to solve the problem with security and mailing lists.

This paper describes the usability problems of email authentication standards and mailing lists and compares different approaches to solve these problems. The structure of the paper is as follows. Section 2 explains the standardized security additions SPF, DKIM, and DMARC. Section 3 describes the usability problems with the mailing lists. Section 4 presents solutions for the problems mentioned in section three. Section 5 discusses the solutions, and Section 6 concludes the paper.

2 Email Security Standards

To send and relay emails, the IETF has standardized the SMTP [11] protocol. SMTP uses Mail Transfer Agents (MTAs) to deliver emails to the receiver. To deliver one message to the MTA for the receiver, the message can be relayed by multiple other MTAs to reach the receiving MTA. SMTP was standardized without any security properties. Later, the IETF published STARTTLS [6] for encryption and SPF, DKIM, and DMARC for authenticating the sender of the email [10, 4, 13]. In this section, we describe these authenticating standards in detail.

SPF Sender Policy Framework (SPF) [10] is used to define the IP addresses of mail servers that are permitted to send emails for a domain. The IP addresses of authorized email servers for a domain are distributed in a Domain Name System (DNS) text record of the domain. The receiving MTA can verify the sending MTA by getting the SPF DNS record of the sender domain and check if the IP address is authorized to send messages for this domain. Additionally, the DNS record can contain information about the action that the receiving mail server should perform if the verification of the sending MTA fails.

SPF is used to prevent email spoofing attacks, where an attacker set up his own MTA and sends emails for a different domain. If the receiving mail server implemented SPF and the domain added an SPF record in the DNS, the mail can be rejected or flagged as spam by the receiving mail server according to the rules provided in the SPF DNS record.

DKIM DomainKeys Identified Mail (DKIM) [4] is used to provide authenticity and integrity of the email during transport between the mail servers. Therefore, an attacker can not modify the content of the message without invalidating the DKIM signature.

Figure 1 shows the sending and validation process for DKIM messages. In the first step, the client sends the email to the MTA of its domain (I). Next, the sending MTA appends the hash of the body of the message to the DKIM header field as well as signing the hash and the list of headers with the domain's private key (II). Afterwards, the sending MTA sends the message, including the DKIM header, to the receiving MTA (III). The receiving MTA then fetches the DKIM record and the public key from the DNS server of the sending domain (IV). Finally, the receiving mail server compares the hash of the email body to the provided hash from the DKIM header (V) and checks the signature of the DKIM header (VI). Only if the signature is valid and the hash of the body matches the one specified in the DKIM header, the DKIM check passes.

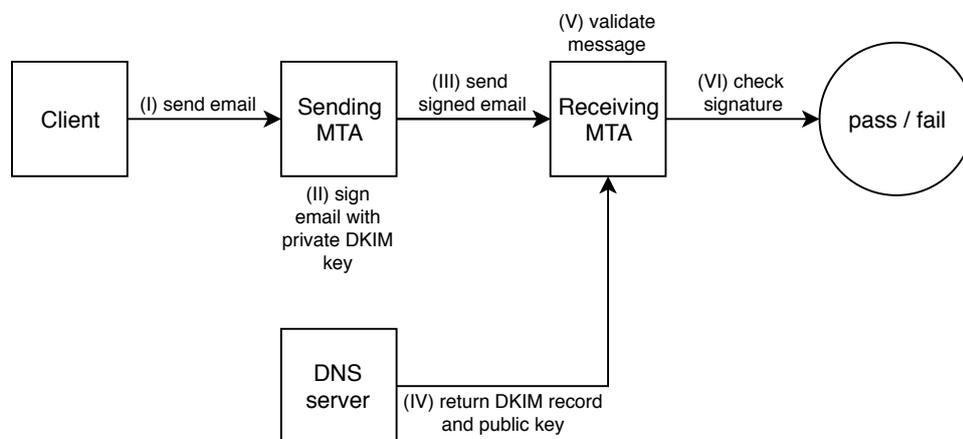


Figure 1. DKIM sending and validation process [21]

In case the email is relayed by multiple mail servers, the header can contain multiple DKIM signatures. In contrast to SPF, the action for a failed verification of the message cannot be specified in DKIM.

In contrast to other security technologies to provide authenticity and integrity from user to user, such as Secure/Multipurpose Internet Mail Extensions (S/MIME) [18] or Pretty Good Privacy (PGP) [3], DKIM is meant to work between sending and receiving mail server. Therefore, the user does not need to manage its own certificates, but the whole validation and key management can be done by the administrator of the mail server for the domain [1]. Additionally to DKIM, S/MIME and PGP can be used for end to end encryption of the message.

DMARC Domain-based Message Authentication, Reporting, and Conformance (DMARC) [13] is based on SPF and DKIM. Each domain can publish one DMARC policy. The DNS record describes what should happen with an email if the SPF or DKIM verification fails. Emails can either be accepted, rejected or flagged for quarantine. Additionally, DMARC provides a way to report failed verifications from the receiving to the sending mail server.

Figure 2 shows an DMARC record. Each record contains information about the DMARC version (v tag), the action to take when a check fails (none, quarantine, reject; p tag), and the email address where failed checks should be reported to (rua tag).

```
_dmarc IN TXT ( "v=DMARC1; p=quarantine; "  
                "rua=mailto:dmarc-feedback@example.com; " )
```

Figure 2. Example of a DMARC DNS record [13]

3 Mailing Lists

A mailing lists is a set of subscribers. Usually every electronic mailing lists has a public email-address. If a message is send to this email-address, the message is send to all subscribers. Mailing lists are often used for discussions or sending newsletter to the subscribers.

Figure 3 shows the differences between sending a email directly to a receiptent (first flow diagram) and using a mailing list (second flow diagram). For sending a message dirctly, the sender (Alice) sends the email, using the SMTP or Hypertext Transfer Protocol (HTTP) protocol, to the MTA of the sender domain (a.com). Next, the sending MTA uses the SMTP protocol to deliver the message to the MTA of the receiver domain (b.com). Finally, the reveiver (Bob) downloads the message from the receiving domains MTA via the Internet Message Access Protocol (IMAP), Post Office Protocol (POP), or HTTP protocol. In contrast to sending the message directly to the receiver, a mailing list uses a Mailing List Manager (MLM) to rely the messages to the subscribers. In the first step the sender (Alice) also sends the email to the MTA of the sender domain (a.com). Afterwards, the sending MTA sends the email to the MLM of the mailing list. The MLM can now modify the message and forwards it to addresses that are subscribed to the mailing list.

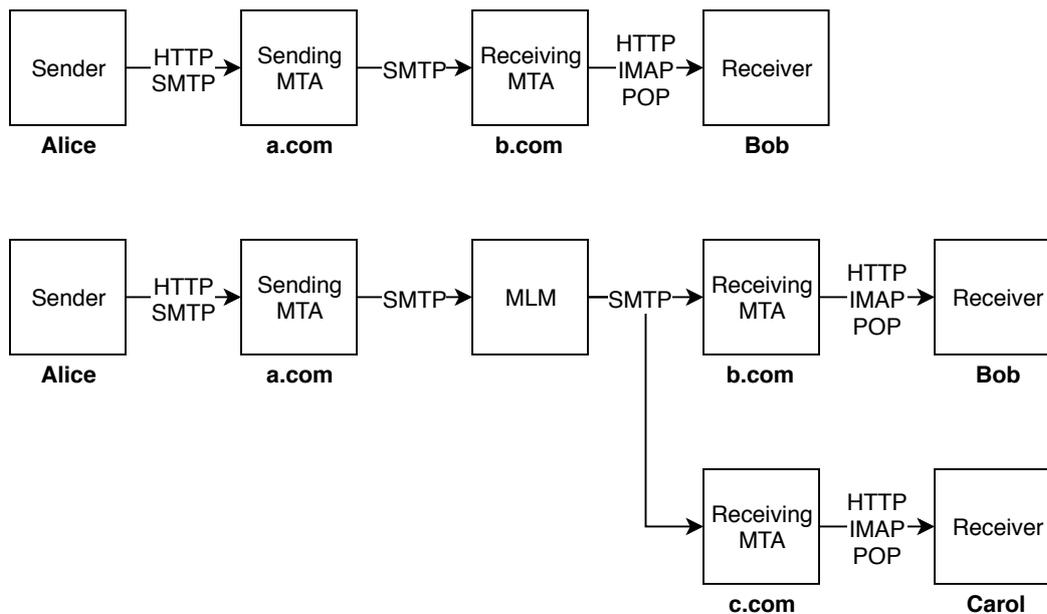


Figure 3. Difference between sending a email directly to using a mailing list [8]

In many cases, the email is changed by the MLM [15]. The following information are typically changed by a mailing lists:

- prepending the name of the mailing lists to the header field, to make it easier for the receiptent to identify the email coming from the mailing lists.
- appending a footer to the body of the email for managing setting for the mailing lists, such as a unsubscribe link.
- removing attachments or reformatting the message to have a consistant layout for all messages.
- encrypting or signing the message using PGP or S/MIME.
- allow moderators to enforce list content policies.

4 Usability Problems with Mailing Lists

For sending emails from one user to another, the authentication methods SPF, DKIM, and DMARC mentioned above are straightforward to implement. For a valid email, the receiving MTA can verify that the sending MTA sent the message and it was not modified during transport.

However, one problem occurs while implementing a mailing list that relays emails to multiple receivers, because these often change the message. When a message is changed during transport, the SPF and DKIM signatures become invalid [15]. Following, we describe the reasons why the checks of SPF and DKIM fail when they are used with mailing lists.

SPF To validate an email using SPF, the receiving mail server first queries the DNS record of the sending domain, as shown in Figure 4. The DNS record in this example allows only the mail servers of the `example.org` domain to send emails for the `example.com` domain. After querying the DNS record, the receiving MTA checks if the IP address of the sending email server is included in the list of authorized mail servers.

If the mail is relayed by a mailing list, the IP address of the sending mail server changes from the IP address of the original sender to that of the mail server of the mailing list. Because the IP address of the mailing list might not be in the list of authorized IP addresses for the domain of the sender, the SPF check fails [10, 7].

```
example.com. TXT "v=spf1 mx:example.org -all"
```

Figure 4. Example of a SPF DNS record [10]

DKIM In order to validate the DKIM record of an incoming email, the receiving mail server also queries the DNS record of the sending domain to get the public key and the signing algorithm of the domain's mail server. After this, the receiving mail server verifies the signature of the DKIM-Signature header of the email. As shown in Figure 5, this DKIM-Signature header contains the signing algorithm (a tag), the header field which is used as an input for the signature (h tag), the hash of the body (bh tag), and the signature (b tag). The version (v tag) and the selector for subdividing the namespace of the domain (s tag) are also mandatory fields. The signature is verified using the public key of the sender.

If an email from a DKIM enabled MTA is sent, the body of the message or the header fields can not be modified by the mailing list without breaking the signature, because these fields are protected by the signature. Furthermore, the MTA of the mailing lists can not create a valid signature for the message, because it does not know the private key of the sending domain [4].

```
DKIM-Signature: v=1; a=rsa-sha256; d=example.net; s=brisbane;  
h=from:to:subject:date;  
bh=MTIzNDU2Nzg5MDEyMzQ1Njc4OTAxMjM0NTY3ODkwMTI=;  
b=dzdVyOfAKCdLXdJ0c9G2q8LoXSlEniSbav+  
↪ yuU4zGeeruD00lszZVoG4ZHRNiYzR
```

Figure 5. An example DKIM signature [4]

5 Solutions

In this section, five solutions for using mailing list with enabled SPF, DKIM, and DMARC policies are explained.

Do not change the body of message In 2011, the IETF published recommendations to use DKIM with mailing lists [12]. In their document about best current practices, they suggest using mailing lists that do not change the body of the message but add all additional data as email headers.

Warn subscribers Additionally, the IETF described a solution if a mailing list provider does not want to follow the IETF recommendation of not changing the body of the message. For some providers, the change of the body is necessary because they want to attach advertisement or unsubscribe links. In this case, the IETF recommend warning new subscribers if the policy of the subscriber domain could cause problems with the mailing list [12].

Change sender to mailing list As Yahoo changed their DMARC policy to reject all messages with an invalid SPF or DKIM check in 2014 [14], researchers developed a solution to continue using mailing list without changing much on the mailing list [9, 19]. To continue using security policies, they suggested not to relay the email as before, but modify the message at the mailing list provider and create a completely new message with the mailing list as the sender. In this case, the mailing list provider can sign the message and DMARC checks would pass again.

Forward as attachment Another solution, mentioned by Huitema [9] in 2014, was to fix the problem with mailing lists in short term by forwarding the email as an attachment to bypass the security checks. In this case, the mailing list's MTA would create a new email with the original email as an attachment. The receiving client could then verify that the email came from the mailing list, but can also verify that the original email (attachment) came from the sender. To provide full authenticity, the client MTA needs to be aware of this practice.

ARC Authenticated Received Chain (ARC) [2, 8] is a protocol to solve the mailing list problem by creating a chain of trust for each mail server that handles the message during transport. Every ARC-enabled server attaches an ARC-Authentication-Results header to the message to provide information about which security properties were fulfilled at the time the server received the message. ARC is based on SPF, DKIM, and DMARC.

To generate the chain of trust, every ARC-enabled mail server checks the message using SPF, DKIM, and DMARC and attaches the results to the message. The authentication results are signed with the private key of the intermediate MTA. Similarly to DKIM, specified header fields and the body of the message are also signed and attached to the email by every mail server during transport.

This provides a way to change the body of the message without triggering a DKIM fail, because the server that modified the message can prove that the message was valid at the point it received it. The SPF verification is also solved by this protocol, because every MTA attaches a header to prove that the signature of the sender was correct when it received the message.

6 Discussion

In this section, we compare and discuss the five solutions from the previous section.

Do not change the body of message Although the recommendations of the IETF are from 2011, they are still a good solution to solve the mailing list problem, because all current security additions are not affected by mailing lists that do not change the body or header fields of the email. By adding the additional information as header fields, features like unsubscribing from a mailing list can be implemented.

The unsubscribe feature was already standardized in 1998 [16] and is implemented in some email clients. Nevertheless, these header fields are not supported by all email clients [20] and therefore most of the mailing lists still append the unsubscribe link to the body of the email. Other features like reformatting the body of the message or appending additional information of the mailing lists are not standardized or implemented. Furthermore, appending Multipurpose Internet Mail Extensions (MIME) attachments, such as images, in header fields of the email does not work and would require changing the body of the message. Although the solution sounds easy, it can not be used by all mailing lists because it restricts the usage of features like reformatting the body or appending attachments.

Warn subscribers Warning subscribers if the email might not be delivered due to possible verification failures is generally a good idea, but not a long term solution for the problem. Additionally, this procedure does not take into account that the policies of the domain might change. To solve this issue, the mailing lists must check the current policy of every subscriber domain before delivering the message. This would cause an overhead due to additional DNS queries.

All in all, warning the subscribers is user-friendly and should be done if the mailing list provider knows that the targeted group of receivers does not use the security standards with strict policies. As soon as the majority strict policies, the mailing lists should implement a solution that solves the problem in the long term and for all possible subscribers.

Change sender to mailing list Moderating incoming email at the mailing lists before sending them to the subscribers would solve the security issues because the mailing list can then sign the message itself. Additionally, SPF checks would pass because the mailing lists server can be configured to be allowed to send emails for the domain of the mailing list. Nevertheless, this solution can not provide authenticity for the original sender of the email, because the original sender is completely removed as the sender.

Forward as attachment Forwarding the email as an attachment solves the above-mentioned authenticity problem of the original sender because the recipient can verify the sender and signatures of the original message. Although, this solution is not user-friendly because email attachments are not opened automatically and therefore it needs additional user interaction to read the email from the mailing list. Another way of solving this problem would be clients that are aware of mailing list and detect such attachments to display them in a user-friendly way.

ARC The currently most promising solution for the mailing lists problem is the Authenticated Received Chain protocol. The protocol was specially designed for the use case of mailing lists and solves all known problems with SPF, DKIM, and DMARC for mailing lists. It also allows mailing lists to modify the body or header fields of an email before relaying it to the subscribers.

Although Google and other providers already implemented the protocol [2], it will take some time until most of the mailing list providers have updated their lists and clients implemented the final verification.

7 Conclusion

The SMTP protocol has the additions Sender Policy Framework (SPF), DomainKeys Identified Mail (DKIM), and Domain-based Message Authentication, Reporting, and Conformance (DMARC) to protect the authenticity of the sender of an email as well as the integrity of the message itself. While all of these standards protect against unwanted modifications of an email for normal email sending workflows, these protection mechanisms do not work for emails that were sent through mailing lists. When using mailing lists, the validation of these security additions may fail and create false positive spam messages. In this paper, we described the reasons why the validation of these additions fail with mailing lists and discussed possible solutions. Especially the new Authenticated Received Chain (ARC) protocol will help to use email authentication mechanism for mailing lists in the future. As the low adoption rates of DKIM and DMARC has shown, new protocols need many years to be adopted by enough providers to use them for filtering emails and avoid false positive spam messages.

References

- [1] Eric Allman. E-mail authentication: What, why, how? *Queue*, 4(9):30–34, November 2006.
- [2] K. Andersen, B. Long, S. Blank, and M. Kucherawy. Authenticated Received Chain (ARC) Protocol. Internet-Draft draft-ietf-dmarc-arc-protocol-23, Internet Engineering Task Force (IETF), December 2018. Work in Progress.
- [3] J. Callas, L. Donnerhacker, H. Finney, D. Shaw, and R. Thayer. OpenPGP Message Format. RFC 4880, Internet Engineering Task Force (IETF), November 2007.
- [4] D. Crocker, T. Hansen, and M. Kucherawy. DomainKeys Identified Mail (DKIM) Signatures. RFC 6376, Internet Engineering Task Force (IETF), September 2011.
- [5] I. D. Foster, J. Larson, M. Masich, A. C. Snoeren, S. Savage, and K. Levchenko. Security by Any Other Name: On the Effectiveness of Provider Based Email Security. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security, CCS '15*, pages 450–464, New York, NY, USA, 2015. ACM.
- [6] P. Hoffman. SMTP Service Extension for Secure SMTP over Transport Layer Security. RFC 3207, Internet Engineering Task Force (IETF), February 2002.
- [7] H. Hu, P. Peng, and G. Wang. Towards Understanding the Adoption of Anti-Spoofing Protocols in Email Systems. In *2018 IEEE Cybersecurity Development (SecDev)*, pages 94–101, September 2018.
- [8] H. Hu and G. Wang. End-to-end Measurements of Email Spoofing Attacks. In *Proceedings of the 27th USENIX Conference on Security Symposium, SEC'18*, pages 1095–1112, Berkeley, CA, USA, 2018. USENIX Association.
- [9] C. Huitema. DMARC or not, can email evolve? <https://huitema.wordpress.com/2014/04/21/about-dmarc-or-can-email-evolve/>, April 2014. Accessed: February 2018.
- [10] S. Kitterman. Sender Policy Framework (SPF) for Authorizing Use of Domains in Email, Version 1. RFC 7208, Internet Engineering Task Force (IETF), April 2014.
- [11] J. Klensin. Simple Mail Transfer Protocol. RFC 5321, Internet Engineering Task Force (IETF), October 2008.
- [12] M. Kucherawy. DomainKeys Identified Mail (DKIM) and Mailing Lists. RFC 6377, Internet Engineering Task Force (IETF), September 2011.
- [13] M. Kucherawy and E. Zwicky. Domain-based Message Authentication, Reporting, and Conformance (DMARC). RFC 7489, Internet Engineering Task Force (IETF), March 2015.
- [14] J. Levine. Yahoo breaks every mailing list in the world including the IETF's, April 2014. https://mailarchive.ietf.org/arch/msg/ietf/J-IsfA0Lb-6T_NeMD1ENKZyb9tA.

- [15] F. Martin, E. Lear, T. Draegen. Ed., E. Zwicky, and K. Andersen. Interoperability Issues between Domain-based Message Authentication, Reporting, and Conformance (DMARC) and Indirect Email Flows. RFC 7960, Internet Engineering Task Force (IETF), September 2016.
- [16] G. Neufeld and J. Baer. The Use of URLs as Meta-Syntax for Core Mail List Commands and their Transport through Message Header Fields. RFC 2369, Internet Engineering Task Force (IETF), July 1998.
- [17] J. B. Postel. Simple Mail Transfer Protocol. RFC 821, Internet Engineering Task Force (IETF), August 1982.
- [18] B. Ramsdell and S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. RFC 5751, Internet Engineering Task Force (IETF), January 2010.
- [19] J. A. Sherry. DMARC broke your mailing lists! <https://blog.jasonsherry.net/2014/04/10/dmarc-broke-your-mailing-lists/>, October 2014. Accessed: February 2018.
- [20] B. Specht. The Ultimate Guide to List-Unsubscribe. <https://litmus.com/blog/the-ultimate-guide-to-list-unsubscribe>, November 2018. Accessed: February 2018.
- [21] V. van Dongen. Verifying email security techniques for Dutch organizations. Master's thesis, University of Amsterdam, August 2018.

A Survey on Fallback Authentication Mechanism

Ruiyang Ding

ruiyang.ding@aalto.fi

Tutor: Siddharth Rao

Abstract

Fallback authentication or recovery mechanisms allows the users to regain control over their online services when one or more of the authentication factors are lost or forgotten. During this seminar course, we did a survey on different fallback authentication mechanisms to analyse each mechanism's advantages and disadvantages. Besides, we also find the improvement of these mechanisms which have been already in use in the industry. In addition, we did a survey on the current research of new fallback mechanisms, and make an introduction to two frontier fallback mechanisms which proposed by researcher in recent year.

KEYWORDS: Security, Authentication, Human-Computer Interaction

1 Introduction

The fallback mechanism is critical for any system which needs user authentication, since it is common for people to forget their credentials, for example, passwords. With the fallback mechanism, the user can log into the system without knowing the credential. Besides, the user can replace the forgotten credential after passing the verification of the fallback mech-

anism.

One commonly used solution on many websites is personal knowledge questions. The website will require the user to answer some personal question in order to retrieve the credentials. But with the personal information being more and more easily get revealed to the internet, this mechanism is not as safe as it used to be[1]. Besides, according to the research[1], and almost none of the website which using this mechanism allows the user to define the questions or utilise the SMS. Then, today, most websites are using email or SMS as the way to identify the user and lead the the user to reset his/her credential, which called Out-of-band communications[2]. When a user making a request on the website, a link or code will be sent, and the user can find the credential back with the link/code. Compare with the question, this mechanism has a higher success rate. Another commonly used way is cooperation of a trusted person, a mechanism which requires other users to give a response which shows the person who is using fallback mechanism is trustable. Facebook[2]. and Wechat forms China are using such mechanism[3].

In this paper, we will summarise the current popular fallback authentication mechanisms, analyse their advantages and disadvantages. Additionally, we will summarise the cutting-edge fallback authentication mechanisms proposed by computer scientists.

2 A Summary of Fallback Authentication Mechanisms

In this section, we sum up several fallback authentication mechanisms which are used on the internet, and also discuss the advantages and disadvantages of these mechanisms.

2.1 Personal Knowledge Question

Introduction

Personal knowledge questions or security questions are currently one of the most widely used fallback authentication mechanisms. Users are required to provide answers to preset questions which related to their personal information. Once the user forgets the password, it can re-access the account by correctly answer the question it set before.

Advantages

The personal knowledge authentication offloads the security risk in the user's side to the people who design the security question. Besides, The people who design the questions usually be more experienced in security issue than the normal usage. Besides, in this mechanism, the only decision that user need to remember is which question to choose, leaving the main decision to mechanism designer[1]. In conclusion, by doing this, the system can be relatively secure if the questions is properly designed.

Disadvantages

The personal knowledge question mechanism has several major defects since this system is intricate, there are many criteria which affect the usability of the system and even the safety of the whole website.

The first criteria is the privacy, the user will store a lot of privacy information on the server, which means this information is in danger of being leaked by the website intentionally and unintentionally. The second one is security, which is how to prevent attacks that what to illegally get the answers to the security questions.

Specifically there are two criteria[4] of security: Guessing difficulty and Observation difficulty.

Guess difficulty indicates how difficult for an attacker to guess the answer to the question. Under these criteria, the question with a very common answer is very easy to guess the answer. For example, in Latvia, near half of population[5] live in Riga, so in a website which mainly used by Latvian users, if the security question is about the birthplace or residency, it's very easy to guess the answer. Observation difficulty[4] means the information shouldn't be available to the public. For example, the security question is the information which could be searched on the internet.

Apart from the security problem. Usability is another problem of personal knowledge question. The usability mainly reflects on the memorability and the accuracy of the question.[4]

The memorability means how easy for the user to remember the answer. A very common situation is that the user forgets the password as well as the answer to the security questions. Under this situation, it is impossible for the user to find the account back. In another aspect, personal knowledge question means the user needs to remember additional information other than password. So, in practice, this mechanism has to

have another fallback mechanism as a backup. According to the survey, the success rate of personal knowledge question recovery is significantly lower than recovery with SMS/Email[6], so the backup mechanism will be more frequent user. The backup mechanism, which usually be costumer service, which will increase the company's expense by recruiting more people. The accuracy means that the question should have a few syntactic representations[4]. For example, in the answer to the question "What is your home address?", the street could be written as St. or street. Under these circumstances, it is very possible for the user to garble the actual answer.

Conclusion

Based on the criteria discussed above, we can make the conclusion that the question design in the personal knowledge question is intricate: to prevent the stolen of the account, the service designer needs to fulfil the privacy, security and usability at the same time. Additionally, the inappropriate question will cause the user to forget the correct answer, and cause the user to lose the account. Due to these reasons, there have new mechanisms emerged to replace personal knowledge question mechanism, which will be discussed below.

2.2 SMS/Email recovery

Introduction

Another very popular mechanism is an SMS/Email recovery. The mechanism is: When a user registers an account, it needs to leave his/her security email or phone number to the service provider. When the user forgets his/her password, it could let website send a recovery message to it via SMS/Email, which contains a link to recover the account, for example, a link to reset user's password.

Advantages

There are some advantages to this mechanism. Firstly, compared with personal knowledge question mechanism, the user doesn't need to remember additional information in order to recover his/her account the user also release the burden of remembering the information for account recovery. Additionally, the SMS/E-mail recovery avoid the risk of privacy leak[7], by avoiding store user's personal information on the server. Furthermore, although anyone can steal the account by the recovery link sent to the user,

it's nearly impossible to guess the link. In this way, this mechanism can also avoid the guessing attack. Furthermore, the content of the recovery message is randomly generated link/code. Or the content can even be a temporary password[8]. Compared with the fixed answer to the personal knowledge question, the randomly generated content is safer.

Disadvantages

This mechanism has its flaws. Firstly, the webmail services and device which receives SMS may insecure. The email is also a kind of online service which using the password and has its own fallback authentication mechanism. If the mail service provider doesn't secure the server properly, or it has a poorly designed, the attacker could get the recovery link by hacking into the user's mailbox. On the other hand, SMS, which seems more secure, also has its own security problem due to its store and forward feature.[9].Store and forward feature is a telecommunication technique which the message will be forwarded by other devices for multiple times before reaches its destination.[10] These devices will also store the message when forwarding it. As the result, due to this feature, it is possible for the attacker to intercept the message. The SMS spoofing[11] is another major security vulnerability of SMS. It is a kind of technology which fakes the sender of the SMS and often occurs when the user is on roaming[12].When the user is waiting to receive the recovery message, the attacker could execute SMS spoofing attack by fake itself to be the website that the user lost the credential. Then, the attacker can send an SMS contains a link which could cheat the user's login credential by the phishing attack. Besides, the SMS is poorly encrypted by stream cipher[13]. Under this case, it is easy to decrypt the message once the attacker intercepts it. Besides, when the user is roaming, the SMS may go through the internet, makes it more easily to be intercepted[9].

Another concern is that both E-mail and SMS are very easy to be faked and sent from the internet. In some case, the user may need to input the new password in the account recovery process. The attacker could use the phishing attack by faking a website, using a very similar domain, try to get the user's password. Through this can be avoided by look thoroughly to the web page and the domain, but still, many users may get their password leaked.

Other than the security problem, this recovery method may become invalid when the user changes his/her contact.This problem is more obvious

in SMS recovery. The user may change his/her phone number after relocates to a new country, if user forget to change the phone number then he/her may have to ask customer service for help or even lost the account.

Improvement

Due to the problems mentioned above, in practice, some software tries to improve the mechanism by avoiding using SMS in the process. In this section, we take Telegram for example to analyse the possible improvement for the mechanism.

If the user forgets the password to his/her Telegram account, but he/she has the same account already logged in another device. In this situation, when the user initials the fallback authentication mechanism, the system will send the recovery code directly to his/her telegram which is already logged in at another device. This improved mechanism is safer since it avoids the SMS which is not as secure as the encrypted communication through Telegram.

Conclusion

In conclusion, SMS/Email recovery is a more convenient solution for both users and service providers, but it has a lot of vulnerabilities which couldn't be ignored, especially in the SMS channel. But luckily, some alternative methods are already implemented by some companies to replace the SMS channel. These methods bring better security to SMS/Email recovery mechanism.

2.3 Trusted Friends (TF) Mechanism

Introduction

In TF mechanism, the user is authenticated by contacts[14]. This kind of approach raised by the development of the social network. When the user requests to reset his/her password, the user will need to select some contacts from his/her contact list. After this, the system will send randomly generated code to each selected contacts. These contacts are supposed to give user these codes through a communication channel. It will either give it to the user via an alternative channel or just sent this code through chat with the user in the platform which the user needs to recover his/her account(Wechat).

Advantages

The TF mechanism is relatively secure. If the attacker tries to get access to the user's phone with this method, he must first know which people in the contact list is user's trust person. Besides, even he knows who is the trusted person, he must instruct the user's trusted person to help with the account recovery[2]. In this process, the user's trusted person will easily found out the true identity of the attacker. This because the attacker can only contact with an unknown identity, either call from an unknown phone or find the user's trusted people from social media and contact them as a stranger. In this case, then making the phone call, the attacker is very easy to be exposed from the voice. On the other hand, people don't trust the stranger form social media, so it is very hard for the attacker to convincing. According to scientific research and servery, the TF mechanism has low usability but with very high security[2]. So in general, the TF mechanism is safe.

Disadvantages

But the attacker can still have possibility to broke this mechanism. The attacker could fake several accounts under his/her control[14], add the victim as friend. Then attacker can start the password recovery process from the victim's side, select his fake account and send the code. By this way, it is possible for the attacker to take control of the victim's account.

Improvements

To prevent such problem, the companies actually adopt this mechanism usually take some additional measure. Facebook is one of the websites using this mechanism. When the user tries to execute the account recovery process, Facebook only allows the user to select recovery contacts from a certain group of contacts. This group usually selected from the contacts that user recently or often contacts with. Some software, like Wechat(A Chinese chatting application from Tencent), uses this mechanism just as an additional security mechanism[3]. In the account recovery process of Wechat, the user's account is blocked even it passes the SMS verification, the user is required to select 3 contacts from the list. After this, these 3 users will receive a unique code, they will be required to send the code to the blocked user to unblock the account.

Conclusion

In conclusion, we hold the opinion that the Trusted Friends mechanism is a relatively safe fallback authentication mechanism. It can defend the user's account from most of the attack. It has a minor disadvantage which could overcome by adding an additional security mechanism.

2.4 Recovery by Contact Support Team

Introduction

This is a method commonly offered in the user system with a small user group, for example, the user system inside a school or a company. When the user forgets the credential, he/she could contact the support team to reset the credential.

A good example is Aalto University[15]. Normally, if the user forgets the password, he/she need to log in to the system with bank credential to reset the password. But, but, for some of the non-Finnish student, the bank credential is unusable. So If a non-Finnish user forgets the password for his/her Aalto IT account, he/she may recover the account by this method. First, fill the contact form on the IT service web page, and then visit the IT centre with ID. With this way, the user can retrieve the access to his/her account.

Advantages

Compared in internet, it is more hard to fake to be someone else in a face to face communication which require people to show their ID. In other case, the only way to hack this system form the outside is to make the staff in support team believe the attacker is the victim, which is very hard.

Disadvantages

On the other hand, the downside of this method is that it needs too many people to maintain this system. In a system with a very large user group, it just has a large support team to make the response fast enough. Because of this, it will be extremely expensive to use such fallback authorisation system.

Besides, the support team members with vicious intentions are actually most dangerous attacker. Compared this mechanism with the previous mechanisms, the human has more impact. So, a staff in the support team can easily get control of any account in the system, which is a threat that

cannot be ignored.

Conclusion

In conclusion, the Recovery by Contact Support Team method is a good option for the smaller user group, it could prevent all of the attacks from the internet, which makes it the safest fallback authentication mechanism for the small user group, for example, inside an organisation or company. On the other hand, the high need for manpower also makes this mechanism very expensive, so it is not suitable for the larger user group.

3 The Latest Researches on Fallback Authentication Mechanisms

In this section, we sum up the latest research results on fallback authentication mechanisms. These research result is based on current fallback authentication mechanisms which brings better security and usability.

3.1 Dynamic Security Question

Dynamic Security Questions for Fallback Authentication is a fallback authentication proposed by Alina Hang, Alexander De Luca, Heinrich Hussmann at their paper[16] published in 2015. And this mechanism is being used if the user forgets the password of his/her phone. In their fallback authentication mechanism design, once the user lost the access to the mobile device by forgetting the password, the mobile device will generate some dynamic questions based on the use history of the phone. After correctly answer these questions, the user can regain access to the phone.

The question is based on information collected from an Android application running on background. Which will record the user's calling record, messaging record, application usage history, etc. To ensure the privacy of the user, the data will not be left the phone. Then, when the user tries to find back their password, the system will pick up the random record, generate a question-based on the log.

In the test, this mechanism has a high success rate: user can answer 74% of question correctly, which is relative high[16]. On the other hand, the mechanism still has its problems. First, is if the user very rarely uses the phone, there may not have enough data to generate the question. Besides, if the attacker knows the user very well, he/she may answers all of the questions correctly. Moreover, there is much data which is very

hard to access for the data collection application, for instance, the data inside Whatsapp[16].

3.2 Gamified Security Question

In the research paper[17] of Nicholas Micallef and Nalin Arachchilage, they proposed a game design which aims to improve the usability of question-based fall-back authentication mechanism.

The researchers adopted the "4 Pics 1 word" game. This is due to the reason the game, which uses picture and cues, helps user a lot with memory[17]. The game is used in the mechanism, which automatically generate the answer to the security questions. Because the answer is only known to the user, and it is not related to the user's personal information, thus this method is more secure than personal knowledge question. Under this condition, is important and also hard for the user to remember the answer. The game aims to help the user remember the answers to the questions by creating a strong association with answers for the user.

In the game, there will be 4 pictures they all related to the same word, which is the answer to the security question. The system will provide user part of the word, and user needs to fill the rest of the letters of this word. The game will be repeated after a certain period of time to enhance the user's memory.

The game is effective according to the previous psychology research, that the repeating[18] and image[19] can help the user to remember a word better. Besides, it adds the pointing system to make the user more engaged in the game, which makes it more effective[17].

4 Conclusion

In the paper, we summarize current fallback authentication mechanism, analyse their advantages and disadvantages, as well as have a survey on the current improvement of these mechanisms that could overcome their disadvantages. The current mechanisms has some shortages, still tech companies already had some methods to make up these disadvantages. Besides, we also do survey on the recent researches of fallback authentication mechanisms, and summarize the potential improvement over current mechanisms that these researches bring to us.

References

- [1] Ariel Rabkin. Personal knowledge questions for fallback authentication: Security questions in the era of facebook. In *Proceedings of the 4th Symposium on Usable Privacy and Security*, SOUPS '08, pages 13–23, New York, NY, USA, 2008. ACM.
- [2] Vlasta Stavova, Vashek Matyas, and Mike Just. Codes v. people: A comparative usability study of two password recovery mechanisms. In *IFIP International Conference on Information Security Theory and Practice*, pages 35–50. Springer, 2016.
- [3] Silas. 4 ways to verify wechat on new device, Feb 2019.
- [4] M. Just. Designing and evaluating challenge-question systems. *IEEE Security Privacy Magazine*, 2(5):32–39, 2004.
- [5] Riga - wikipedia. <https://en.wikipedia.org/wiki/Riga>. (Accessed on 03/11/2019).
- [6] Joseph Bonneau, Elie Bursztein, Ilan Caron, Rob Jackson, and Mike Williamson. Secrets, lies, and account recovery. *Proceedings of the 24th International Conference on World Wide Web - WWW 15*, 2015.
- [7] Maximilian Golla and Markus Dürmuth. Analyzing 4 million real-world personal knowledge questions (short paper). In *PASSWORDS*, 2015.
- [8] Password recovery | connectnetwork. <https://web.connectnetwork.com/help/password-recovery/>. (Accessed on 03/11/2019).
- [9] M. Toorani and A. Beheshti. Ssms - a secure sms messaging protocol for the m-payment systems. *2008 IEEE Symposium on Computers and Communications*, 2008.
- [10] M. Thomas and V. Panchami. An encryption protocol for end-to-end secure transmission of sms. In *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]*, pages 1–6, March 2015.
- [11] Graeme Horsman and Lynne R. Conniss. An investigation of anonymous and spoof sms resources used for the purposes of cyberstalking. *Digital Investigation*, 13:80–93, 2015.
- [12] Sms spoofing - wikipedia. https://en.wikipedia.org/wiki/SMS_spoofing. (Accessed on 03/11/2019).
- [13] M. Toorani and A. Beheshti. Solutions to the gsm security weaknesses. In *2008 The Second International Conference on Next Generation Mobile Applications, Services, and Technologies*, pages 576–581, Sep. 2008.
- [14] Ashar Javed, David Bletgen, Florian Kohlar, Markus Durmuth, and Jorg Schwenk. Secure fallback authentication and the trusted friend attack. *2014 IEEE 34th International Conference on Distributed Computing Systems Workshops*, 2014.
- [15] Resetting the password, and a forgotten password. <https://it.aalto.fi/instructions/resetting-password-and-forgotten-password>. (Accessed on 03/25/2019).

- [16] Alina Hang, Er De Luca, and Heinrich Hussmann. I know what you did last week! do you? dynamic security questions for fallback authentication on smartphones.
- [17] A gamified approach to improve users' memorability of fall-back authentication. In *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017)*, Santa Clara, CA, 2017. USENIX Association.
- [18] John R. Anderson and Gordon Bower. Recognition and retrieval processes in free recall. *Psychological Review*, 79:97–123, 03 1972.
- [19] ALLAN PAIVIO, T B. ROGERS, and PADRIC C. SMYTHE. Why are pictures easier to recall than words? *Psychonomic Science*, 11:137–138, 04 1968.

A Comparison of Intelligent Edge Computing Platforms

Aleksandra Zhuravleva

aleksandra.zhuravleva@aalto.fi

Tutor: Behrouz Jedari

Abstract

Due to the exponential growth of mobile data traffic, the capacity of wireless and cellular networks should be expanded to cope with the ever-increasing demands mobile users. Mobile edge computing (MEC) has emerged as a promising solution in fifth-generation (5G) networks and Internet of Things (IoT) to bring the network resources as close as possible to mobile users, thus improves the resource utilization and user satisfaction (e.g., in terms of latency). In this report, we study intelligent edge computing platforms which provide the network resources (e.g., storage and computing) at the network edge (e.g., small-cells or even user equipment) to improve the performance of data delivery, and real-time data streaming and analytics in future 5G networks. In particular, we introduce three well-known platforms, namely (i) Azure IoT Edge, (ii) IBM Watson IoT and (iii) Amazon Web Services IoT Greengrass. Next, we discuss the architectures, features, and advantages of the platforms comparatively. The outcome of our research shows that the platforms have different architectures but they support a variety of certified hardware for IoT devices and modern technologies such as IoT protocols and programming languages for configuring devices and developing applications.

KEYWORDS: 5G networks, Internet of things, mobile edge computing, edge computing platforms.

1 Introduction

With the explosive increase of Internet of Things (IoT) devices and the user demands, network operators attempt to expand the capacity of the wireless and cellular networks to cope with the fast-increasing volume of mobile data traffic. In addition, novel multimedia applications and services have emerged which require high-bandwidth and low-latency Internet connectivity to satisfy the quality requirements of their subscribers (i.e., mobile users). Mobile edge computing (MEC) is a key network architecture and technology in fifth-generation (5G) networks which aims at bringing the network resource closer to end users, thus reduce the backhaul network traffic and improve user quality of experience (QoE).

The architecture of heterogeneous networks (HetNets) is known as a key networking model to enable MEC. In HetNets, multiple small-cell base stations (BSs) (e.g., Femto-cells or Pico-cells) are deployed in the coverage of macro BSs to provide efficient Internet services to mobile users. In contrast to traditional wireless networks, the combination of HetNets and MEC can enable new capabilities in next-generation wireless networks. For example, Zhang et al. propose a cooperative content caching architecture for 5G networks that enhances traditional approaches by leveraging MEC [23]. They introduce hierarchical mobility-aware edge caching strategies that keep popular data in BSs passed by mobile users. Zhou et al. demonstrate a powerful mechanism to integrate Heterogeneous Networks (HetNet) with Information-Centric Networking (ICN) and MEC techniques [24]. Real-world applications of MEC may include the following aspects: dynamic content optimization, computational offloading in IoT, Mobile big data analytics and smart transportation [11].

Although the above research makes the network infrastructure more flexible and efficient, they lack smart and sophisticated caching approaches to precisely identify content popularity and user requirements. However, big data analytics [7], deep learning [13], federated learning [21] offer broad perspectives to promote intelligent analysis of the delivered data. Chang et al. believe that big data analytics are indispensable in future networks [7]. For this purpose, they present machine learning schemes that are implemented on the network edges in order to analyze the content of delivered services and make accurate caching decisions based on

collected data features. Li et al. [13] consider deep learning as the most promising approach to reliably mine IoT data from the noisy and complex environment and to make accurate predictions about content edge placement.

Recently, some promising edge platforms have been deployed that provide real-time processing and intelligent data analysis services at the network edge (e.g., through small-cell BSs and even mobile devices). To meet high wireless data traffic demands, these platforms combine intelligent schemes for content caching and delivery, and high-performance streaming analytics approaches in the light of Big Data and IoT. Using these platforms relieves customers from writing own algorithms and offers existing and proven methods. It helps to easily develop and deploy solutions, capture and analyze data from sensors, mobile devices, applications, equipment, vehicles, appliances and all kinds of the IoT.

In this report, we study popular edge computing platforms that bring intelligence into the network edge and promise innovative ways of improving overall performance in future wireless systems. We discuss different types of platforms and architectures. In addition, we highlight their important features, advantages, and shortcomings. Our research may help researchers and practitioners to choose an appropriate platform to develop their services with respect to their features.

The rest of the report is organized as follows. Section II defines the edge computing architecture and platforms. Section III discusses architectures and features of popular Edge Computing Platforms such as Azure IoT Edge, IBM Watson IoT Platform and AWS IoT Greengrass. A comparison of platforms characteristics is presented in section VI. Section V concludes the article. Finally, appendices provide useful guides for considered platforms using dashboard or command line interface.

2 Mobile Edge Computing: Architecture and Platforms

The network infrastructure is a complex hierarchy of network segments ranging from user gadgets to corporate clouds and data centers. The structure of an IoT network can typically be divided into three aspects: sensors/devices, IoT gateways/local network, and backhaul network/cloud, representing the data source, data communication networks, and data processing, respectively [22].

Fig. 1 illustrates the hierarchy of these three network levels. The first

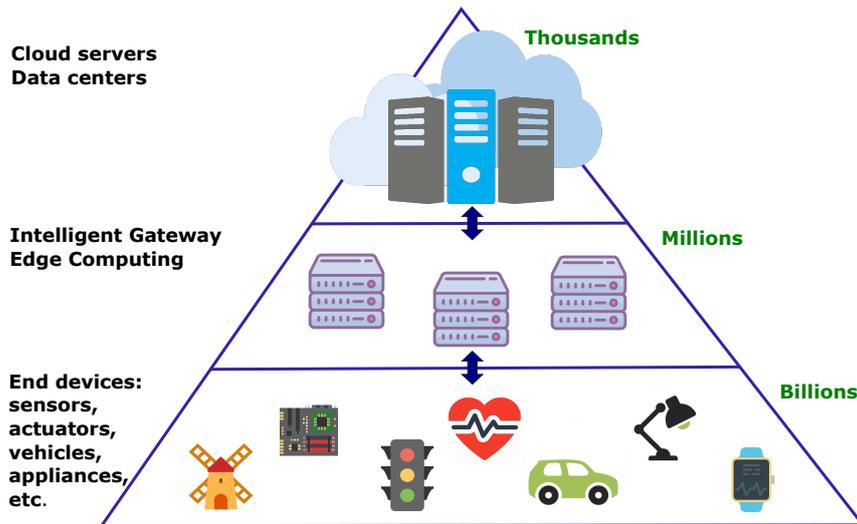


Figure 1. The structure of an IoT network

level is called *end devices* and is represented by sensors, actuators, vehicles, appliances, and other different types of IoT devices. The computational power of these various gadgets can provide real-time services for some applications. Nonetheless, due to the limited capacity of the end devices, the requirements of most applications cannot be fulfilled at the front edge environment [22]. In these cases, the end devices have to forward data for processing to the edge servers. On this level of IoT networks, gateways support the vast majority of the traffic flows and have more power and storage. Therefore, they can perform real-time data processing, data caching, and computation offloading. Such a structure allows user devices to achieve much better performance on data computing and capacity, with a small increase in the latency [22].

EC software allows data from applications, sensors, controllers to be analyzed at the edge of the network before being sent to a data center or cloud. This reduces the communications bandwidth required among gadgets and the central data center by performing analytics at or near the source of the data [12]. Working with Edge Computing Platform, a device does not constantly transfer data to the cloud for analysis, but instead, it is enabled to accomplish tasks on its own.

In this report, we study Azure IoT Edge, Watson IoT Platform, and AWS IoT Greengrass platforms. They give developers an ability to easily connect to IoT devices and cloud services using IoT industrial protocols, develop and manage edge computing applications, visually compose data

flow to analyze and route data [12]. EC models can be ported across different system hardware architectures, i.e., embedded in gateways or small footprint edge devices. Working in conjunction with centralized analytic systems, EC platforms can provide efficient and timely analytics across the whole IoT ecosystem: from the center to the edge.

3 Overview of well-known Edge Computing Platforms

3.1 Azure IoT Edge

Azure IoT Edge is an open source project introduced by Microsoft which is publicly available on Github ¹. This product offers intelligent edge technologies, fully hosted and managed Software as a Service (SaaS) solutions and specialized Platform as a Service (PaaS) solutions [1]. With Azure IoT Edge you can move custom application logic and cloud analytics to devices and then monitor work from the cloud.

In the big picture, end-to-end architecture with Azure IoT Edge contains three main parts. These are devices, Azure IoT Edge itself and cloud-based interface (Azure IoT Control Plane). Devices can be sensors or actuators that are connected to the local edge computing layer. Azure IoT Control Plane interacts with the edge layer, remotely monitors and manages IoT Edge devices.

Azure IoT Edge includes IoT Edge modules and IoT Edge runtime. Modules are built from standard Dockerfile containers. They execute Azure services, third-party services or own code on IoT Edge devices. Each device is assigned to a module registered with the edge layer, but not every module needs to be mapped to a device. IoT Edge runtime runs on each device, manages deployed modules and performs the interaction with Azure IoT control [15].

Azure IoT Edge Agent and Azure IoT Edge Hub are two components of IoT Edge runtime. The first one downloads the deployment manifest from the cloud and maintains the required state of a device's configuration. The second one provides offline capabilities of an IoT Hub in the public cloud by offering authentication and communication services to the devices [15]. Fig. 2 shows Azure IoT Edge architecture.

Artificial Intelligence (AI) on the edge can be performed by complex

¹<https://github.com/Azure/iotedge>

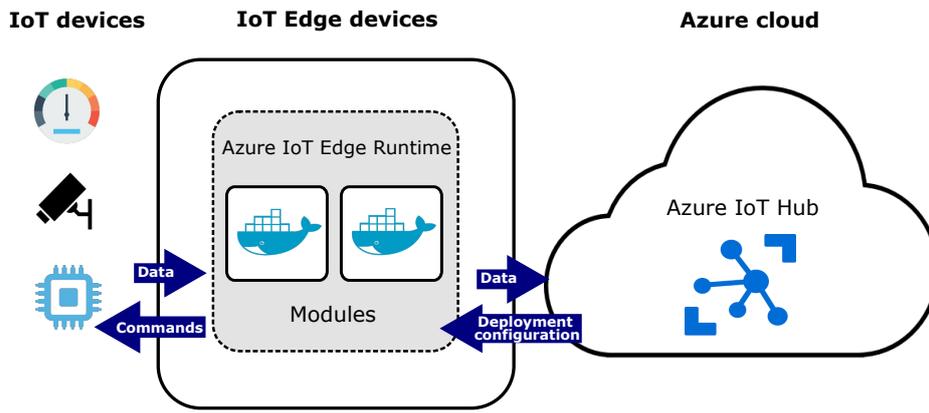


Figure 2. The architecture of Azure IoT Edge

event processing, machine learning, image recognition, and other high-value AI. In addition to offered services such as Azure Functions, Azure Stream Analytics, and Azure Machine Learning, we are able to implement own AI modules. For that purpose, Azure IoT Edge supports Java, .NET Core 2.0, NodeJS, C, and Python programming languages [3].

Microsoft Azure website provides a variety of useful tutorials, e.g., deploying Azure Stream Analytics as an IoT Edge module or performing image classification at the edge with Custom Vision Service. Below we describe an example of a quick start with deploying the first IoT Edge module to a Linux device.

3.2 IBM Watson IoT Platform

The IBM Watson IoT Platform (WIoTPlatform) together with Edge Analytics Agent (EAA) performs Edge services for devices and gateways. This platform provides powerful application access to IoT devices and data, device management operations, secure communication by using MQTT (Message Queuing Telemetry Transport) and TLS (Transport Layer Security) protocols [4].

To use this platform at the network edge, we need to create Edge nodes and configure services that will be deployed to these nodes. An Edge node consists of a gateway and a device that resides at the edge of the network. When services are executing on nodes, Edge nodes can send messages from devices with the Edge gateway. The gateway parses the messages relying on configured edge analytics rules. Critical data and alerts might be sent to Watson IoT Platform, trigger an alert on the gateway, or be

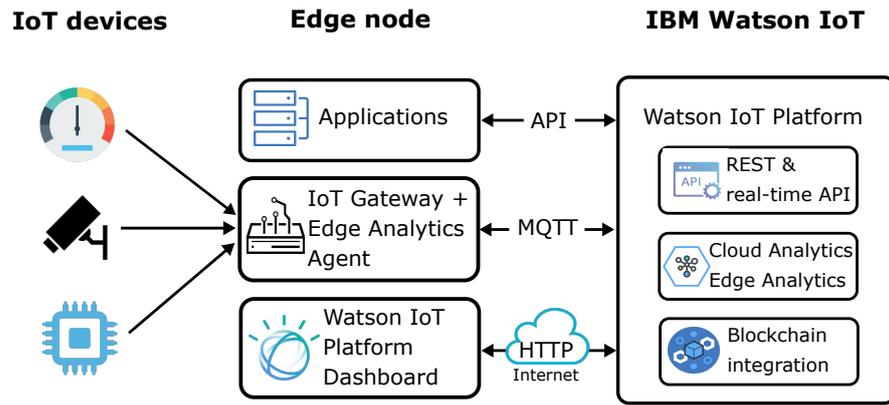


Figure 3. The architecture of the Watson IoT Platform Edge.

written to a text file that is local to the gateway [9]. The diagram on Fig. 3 demonstrates the general architecture of the Watson IoT Platform edge analytics environment. WIoTP acts there as asynchronous glue between all components in the IoT operational model.

Now we are going to describe in details each component from the presented diagram, namely, devices, gateways, applications, MQTT and IBM add-ons solutions such as Watson IoT Platform on Blockchain and Watson IoT Platform Analytics.

A device is anything that has a connection to the Internet and that can push data into the cloud [4]. Instead of direct communication with other devices, devices receive commands from applications and send events back to applications. Each device in the WIoTP has a unique authentication token and must be registered before it connects to the WIoTP. To build and manage devices, we can use such technologies as MQTT messaging protocol, Python, NodeJS, Java, C#, Embedded C and Mbed C++ [8].

Gateways serve as access points for other devices because they have combined capabilities of an application and device [4]. Gateways must also be registered before they can connect to the WIoTP.

An application can be anything that has a connection to the Internet, interacts with data from devices and manage the behavior of those devices. Instead of registering the application as in case of devices and gateways, we must provide valid API key and a unique application ID to identify the application in WIoTP. However, API key has to be previously registered.

MQTT is a lightweight protocol for events and messages that allows devices to effectively communicate across constrained networks to remote

systems [5]. This protocol supports different data formats, therefore, it is possible to send images, texts and almost every type of data in a binary format.

WIoTTP is expanded by such add-ons solutions as Watson IoT Platform on Blockchain and Watson IoT Platform Analytics. Blockchain enables IoT resources to participate in blockchain business networks [6]. Analytics extension provides built-in configurable analytics functions to easily interact with the raw data from IoT devices.

3.3 AWS IoT Greengrass

AWS IoT Greengrass platform provides cloud capabilities for different types of devices. For example, like other platforms, it enables devices to collect and analyze data closer to the source of information. Devices could vary in size, from smaller microcontroller-based devices to large appliances [17]. The platform allows to react autonomously to local events, and communicate securely with each other – even when not connected to the Internet. AWS IoT Greengrass protects user data through the secure authentication and authorization of devices, through secure connectivity in the local network and between local devices and the cloud [20].

Connected devices in AWS IoT Greengrass are able to execute predictions based on different ML models and keep device data in sync. AWS IoT Greengrass developers can use AWS Lambda functions and pre-built connectors to create serverless applications that are deployed to devices for local execution. With pre-built connectors, developers can easily extend edge device functionality without writing code. Building IoT devices and application logic can be done through cloud-based management that runs on devices. Locally deployed Lambda functions and connectors are triggered by local events, messages from the cloud, or other sources. [20]. To execute lambda code, device has to support ARM or x86 architectures on which AWS IoT Greengrass Core can be hosted [17]. Greengrass supports Lambda functions authored in Python language, NodeJS, Java, C, C++ [18].

The basic Greengrass architecture is presented in Fig. 4. The platform consists of Software distributions, Cloud service, and Features. Cloud service is called AWS IoT Greengrass API. Features include Lambda runtime, Message manager, Over-the-air update agent, Local resource access, Local machine learning inference, and some other useful features.

There are three different types of AWS IoT software distributions such

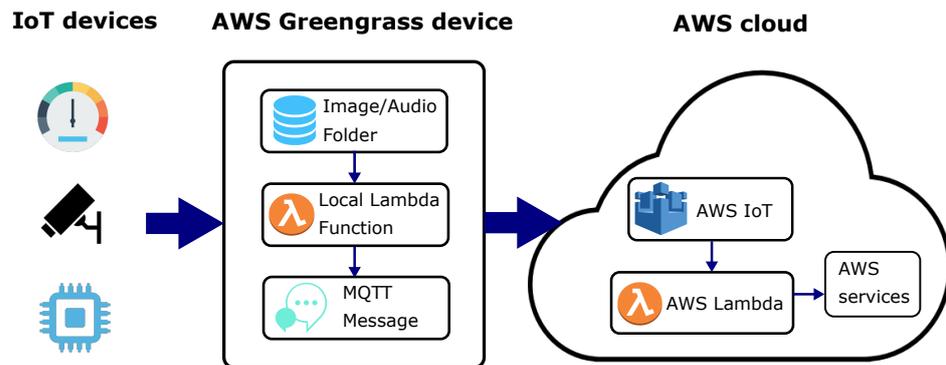


Figure 4. The architecture of AWS IoT Greengrass.

as Greengrass Core, Greengrass SDK and Device SDK [18]. AWS IoT Greengrass Core works on CPU-based devices (x86 or ARM) that run a general-purpose OS such as Linux. The purpose of this distribution is to provide local services (computing, messaging, security), and communicates locally with devices that run the AWS IoT Device SDK. Greengrass SDK allows Lambda functions to interact with local services inside a Lambda function deployed to AWS IoT Greengrass Core. Device SDK helps devices to interact locally with AWS IoT Greengrass Cores.

4 A comparison of Edge Computing Platforms

This section compares the platforms and presents key characteristics in the Table 1. Key characteristics include pricing, open source, data protocols, certified hardware, SDK/language, connection, analysis.

Pricing. All three platforms offer a free trial account for a period of time with limited functionality. After a free period, payments are calculated in different ways depending on the platform. Whereas AWS IoT Greengrass relies only on a total number of messages per day, both Azure IoT Edge and IBM Watson IoT calculate price from a combination of used components. These include a number of devices, data traffic and data storage.

Open source. Unlike IBM Watson IoT and AWS IoT Greengrass, Azure IoT Edge is an open source project available on Github. Everyone can see the code, develop desired behavior for yourself or contribute to the project fixing any of issues.

Data protocols. The platforms support HTTP (Hypertext Transfer Protocol) and MQTT data protocols. In addition, Azure IoT Edge allows

Characteristics	Azure IoT Edge	IBM Watson IoT	AWS IoT Greengrass
Pricing	Paying for IoT Hub unit related to number of devices and messages per days	Paying related to number of devices, data traffic and data storage	Paying for total number of messages/day per unit
Open source	Yes	No	No
Data protocols	HTTP, MQTT, AMQP	HTTP, MQTT, TLS	HTTP, MQTT, WebSockets
Certified Hardware	Texas Instruments, Intel, Raspberry Pi, Freescale,	Texas Instruments, Raspberry Pi, ARM mbed, Arduino Uno	Texas Instruments, Intel, Broadcom, Marvell, Renesas, Microchip
SDK/Language	Python, NodeJS, Java, C#, C	Python, NodeJS, Java, C#, Embedded C and Mbed C++	Python, NodeJS, Java, C, C++
Connection	Easy configuration	Difficult configuration	Easy configuration
Analysis	Medium configuration	Difficult Configuration	Easy configuration

Table 1. Characteristics of IoT platforms (adopted from [14]).

messages through AMQP (Advanced Message Queuing Protocol), IBM Watson IoT – through TLS, AWS IoT Greengrass – through WebSockets.

Certified Hardware. Texas Instruments hardware is supported by all the platforms. Similar to Azure IoT Edge, AWS IoT Greengrass provides compatibility with Intel hardware. Both Azure IoT Edge and IBM Watson IoT are able to work with Raspberry Pi hardware. Some other well-known types of IoT hardware are supported in the platform. More detailed lists are presented in the Table 1.

SDK/Language. While Python, NodeJS, Java and C are the common languages in the platforms, Azure IoT Edge and IBM Watson IoT supports C# language as well, and both IBM Watson IoT and AWS IoT Greengrass support C++.

Connection. The difficulty of connecting a device is measured on the time and difficulty of creating the gateway [14]. Azure IoT Edge and AWS IoT Greengrass, in contrast to IBM Watson IoT, provide easy configuration. You can see examples of creating a device with Azure IoT Edge and

IBM Watson IoT in appendices A and B respectively.

Analysis. This characteristic describes a degree of difficulty how stored data can be accessed and transferred to visualization analysis [14]. Similarly to the previous characteristic, IBM Watson IoT is the most difficult platform in analysis configuration comparing to the other two platforms. AWS IoT Greengrass provides easier configuration than Azure IoT Edge.

5 Conclusion

With the massive increase in the collected data from IoT devices, wireless networks meet challenges including increased demand for network resources and a decrease in the efficiency of content delivery. We presented an overview of popular edge computing platforms such as Azure IoT Edge, Watson IoT Platform and AWS IoT Greengrass which fulfill the diverse requirements of IoT applications and data inference. The paper discusses architecture, features, and benefits of the platforms. Promising approaches of these edge computing platforms combine intelligent schemes for content caching and delivery. It helps to enhance the performance of networks and provide low-latency transmission. The comparison of main characteristics shows that the platforms support a sufficient amount of certified hardware for IoT devices and plenty of modern technologies such as IoT protocols, SDK and programming languages. Azure IoT Edge and Amazon Web Services IoT Greengrass are slightly simpler than IBM Watson IoT for configuring IoT devices, developing and managing edge computing applications.

References

- [1] Microsoft Azure. Azure IoT Fundamentals. <https://docs.microsoft.com/en-us/azure/iot-fundamentals/>. Accessed: 2019-02-22.
- [2] Microsoft Azure. Register a new Azure IoT Edge device with Azure CLI. <https://docs.microsoft.com/en-us/azure/iot-edge/how-to-register-device-cli>. Accessed: 2019-02-28.
- [3] Microsoft Azure. What is Azure IoT Edge. <https://docs.microsoft.com/en-us/azure/iot-edge/about-iot-edge>. Accessed: 2019-02-22.
- [4] IBM Knowledge Center. About Watson IoT Platform Service. https://www.ibm.com/support/knowledgecenter/en/SSQP8H/iot/platform/iotplatform_overview.html. Accessed: 2019-02-23.
- [5] IBM Knowledge Center. MQTT standards. <https://www.ibm.com/support/>

- knowledgecenter/en/SSQP8H/iot/platform/reference/standards_and_requirements.html. Accessed: 2019-02-24.
- [6] IBM Knowledge Center. Product overview. <https://www.ibm.com/support/knowledgecenter/en/SSQP8H/iot/overview/overview.html>. Accessed: 2019-02-24.
- [7] Zheng Chang, Lei Lei, Zhenyu Zhou, Shiwen Mao, and Tapani Ristaniemi. Learn to Cache: Machine Learning for Network Edge Caching in the Big Data Era. *IEEE Wireless Communications*, 25(3):28–35, June 2018.
- [8] IBM Cloud. Developer documentation for Watson IoT Platform. https://console.bluemix.net/docs/services/IoT/developer_doc_overview.html. Accessed: 2019-02-24.
- [9] IBM Cloud. Edge analytics. https://console.bluemix.net/docs/services/IoT/edge_analytics.html. Accessed: 2019-02-23.
- [10] IBM Cloud. Getting started tutorial. <https://console.bluemix.net/docs/services/IoT/getting-started.html#getting-started-with-iotp>. Accessed: 2019-02-28.
- [11] Weiqing Huang. A Survey on Mobile Edge Computing. *Aalto University*, 2016.
- [12] Everyware Software Framework Developer’s Hub. Edge Computing Platform. <https://esf.eurotech.com/v5.0.0/docs/edge-computing-platform>. Accessed: 2019-02-20.
- [13] He Li, Kaoru Ota, and Mianxiong Dong. Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing. *IEEE Network*, 32(1):96–101, January 2018.
- [14] Daniel Torán Mercadé. Comparison of different Internet of Things platforms. Master’s thesis, Barcelona School of Industrial Engineering, January 2018.
- [15] Janakiram MSV. Azure IoT Edge: A Technology Primer. <https://thenewstack.io/azure-iot-edge-a-technology-primer/>. Accessed: 2019-02-21.
- [16] Amazon Web Services. Access Local Resources with Lambda Functions. <https://docs.aws.amazon.com/greengrass/latest/developerguide/access-local-resources.html>. Accessed: 2019-03-18.
- [17] Amazon Web Services. AWS IoT Greengrass. https://aws.amazon.com/greengrass/?nc1=h_ls. Accessed: 2019-03-16.
- [18] Amazon Web Services. AWS IoT Greengrass FAQs. https://aws.amazon.com/greengrass/faqs/?nc1=h_ls. Accessed: 2019-03-16.
- [19] Amazon Web Services. How to Configure Local Resource Access Using the AWS Command Line Interface. <https://docs.aws.amazon.com/greengrass/latest/developerguide/lra-cli.html>. Accessed: 2019-03-18.
- [20] Amazon Web Services. What Is AWS IoT Greengrass? https://docs.aws.amazon.com/en_us/greengrass/latest/developerguide/what-is-gg.html. Accessed: 2019-03-15.

- [21] Xiaofei Wang, Yiwen Han, Chenyang Wang, Qiyang Zhao, Xu Chen, and Min Chen. In-Edge AI: Intelligentizing Mobile Edge Computing, Caching and Communication by Federated Learning. *ArXiv*, September 2018.
- [22] Wei Yu, Fan Liang, Xiaofei He, William Grant Hatcher, Chao Lu, Jie Lin, and Xinyu Yang. A Survey on the Edge Computing for the Internet of Things. *IEEE Access*, 6:6900–6919, November 2017.
- [23] Ke Zhang, Supeng Leng, Yejun He, Sabita Maharjan, and Yan Zhang. Cooperative Content Caching in 5G Networks with Mobile Edge Computing. *IEEE Wireless Communications*, 25(3):80–87, June 2018.
- [24] Yuchen Zhou, Fei Richard Yu, Jian Chen, and Yonghong Kuo. Communications, Caching, and Computing for Next Generation HetNets. *IEEE Wireless Communications*, 25(4):104–111, August 2018.

Appendices

A Guide for registering a new IoT Edge device with Azure CLI [2]

You are able to register a device using Azure portal, Azure CLI or Visual Studio Code. In this short tutorial, we describe steps how to register a new Azure IoT Edge device with Azure CLI. Azure CLI is the open-source cross-platform command-line tool for managing Azure resources such as IoT Edge. All presented commands are executed in Azure CLI. Moreover, be aware that before you will follow these steps, you must already have IoT hub in Azure subscriptions, Azure CLI and IoT extension for Azure CLI.

1. Create a device

```
|| az iot hub device-identity create --device-id [device id] --hub-  
||   name [hub name] --edge-enabled
```

2. View all devices

```
|| az iot hub device-identity list --hub-name [hub name]
```

3. Retrieve the connection string

```
|| az iot hub device-identity show-connection-string --device-id [  
||   device id] --hub-name [hub name]
```

We need the connection string that links our physical device with its identity in the IoT hub when we are ready to set up our device.

B Guide for registering a new IoT Edge device using Watson IoT Platform dashboard [10]

We can add devices one at a time from the Watson IoT Platform dashboard, or we can use the Watson IoT Platform API External link icon to add one or more devices at a time. In this short tutorial, we describe steps on how to register a new IoT Edge device from the Watson IoT Platform dashboard.

1. In the IBM Cloud console, we should click Launch on the Watson IoT Platform service details page.

Open the Watson IoT Platform web console in a new browser tab at the following URL:

```
|| https://org_id.internetofthings.ibmcloud.com/dashboard/##  
|| overview
```

Here, *org_id* is the ID of our Watson IoT Platform organization.

2. In the Overview dashboard, from the menu pane, select Devices and then click Add Device.
3. Then, create a device type for the device that we are adding. Furthermore, we should enter a device name and optionally we enter device type attributes and metadata. After that, click *Next*.
4. Enter a device ID, for example *my_first_device*. Click *Next*.
5. Provide an authentication token, or accept an automatically generated token.
6. Finally, verify the summary information is correct, and then click *Add* to add the connection.

C Guide for creating local resource using AWS CLI

On Greengrass cores running Linux, these locally deployed Lambda functions can access local resources that are physically present on the Greengrass core device [16]. There are two types of resources: volume resources and device resources. Volume resources can be files or directories on the root file system. However, for device resources, we are allowed to use only character devices or block devices under */dev*. This short guide shows an example of creating TestCamera resource using the AWS Command Line Interface (CLI).

We use *CreateResourceDefinition* command to create a resource definition that specifies the resource to be accessed [19]:

```
aws greengrass create-resource-definition --cli-input-json '{
  "Name": "MyLocalVolumeResource",
  "InitialVersion": {
    "Resources": [
      {
        "Id": "data-device",
        "Name": "TestCamera",
        "ResourceDataContainer": {
          "LocalDeviceResourceData": {
            "SourcePath": "/dev/video0",
            "GroupOwnerSetting": {
              "AutoAddGroupOwner": true,
              "GroupOwner": ""
            }
          }
        }
      }
    ]
  }
}'
```

The option *-cli-input-json* indicates that we will provide resource definition in JSON format. In this example we specified */dev/video0* as the source of *TestCamera* resource.

Survey on Software Defined Radios in IoT

Javier Benítez Fernández

javier.benitezfernandez@aalto.fi

Tutor: Verónica Toro-Betancur

Abstract

Low-Power Wide Area Networks (LPWANs) have been present on a wide variety of IoT deployment on the last decade. On the other hand, Software Defined Radios (SDRs) have been present in workstations and servers for a long time but they are rarely found in IoT devices or IoT gateways. In this paper, we aim to explore the state-of-the-art of SDRs and the requirements of the LPWANs to analyze the viability of IoT deployments in multiple scenarios using SDRs.

KEYWORDS: *IoT, LWPAN, SDR, LoRa, NB-IoT, SigFox, M2M*

1 Introduction

The IoT ecosystem has been increasing in deployments and volume of devices in the last years. The IoT devices have been utilized for different purposes during this time. Several setup configurations can be found for different purposes such as smart cities [3] or dam sensors [14].

This paper presents some approaches to IoT networks. One of them is Low-Power Wide Area Networks (LPWANs) based deployments. LPWANs have been used for massive deployments such as Smart City applications

[3]. In this approach, instead of having every device connected to a 3G/4G Base Station (BS), a set of devices communicate with an Access Point (AP) or gateway using a short-range technology such as LORA, Sigfox or NB-IoT [3]. To achieve this purpose, as on the traditional protocols, every device is required to have an embedded antenna.

There are different approaches to address the problem that appears executing the protocol. A traditional approach is to handle the protocol communication in the application-specific integrated circuit (ASIC) or peripheral antennas. This means that the protocol is static and, it is not possible to modify it [13]. Every upgrade should be done by replacing the device. This issue could be relevant when the goal is to deploy a massive network of IoT devices.

In contrast, Software Defined Radios (SDRs) have been implemented on different platforms as FPGAs and PCs [9]. SDRs enable the possibility to extract the protocol implementation from the ASIC and manage it directly in the CPU. It allows updating the protocol to prevent major bugs and even upgrade to a newer version without replacing the deployed devices [13].

However, SDRs involve a number of drawbacks as well. SDRs approach present different advantages and disadvantages for a variety of scenarios regarding the ASIC approach. The aim of this paper is to analyze the viability and possible SDRs use cases in IoT environments, particularly LPWANs.

This paper is organized as follows. Section 2 presents LPWANs related technologies, and Section 3 presents SDRs methodologies. Section 4 describes LWPAN deployment use cases where SDR have a better performance. Finally, Section 5 offers some concluding points related to the state-of-art of IoT deployments using SDR.

2 Low-Power Wide Area Networks (LPWAN)

On the rise of multiple IoT deployments in country-side, it became evident two main drawbacks of the traditional technologies as LTE [10] or IEEE 802.16m [1]. First, the current networks are not prepared to support the

amount of M2M devices that could be deployed [11]. However, deployments in small cities or country-side usually have out-of-range areas [14]. Since traditional cellular protocols such as 3G or 4G have limited operational areas, mobile and country-side deployments require a more flexible and long-range technology that allow having a continuous connection. Second, IoT devices are usually energy-constrained, which means that they have limited energy resources.

There are two types of traditional technologies that have been used in country-side deployments: short-range and cellular communications. Short-range technologies, such as Bluetooth and ZigBee, are not adaptable to long-range deployments as a consequence of the usual energy-constraint. Cellular communication technologies offer high throughput, solutions using these technologies could provide wider networks, however, they also consume an unreasonable amount of energy. Hence, short-range communications and cellular communications are hardly suitable for resource-constraint IoT devices.

For IoT systems to be readily adapted, they must meet some common requirements such as cost-effectiveness, long range coverage and low energy consumption. As a result, these requirements force the systems to transmit at a low data rate. For this reason, LPWAN technologies have become popular in the industry and the research community, because they provide low energy consumption, long range and low-cost communication features [11]. They provide a coverage range of 10-40 km in rural areas and up to 1-10 km in urban areas depending on the technology and other factors such as radio interferences [3].

Low-Power Wide Area (LPWA) is a rough classification of very different technologies that allow long-range communications at a low cost [16]. The technologies vary on characteristics, requirements and use cases. For the following sections, we analyse the differences between the main technologies: LoRa, SigFox and NB-IoT [11].

2.1 *LoRa*

LoRa is a proprietary wireless PHY protocol designed for long-range wireless communication. It was originally designed by Semtech Corporation. It was designed to operate over unlicensed ISM bands, 169 MHz, 433 MHz

and 915 MHz in USA and 868 MHz in Europe. The communication is provided by *Chirp Spread Spectrum* (CSS) that spreads a narrow-band signal over a wider bandwidth channel. The technology supports data rate on a range between 300 bps and 50 kbps [11].

LoRaWAN, the communication protocol built on the top of LoRa, is developed by LoRa Alliance under the open-source paradigm. It extends the LoRa PHY layer to provide a communication protocol and a system architecture. As an architecture example, LoRaWAN nodes are not associated with an specific gateway, instead, the packages will be received by multiple gateways. These packages will be sent to a network server (typically cloud based) via a traditional protocol such as Ethernet, WiFi or LTE [16]. Using this redundancy, it is possible to achieve a higher received packages ratio by compromising network deployment costs due to an increased traffic [11].

Recently, Centenaro et al. described in their research a recent deployment for urban applications using LoRa. The described IoT deployment relay on 30 gateways deployed on an area of 100km² in Padua in Italy with a capability of 1500 nodes per gateway [18]. Another deployment test, as a proof of concept, had the objective of monitoring and control the temperature and humidity of the different rooms on a building of 19 floors. This was composed of 32 nodes around the building and a single gateway [3].

2.2 NB-IoT

NB-IoT is a narrow band IoT PHY protocol developed by 3GPP [19]. It operates along cellular protocols (GSM and LTE), relaying on 3G, 4G and 5G networks, under licensed frequency bands, 700 MHz, 800 MHz and 900 MHz. It does not allow full compatibility but it allows to make use of traditional cellular networks without compromising the performance of the GSM or LTE [19]. LTE design has been reused to make it easier to implement by the vendors that already offer LTE solutions [11].

In contrast with LTE and GSM, NB-IoT aims to get efficient battery cost, deployment flexibility and large scalability, up to 100 thousand end devices per AP, using a small part of the existing spectrum. NB-IoT supports a data rate of 200 kbps for the downlink and 20 kbps for the uplink with

a maximum payload length of 1600 bytes [19].

A simulation of a NB-IoT deployment was performed in [2]. An area of 800km² was covered based on the local communications company network supporting up to 25,000 devices. NB-IoT also provides service to deep indoor nodes [18].

2.3 SigFox

SigFox is a PHY protocol that uses Ultra Narrow Band radio technology based on D-BPSK and GFSK modulations. This technology operates in unlicensed bands, 868 MHz in Europe, 915 Mhz in US and 433 MHz in Asia [15]. It is a patented technology that was developed by the French company that shares the name.

SigFox designed a proprietary communication layer on top of the PHY layer. The basic deployment consists of a gateway connected to the internet to access a cloud service that handles the transmitted data and the end devices in the gateway coverage area. This protocol is meant to be used in networks with low traffic since uplink communication is limited to 140 messages per day with a payload of 12 bytes [11].

2.4 Comparison

The previously introduced technologies cover a diversity of use cases. When choosing a convenient LPWAN technology for a specific use case, the following factors are relevant. The differences between these technologies are shown in Fig. 1.

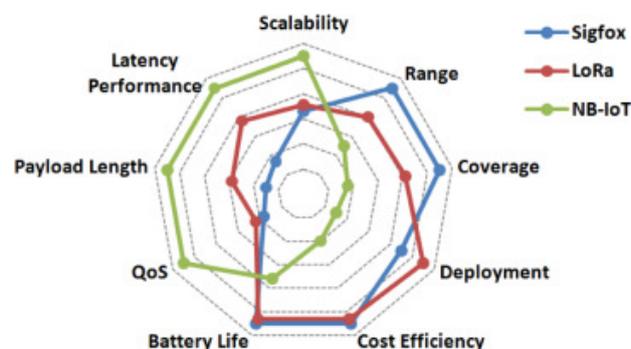


Figure 1. LPWAN comparison of advantages and disadvantages [11]

Energy Cost

The main difference between SigFox, LoRa and NB-IoT is the communication model. NB-IoT uses asynchronous communication and other features such as OFDM/FDMA access modes and QoS handling that allow to have a stable and highly reliable communication [12]. However, it increases radically the energy consumption as compared to SigFox and LoRa.

Scalability and Coverage range

Another difference between SigFox, LoRa, and NB-IoT is the necessity to deploy their own gateways in contrast to use existing infrastructure to connect end devices. These gateways need to have internet access and be on the range of the devices. This approach could be costly on some scenarios such as long-distance routes deployments or large deployments on countryside with a weak signal for cellular networks. However, SigFox and LoRa provide flexibility on location, it is possible to have a network in the countryside or other places that are not covered by LTE or other traditional cellular networks.

There is another difference that should be taken into account: NB-IoT covers a shorter range (< 10 km) in comparison of Sigfox (< 40km) and LoRa (< 20km) [3]. The two latter are able to cover larger areas with less APs, making them more suitable for rural areas.

Quality of Service

NB-IoT uses licensed frequency bands that are more stable than unlicensed bands as the ones used by SigFox and LoRa. Unlicensed bands can lead to interferences and weaker signals that largely deteriorate the quality of the communication [11]. SigFox and LoRa rebound from interferences, fading and multipath. However, SigFox and LoRa are not able to compete with the QoS of NB-IoT because of synchronous communication and the LTE legacy. For this reason, NB-IoT offers an high QoS. [19].

3 Software Defined Radios (SDRs)

As it was explain in previous sections, the current IoT systems have some constraints that limit the design and architecture of deployments. A traditional challenge in computer architecture design is the method used to process the communication protocol in peripheral antennas. A traditional

approach is to handle the communication protocol execution in the ASIC. This solution provides a low energy consumption compromising the quality of the communication. On the other hand, SDRs method handles the communication protocol execution in the CPU. This method provides a smarter use of the antenna, allowing dynamic capabilities such as multiple standards or on-the-fly updates [13].

SDRs came through two generations, FPGAs and PCs. SDR evolution has been slow since J. Mitola first defined it in 1991 [6]. The main reason of the delay of this process was a lack of real-time signal processing. However, the current computational power allows real-time signal processing but this untied another constraint, energy consumption.

SDRs presents multiple advantages and disadvantages related to the early stage of SDRs in IoT deployments. This limit the usability of the implementations in current deployments. However, there are some deployments using SDRs in some of the components. These topics are discussed on next sections.

3.1 Advantages and disadvantages

On the other hand, SDRs handle the communication protocol execution directly in the CPU. SDR, as a new technology that it is trying to replace traditional ASIC methodology, has some benefits but it also has some drawbacks.

Advantages

- **Multiple Standards.** Currently, there are hundreds of communication standards. Traditional ASIC methods require to have a peripheral antenna for each protocol, while SDR allows to reuse one antenna for multiple protocols by programming the processor to dynamically switch between protocols [4].
- **Supports on-the-fly updates.** It allows to change the implementation of the communication protocols to allow adaptability to a specific IoT system. This could be easily achieve by downloading a software/firmware [4].
- **Shared Modules.** As the implementation of different protocols share

common processing, it is possible to improve computational efficiency by centralizing the execution of these modules [4].

Disadvantages

- **Energy cost.** Traditional SDRs present a high energy consumption that would not be suitable for most IoT devices. It mainly affects in two stages: initialization and operation. In the *initialization* stage, the device needs to configure and initialize the antenna. In the *operation* stage, the device will present a consumption of energy on idle mode [7]. However, over the last years, some experiments showed how it is possible to reduce the energy consumption and in some cases even make it lower than a traditional ASIC [4].
- **Added latency.** The performance of the protocols will depend on the processor architecture and how the implementation adapts to it. In some situations, it could be found a considerable improvement in energy cost and performance [13].

3.2 SDR in IoT deployments

IoT deployments using SDR stay on an experimentation stage. Limited information on deployments using these methodologies can be found. However, multiple experiments have been performed that expose the constraints and advantages of using SDR in IoT deployments.

Use cases

- Hson-Wei Cho et al. proposed in [5] a 5G IoT deployment using SDR on the gateways that reduce the error ratio of the transmissions with a high configurability allowing adaptability of the gateway depending on location constraints and other requirements.
- Manolis Surligas et al. introduce in [17], a platform that allows a gateway to communicate with different protocols using SDR. Their current implementation allows handling 6LoPWAN and IEEE 802.11 PHY layers using an SDR driver. On their tests, the driver requires between 29% to 36% of the total CPU power while being idle. The results are promising for the stability and flexibility achieved. Nevertheless, the

current implementation is roughly suitable for a production deployment due to the high computational costs.

- At present, Yajing Chen et al. (2016) shown on their research, it is possible to keep the overhead of the power consumption on a minor 8%. This reduces the total power consumption permitting to a wider reach of IoT devices.

4 Discussion

SDR has some benefits over traditional methods but it also has some disadvantages. On the other hand, there are multiple LPWAN technologies that are suitable for different uses cases. In this section, various LPWAN scenarios are discussed where SDR could be implemented.

4.1 IoT deployment with multiple communication technologies

Industrial IoT (IIoT) is an old industry. First deployments are dated on the late 1960s [8]. Originally, most of the applications were focused in building automation and sensor analysis. One of the major challenges that it confront as an Industry is the modernization of the infrastructure, keeping a centralized management.

These kind of scenarios require a flexibility on the gateways, allowing then to communicate using different protocols such as WiFi, LoRa, BLE or Zigbee. SDR permits to dynamically switch between protocols and it reduces the energy overhead that a normal ASIC architecture produces.

4.2 Real-time scenarios

On a scenario where real-time is required and there are no energy consumption limitations, low latency and QoS are the main goals to achieve. One application is real-time machinery monitoring for a manufacturing automation to more use efficiency on the production line.

As this kind of applications require high quality service with a low latency, SigFox and Lora are not suitable, meanwhile, NB-IoT provides high QoS and low latency. On the other hand, using SDR on the devices could

be beneficial for increasing the performance of the communication. SDR allows to dynamically change the configuration to adapt to the situation and optimize the configuration for it. Regardless the previous benefits, SDRs are not suitable when energy consumption is a concern on the design of the system.

5 Conclusion

This paper has summarized the main LPWAN technologies and the SDR approaches, discussed their advantages and disadvantages and concluded on the possible use cases where they could be used together.

On the previous sections, there were exposed different scenarios where LPWANs have been used, showing the advantages and the limitations of these technologies. NB-IoT allows lower latency with a higher data rate than LoRa and SigFox meaning that it is suitable for real-time scenarios such as electric metering [11]. Nevertheless, NB-IoT, in contrast to LoRa and SigFox, has a large energy consumption. LoRa and SigFox are the recommendable options when the scenario has an energy constraint such as smart farming. Also, LoRa and SigFox are the recommendable option for long distance deployments on applications such as tracking for logistic. However, the deployment of an BS for NB-IoT communications has a considerably higher cost than for LoRa or SigFox [11].

On the other hand, there were exposed different scenarios where SDRs have been successfully used on IoT deployments as consequence of the early stage of the technologies in an IoT context. The main limitation is the high computational and energy cost of the implementations. However, the SDR in IoT use cases exposes a high settings flexibility and an increase in the quality of the radio signal handling.

The technologies and methodologies that are used for SDRs are slowly evolving to permit low consumption and expose the other advantages of SDRs to the wide variety of devices and nodes that are used for IoT deployments.

References

- [1] IEEE Standard for WirelessMAN-Advanced Air Interface for Broadband Wireless Access Systems –Amendment 2: Higher Reliability Networks. *IEEE Std 802.16.1a*, 2013.
- [2] A. Adhikary, X. Lin, and Y. . E. Wang. Performance evaluation of nb-iot coverage. In *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, pages 1–5, Sep. 2016.
- [3] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi. Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios. *IEEE Wireless Communications*, 23(5):60–67, October 2016.
- [4] Y. Chen, S. Lu, H. Kim, D. Blaauw, R. G. Dreslinski, and T. Mudge. A low power software-defined-radio baseband processor for the Internet of Things. In *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 40–51, March 2016.
- [5] H. Cho and H. Wei. A Flexible IoT RAN System Based on SDR with Optimal Antenna Distribution. In *2017 IEEE Globecom Workshops (GC Wkshps)*, pages 1–5, Dec 2017.
- [6] Markus Dillinger, Kambiz Madani, and Nancy Alonistioti. *Software defined radio: Architectures, systems and functions*. John Wiley & Sons, 2005.
- [7] Prabal Dutta, Ye-Sheng Kuo, Akos Ledeczki, Thomas Schmid, and Peter Volgyesi. Putting the software radio on a low-calorie diet. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Hotnets-IX*, pages 20:1–20:6, New York, NY, USA, 2010. ACM.
- [8] Alasdair Gilchrist. *Industry 4.0: the industrial internet of things*. Apress, 2016.
- [9] Stefan Knauth. Implementation of an IEEE 802.15. 4 transceiver with a software-defined radio setup. *Proceedings of the embedded world 2008 Conference, Nuremberg, Germany, 02 2008*.
- [10] The LTE standard. Standard, Signal Research Group, Ericsson and Qualcomm, Minnesota, MN, 2014.
- [11] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express*, 2018.
- [12] Sung-Min Oh and JaeSheung Shin. An efficient small data transmission scheme in the 3GPP NB-IoT system. *IEEE Communications Letters*, 21(3):660–663, 2017.
- [13] Yongtae Park, Jiung Yu, JeongGil Ko, and Hyogon Kim. Software radio on smartphones: Feasible? In *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications, HotMobile '14*, pages 17:1–17:6, New York, NY, USA, 2014. ACM.
- [14] M. Saravanan, A. Das, and V. Iyer. Smart water grid management using LPWAN IoT technology. In *2017 Global Internet of Things Summit (GIoTS)*, pages 1–6, June 2017.

- [15] SigFox. SigFox radio technology. <https://www.sigfox.com/en/sigfox-iot-radio-technology>, 03 2019.
- [16] Rashmi Sharan Sinha, Yiqiao Wei, and Seung-Hoon Hwang. A survey on LPWA technology: LoRa and NB-IoT. *ICT Express*, 3(1):14 – 21, 2017.
- [17] Manolis Surligas, Antonis Makrogiannakis, and Stefanos Papadakis. Empowering the IoT heterogeneous wireless networking with software defined radio. In *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE, 2015.
- [18] Veronica Toro Betancur and Mario Di Francesco. Management of Resource-Constrained IoT Devices in Urban Scenarios. pages 55–69, 2018.
- [19] Y.-P. Eric Wang, Xingqin Lin, Ansuman Adhikary, Asbjörn Grövlén, Yutao Sui, Yufei W. Blankenship, Johan Bergman, and Hazhir Shokri-Razaghi. A Primer on 3GPP Narrowband Internet of Things (NB-IoT). *CoRR*, abs/1606.04171, 2016.

QKD in Small IoT Devices

Tuomas Ebeling

tuomas.ebeling@aalto.fi

Tutor: Jukka Nurminen

Abstract

Quantum key distribution (QKD) offers a way to generate a symmetric encryption key for securing communications. QKD is one of the leading ways for future-proofing security as the computational power of an attacker is expected to dramatically grow with the development of quantum computers. Nodes for the internet of things are required to be cheap, energy-efficient and small. This paper studies quantum key generation and its security aspects and the possibility for adapting the technology to small IoT devices.

KEYWORDS: QKD, quantum key distribution, Internet of Things, Quantum Channel, Cryptography

1 Introduction

Current practice in securing communications for commercial use is heavily focused on public key cryptography. With the evident rise of quantum computers, the infeasibility of factoring large integers might not provide sufficient protection in the long run [1]. A truly secure symmetrical key, which was described by Shannon already in 1949 [2], has suffered from the problem of key delivery to the communicating parties.

Quantum key distribution (QKD) is a promising way to deliver this symmetrical key, often referred to as a "one-time-pad". Previous studies [3, 4] show that the generation of this can be made in real time for the communicating parties, with removing the possibility of eavesdropping. This key can then be used in a regular radio channel for secure communications.

This paper focuses on the application possibilities of using QKD in small devices, namely nodes for the Internet of Things (IoT). The paper studies requirements that IoT presents for QKD transmissions and reviews possible solutions for these requirements.

The paper is structured as follows: Section 2 studies the differences between symmetric key and public key cryptography; Section 3 goes through basic functionality of quantum key distribution; Section 4 discusses possibilities to use QKD in IoT devices and Section 5 concludes the paper.

2 Cryptography

Cryptography is a practice in which the goal is to hide communication between two parties from an eavesdropper. In this paper, cryptography is used as means to protect the communication over a public channel. In a public channel, anyone is assumed to be able to access it, so the information sent over the channel must be useless for all but the communicating parties.

2.1 Symmetric Cryptography

Symmetric cryptography is encrypting a message M with a randomly generated key K to form a ciphertext C that is sent over a channel. As the name suggests, the same key K is used for message retrieval through decrypting the ciphertext, demonstrated in figure 1. The security of a symmetric cryptography system is perfect, since if an eavesdropper receives the cipher text, the best possible solution for decrypting the message without key K would be to guess the message [2]. This is explained in more detail in section 2.3. While the system provides secrecy, it suffers from a more practical point of view; how to ensure that the communicating parties share the same key while keeping it a secret.

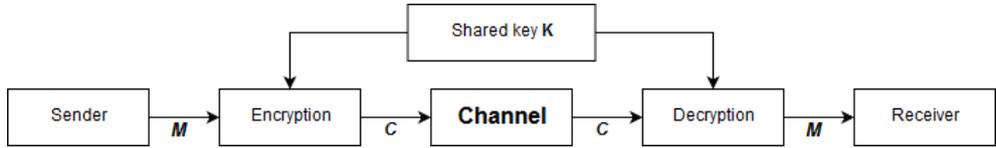


Figure 1. Symmetric Cryptography

2.2 Public Key Cryptography

Public key cryptography, on the other hand, uses a different key for encryption and decryption. This method is widely used in modern communications, since it allows to freely distribute the public part of the key. Figure 2 demonstrates public key encryption. The message M is encrypted by a public key from the receiving end and can only be decrypted with the corresponding private key, safely stored by the receiver. The drawback of public key cryptography is that the system, theoretically, can be broken mathematically. As in all security aspects, it is important to assess vulnerabilities for future developments. The development of quantum computers might offer ways to break the security of public key cryptography systems.

One commonly used way to generate a public key is the RSA-algorithm [5]. RSA is based on the integer factorization problem (IFP), which states that if two large primes are multiplied together, it is so hard to factor the resulting integer back to the original primes that the solving would be infeasible. In RSA, the sender generates a public key-pair (e, n) for encryption, and a private key-pair (d, n) for decryption, where n is the multiplicative of two large primes:

$$n = p \cdot q \quad (1)$$

The source that generated the key is the only one to know p and q . They are also used to compute the decryption key d , by:

$$\gcd(d, (p-1) \cdot (q-1)) \quad (2)$$

Finally, e is calculated as the "multiplicative inverse" of d , such that:

$$e \cdot d \equiv 1 \pmod{(p-1) \cdot (q-1)} \quad (3)$$

When formulas 2 and 3 are satisfied, a message M can be enciphered to ciphertext C and deciphered back to M as:

$$C \equiv M^e \pmod{n} \quad (4)$$

$$M \equiv C^d \pmod{n} \quad (5)$$

The RSA-algorithm is as strong as the secrecy of the original prime numbers p and q . If they were to be compromised to the attacker, all communications can be easily decrypted and even maliciously modified. The proof of the mathematics in the algorithm can be found in the original paper of Rivest, Adleman and Shamir [5].

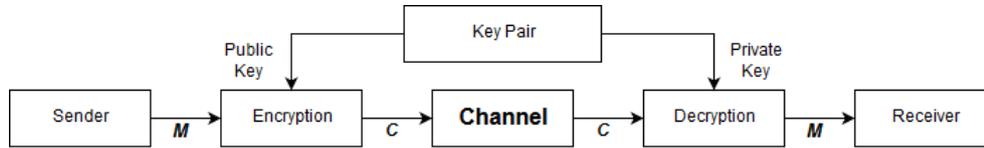


Figure 2. Public Key Cryptography

2.3 Security

The security in public key cryptography is based on DLP and IFP problems, which state that the solving of the private key with current algorithms is infeasibly costly. With algorithms that are run on a modern computer, the time complexity of solving a DLP or an IFP is $\mathcal{O}(2^{n^k})$, meaning sub-exponential growth in time as the input grows. Quantum algorithms, on the other hand, would decrease the time complexity to a polynomial time $\mathcal{O}(n^k)$, making the solutions feasible for attackers to compute. The security of these algorithms is solely based on them enduring any attempted attacks of current algorithms with current hardware. It is however not proven that the public-key algorithms are secure in a way that a vulnerability cannot be discovered, provided enough computational power [6].

For the symmetric key to be truly secure, it must satisfy the following requirements:

- Every key K is used exactly once.
- The selection on K is completely random.
- M and C are independent from each other, meaning K must satisfy the previous requirement and be at least as long as M .
- The key K is delivered to the communicating parties securely.

With the first three requirements satisfied, the shared, or mutual information of M and C is 0:

$$I(M; C) = H(M) - H(M|C) = 0 \quad (6)$$

This means that the best way for an attacker to deduce M from C is to take a guess, yielding perfect security for the system [2]. The last requirement, however, is more difficult to satisfy, as physically transferring the key is not practical in modern communications. Therefore, QKD is considered to transfer key bits over the network securely.

This paper is concerned in finding ways to use quantum key distribution for symmetric key generation in small devices. QKD provides means to transfer a key from a sender to receiver securely, but also suffers from limitations. These are further discussed in section 3.

2.4 Cryptography in IoT Devices

For IoT devices, besides security, there are other aspects to address as well. In order to get the maximum potential from an IoT network, it has to be designed for low-cost and efficiency. The focus for designing new IoT protocols is in providing good energy efficiency and keeping the required memory size and computational complexity in the IoT node at a minimum, thus reducing cost and energy drain. The devices are often battery powered, resulting in a need for low energy drain. The other goal of future IoT is to have a very high amount of nodes connected into the network, hence the low cost [7].

The usual attacks on IoT devices are denial of service attacks, man in the middle attacks and eavesdropping. The attacks can be further divided into three subcategories: physical layer security, application level security and network layer security. The paper will focus on the communications side of security e.g. network layer, as it is provided with cryptographic measures. The physical security is hard to achieve, as the device itself should not be in the presence of any attacker. Application layer security is very important in IoT devices, but also not addressed in this paper [8].

The nodes in the IoT network generate lots of data and are to send it to the server. For secure connectivity, the device must be authenticated before the actual communications, and then the resulting communications traffic must be encrypted. The authentication itself requires heavy computation from the node [7], especially if node-to-node communications are enabled. All computations reduce the battery life the node, thus complexity and excess traffic should be avoided while establishing secure connectivity [8].

Below, some of the common network layer attacks have been explained [9]. They are explained for evaluation of the QKD-protocol in IoT in sec-

tion 4.

Traffic Analysis Attack

The attacker obtains information about the network by accessing it. All kinds of information, such as open ports or traffic type can be used in further attacks.

Man in The Middle

An attacker in the middle of the channel intercepts the sent data and eavesdrops or even modifies it. This is a big concern in IoT nodes, where sensitive information can be leaked or a node can be maliciously controlled.

Denial of Service

The network is flooded to deny any real information to pass. This makes the system inoperable and useless for the time of the attack.

3 Quantum Key Distribution

As concluded in subsection 2.3, public key cryptography based communications are vulnerable for eavesdropping, provided that the computational power of the attacker is sufficient to make such attack feasible. A one-time-pad, the only once used shared symmetric key between the communicating parties, provides the level of secrecy that satisfies any future developments. This section addresses the key delivery problem, and offers QKD as a way to transfer key bits over a network.

Quantum key distribution is a way to generate a symmetric key between communicating parties without providing any means for a possible eavesdropper to gain information about the conversation. In a general scenario, the sender is referred to as Alice, the receiver as Bob and the eavesdropper as Eve. These names will be used throughout the rest of this paper.

The process of generating this key is dependent of having a quantum channel, which is used to transfer quantum bits, qubits. The characteristics of qubits provide that any information sent through the quantum channel cannot be copied, thus not replicated nor monitored in any way by an attacker without instant detection. This qubit's information lies in its state, and that state changes when measured. After each measurement, the original state of the qubit cannot be restored since the measurer only knows of the current, changed state [6].

BB84

The quantum channel can be used to transmit polarized light. The light photons can be polarized to an angle and then filtered at the receiving end. If a photon is transmitted with an angle α , and received through a filter with an orientation of β , the light passes through the filter or is absorbed with probabilities of:

$$P_{transmitted} = \cos^2(\alpha - \beta) \quad (7)$$

$$P_{absorbed} = \sin^2(\alpha - \beta) \quad (8)$$

As discussed earlier, the successful reception of a photon makes it change its polarization to the measurement angle β . From equation 7 can be seen that if the original polarization angle is the same as the measurement angle, the photon goes through the filter successfully. On the other hand, if the angles α and β are orthogonal with each other, the photon is guaranteed to be absorbed. A polarization angle α that is exactly 45° off the measurement angle passes through with $P = 0.5$ [3].

The characteristics of photons allow Alice to send bits encoded in the angle of the photon's polarization. Random polarization bases are chosen for the transmission and reception, and only those bits of which Alice and Bob have chosen the same angle can be used for key generation. After key transmission, the measurement bases used by Alice and Bob are exchanged over a public channel. This provides no useful information to an attacker, since the bits measured are still secret.

The previous example of polarized photons is utilized in the BB84 protocol, developed by Bennet and Brassard [3], which is the basis for a wide variety of quantum cryptography nowadays. The security of the protocol lies in the probability of successful key recovery after filtering. On average, 50% of the transferred key bits are received correctly over an undisturbed channel. If Eve were to listen to the channel, this number would instantly drop, since Eve's measurements would change the states of the sent photons. This way the presence of an eavesdropper can always be detected.

BB84 is a two-dimensional QKD-protocol, meaning it uses only two measurement bases. It is subject to low key rates and quite short distances [10], but a valid way to securely generate a one-time-pad between Alice and Bob.

3.1 Range and Key Rates

The rate that a key is generated between Alice and Bob reduces in almost linear fashion as distance increases, but when approaching the maximum distance, the key rate drops rapidly. The maximum distance is a result of the quantum bit error rate becoming high enough so that the secure key cannot be extracted from the sent data. Successful quantum key distribution needs the noise to be below a threshold for key extraction. This distance can be grown with adaptive coding methods and the use of different channel models, but still sets a limit for a single fiber length [11]. The maximum distances vary between adaptations, but range typically between 50km to 200km.

There are many steps to be taken before the secure key can be extracted from the sent data. First, most of the sent photons fade in the channel, resulting in Bob receiving only a fraction of them. Once sifting, comparison and privacy amplification is done, the resulting secret key is very small compared to the originally sent raw key [11]. This is why QKD networks are a necessity for wide QKD use.

3.2 QKD Networks

To form a QKD network, there must be a way to replicate the key through the nodes in the network. This is problematic, since the quantum characteristics of the signal limit possibilities, as the signal states cannot be cloned. One possibility is to have the signal stay at its quantum state throughout the travel, using quantum repeaters. This would mean that the network still acts like a straight link between Alice and Bob, and the repeater nodes in the network do not have to be trusted. The technology is still quite theoretical though, since the quantum operators required for quantum repeaters are not yet achievable. It is still a valid possibility in the future [12].

A more practical approach, which has seen experimentation [13], relies on trusted nodes. These nodes are used to decode the key bits of the message, encode them again and send them forward in the network. The process is replicated in every trusted node that the message passes. The solution produces an information theoretically secure system, provided the integrity of all trusted nodes. Although theoretically secure, the event of even one node being compromised renders the whole system unusable. The prototype network created in Vienna by SECOQC [13], proved that

this kind of network is possible for large-scale use. The obtained key rates in the experiment were not enough for practical use, with only one gigabyte of secure communications a month, but the development of new machinery and encoding methods have and will greatly grow this value.

A possible adaptation of the trusted node QKD network could be the use of satellites. A study by Meyer-Scott et. al. from 2011 [14] proves the possibility of such channel, where a satellite is used as an uplink, quantum data sent to the space. The paper concludes that a downlink sending a key back to earth is more complex to accomplish, but it offers an alternative possibility for future research to provide a trusted node for the QKD network. QKD's implementation to current SDN controlled fiber-optical networks have also seen study and experimentation [15, 16], so there might be various ways to produce a secure QKD network in the near future.

4 Quantum Key Distribution in IoT

Adapting quantum key distribution to IoT devices faces some practical issues. First and foremost, the IoT devices are usually small of size, meaning that large equipment readily available for QKD are not suitable for an IoT node. Second, the IoT nodes are mostly wireless, so a fiber connecting the communicating parties to a quantum channel is impractical at best. Other problems facing QKD adaptation is energy efficiency, provided that the node is battery powered and range, when the node is at a distant site.

The range issue was discussed in subsection 3.2, with trusted nodes acting as repeaters for the signal. The following subsections give some possible solutions for the remaining problems.

4.1 Chip Based QKD

Recent developments in microelectronics have paved way for integrating photonics to system chips. The chip-based quantum key distribution allows to make the receiving hardware very small, enabling integration to IoT devices. In a 2016 article by P. Sibson et. al. [17], methods for chip utilization were tested with different protocols. The results were optimistic about the possibility to use chip-based QKD for device miniaturization and flexibility, allowing key rates of about 345 kbps with the BB84 protocol [17]. The proposed solutions with further developments might enable

compact QKD devices that are of low cost, suitable for use in the internet of things [18].

4.2 Wireless QKD

Most experimentations with QKD have been carried out with a fiber link between Alice and Bob. Wireless QKD has emerged as a possible solution to serve indoor usage [19], at least in some specific scenarios, with the use of optical wireless communications.

A wireless quantum channel between devices faces problems that weaken the reliability and efficiency of key generation. The largest issue is the amount of noise that light based communications, for example Light Fidelity (Li-Fi), suffer from. Light that the sun or any other light source produce presents noise to the channel, which is problematic for QKD since it requires noise levels to be below a threshold explained in section 3.1. The wireless channel must then be formed as a beam between the devices, by angling them to point at each other with a low field of view. This limits the excess light in the channel [19].

The method does not provide mobility, as the channel is disturbed when the angle changes. However, it might be used in scenarios where the receiving device requires a wireless channel but is stationary, making it a possibility for IoT applications.

4.3 Coexistence of quantum and classical channels

As for security aspects, a full point-to-point quantum channel would be ideal for key transmission. QKD can be, however, used to provide some additional security over a fibre connection, and then transferred to an IoT node separately via a classical channel. This also could reduce battery usage in the IoT device, as the device would not need to generate keys locally.

The transmission method from the QKD receiver to the node could be of many different types. One way would be to generate the key bits to the receiver, and then manually fetch them with physical access. This is not the most practical way to distribute keys for multiple small devices, but a secure one. Another way could be the use of Li-Fi connection from the QKD receiver to the node. Optical wireless transmissions provide some level security for eavesdropping, as light does not penetrate walls. This could be used together with current public key cryptographic means.

Mavromatis et. al. [18] experimented with the possibility of a HTTPS connection to the node, removing the need for local key generation. The experiment used a fiber network for quantum key distribution. The results provided better battery performance with QKD+HTTPS compared to locally generated keys.

4.4 Overview of the proposed solutions

For quantum key generation to be usable in the internet of things, networking, hardware and transmission techniques must be further developed. Network access layer security issues in IoT were discussed in section 2.4. The requirements for IoT cryptography were to provide defense against the listed types of attacks, plus fit the use cases of IoT systems.

The use of QKD would solve any attempt to eavesdrop the channel, as the presence of Eve can always be noticed, guaranteed by the laws of physics. This also mitigates man in the middle attacks. Traffic analysis attacks might still be possible, since the actual conversation is carried out over a classical channel, but as the communication is perfectly secure the list of vulnerabilities can be mitigated. A quantum channel still is vulnerable to denial of service, as the quantum channel can easily be blocked. The presence of Eve can also be used to deny the flow of information, since an attempt to eavesdrop renders the qubits unusable.

The implementation of QKD to IoT devices have clear benefits for both secrecy and battery consumption [18]. Chip-based quantum cryptography is the main enabler of small QKD capable systems. The miniaturization, low cost and energy efficiency of future solutions enable the hardware to be used in a wide range of IoT devices [17].

As the methods for wireless QKD channels develop, a complete quantum connection between communicating devices can be formed. This would be ideal for secrecy throughout the system. Before these developments, the coexistence of both quantum and classical channels can be considered as a viable alternative for IoT connectivity. Optical wireless communications seem promising for providing a theoretically working quantum channel for key exchange, but the requirements for such perfect channel conditions can be difficult to accomplish in practice.

It is possible to create IoT networks secured by a quantum key distributed symmetric key, but the technology must be developed further for any practical or even commercial use.

5 Conclusion

This paper focused on the security provided by QKD and its integration possibilities for the internet of things. Quantum key distribution is a secure way to fend off attackers of unlimited computational power, but the technology has to develop to be fully usable in small IoT nodes.

QKD networks offer a way to extend the coverage area beyond a single link. A large-scale network will also offer protection against link failure with routing possibilities.

The use of QKD in IoT requires the receiving hardware to be small, cheap and energy efficient, while offering flexibility in the placement of nodes. Proposed solutions include both wireless QKD and the use of classical and quantum channels as a hybrid system. The latter provides a more reliable solution for the near future, but as time passes, point-to-point QKD would be more desirable.

QKD is a promising way to generate secure keys, and its use in IoT is possible, although many developments must be made for commercial use.

References

- [1] P. W. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM J. Comput.*, vol. 26, pp. 1484–1509, Oct. 1997.
- [2] C. E. Shannon, "Communication Theory of Secrecy Systems," *The Bell System Technical Journal*, vol. 28, pp. 656–715, Oct. 1949.
- [3] C. H. Bennett and G. Brassard, "Quantum Cryptography: Public Key Distribution and Coin Tossing," *Theoretical Computer Science*, vol. 560, pp. 7–11, 2014.
- [4] S. Wiesner, "Conjugate Coding," *SIGACT News*, vol. 15, pp. 78–88, Jan. 1983.
- [5] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, pp. 120–126, Feb. 1978.
- [6] C. P. Williams, *Explorations in Quantum Computing*. Springer, 2 ed., 2011.
- [7] U. Banerjee, C. Juvekar, A. Wright, and A. P. Chandrakasan, "An energy-efficient reconfigurable dtls cryptographic engine for end-to-end security in iot applications," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, pp. 42–44, Feb 2018.
- [8] S. Satyadevan, B. Kalarickal, and M. Jinesh, "Security, trust and implementation limitations of prominent iot platforms," in *Proceedings of the 3rd*

International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014. Advances in Intelligent Systems and Computing, vol. 328, Springer, 2015.

- [9] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of things: Security vulnerabilities and challenges," in *2015 IEEE Symposium on Computers and Communication (ISCC)*, pp. 180–187, July 2015.
- [10] I. B. Djordjevic, "Fbg-based multidimensional qkd," in *2018 20th International Conference on Transparent Optical Networks (ICTON)*, pp. 1–5, July 2018.
- [11] I. Tittonen, O. Tirkkonen, J. Lietzen, M. Kiviranta, T. Manninen, Ülo Parts, and I. Elonsalo, "Applications of the Quantum Key Distribution (QKD) Method," *MATINEn tutkimusseminaari*, 2016.
- [12] L. Salvail, M. Peev, E. Diamanti, R. Alleaume, N. Lütkenhaus, and T. Länger, "Security of trusted repeater quantum key distribution networks.," *Journal of Computer Security*, vol. 18, no. 1, pp. 61 – 87, 2010.
- [13] M. Peev, C. Pacher et. al., "The SECOQC quantum key distribution network in vienna," *New Journal of Physics*, vol. 11, jul 2009.
- [14] E. Meyer-Scott, Z. Yan, A. MacDonald, J.-P. Bourgoin, H. Hübel, and T. Jennewein, "How to implement decoy-state quantum key distribution for a satellite uplink with 50-db channel loss," *Phys. Rev. A*, vol. 84, p. 062326, Dec 2011.
- [15] Y. Zhao, Y. Cao, X. Yu, and J. Zhang, "Software defined optical networks secured by quantum key distribution (qkd)," in *2017 IEEE / CIC International Conference on Communications in China (ICCC)*, pp. 1–4, Oct 2017.
- [16] A. Aguado, E. Hugues-Salas, P. A. Haigh, J. Marhuenda, A. B. Price, P. Sibson, J. E. Kennard, C. Erven, J. G. Rarity, M. G. Thompson, A. Lord, R. Nejabati, and D. Simeonidou, "Secure nfv orchestration over an sdn-controlled optical network with time-shared quantum key distribution resources," *Journal of Lightwave Technology*, vol. 35, pp. 1357–1362, April 2017.
- [17] P. Sibson, C. Erven, M. Godfrey, S. Miki, T. Yamashita, M. Fujiwara, M. Sasaki, H. Terai, M. G. Tanner, C. M. Natarajan, R. H. Hadfield, J. L. O'Brien, and M. G. Thompson, "Chip-based quantum key distribution," *Nature Communications*, vol. 8, Feb 2017.
- [18] A. Mavromatis, F. Ntavou, E. H. Salas, G. T. Kanellos, R. Nejabati, and D. Simeonidou, "Experimental demonstration of quantum key distribution (qkd) for energy-efficient software-defined internet of things," in *2018 European Conference on Optical Communication (ECOC)*, pp. 1–3, Sep. 2018.
- [19] O. Elmabrok and M. Razavi, "Wireless quantum key distribution in indoor environments," *J. Opt. Soc. Am. B*, vol. 35, pp. 197–207, Feb 2018.

Machine Learning for image and video encoding

Mohammad Elhariry

mohammad.elhariry@aalto.fi

Tutor: Gazi Illahi

Abstract

Digital media, including images and videos, represent a large portion of internet traffic. Although telecommunication bandwidths are vastly expanding, it is not sufficient to solely depend on this expansion to provide adequate performance for multimedia streaming. As a result, the need for image and video compression arises, leading to an increase in the significance of compression as a research topic. There are various methods and algorithms to perform image and video compression, and as machine learning algorithms become more integrated into various applications, compression is no exception, because Machine Learning has the capability of improving - or replacing - some traditional algorithms. This paper surveys the state of the art in image and video encoding using Machine Learning, through analyzing different algorithms and comparing their results.

KEYWORDS: *Image compression, video compression, machine learning, neural networks*

1 Introduction

In recent years, image encoding has become a crucial topic of research due to the increase of digital media transmitted over the internet. Image encoding is the process of compressing an image to be represented by a fewer number of bits without causing a noticeable effect on the quality of the original image. The main aim of this process is to minimize both the transmission bandwidth and the space needed to save an image either in memory or storage, e.g., hard-drives or flash drives [14]. Video encoding is fairly similar to image encoding since a video is formed of multiple frames of images. For both image and video encoding, the transmitter encodes the data prior to transmission and the receiver reverses the procedure on reception, which is known as decoding or decompression.

One approach for compressing images is to group the patterns found in images into features, such as edges, points or objects, then statistically analyze those features. The features that are frequently repeated can be represented by shorter bits whereas the less frequent features are represented by longer bits. Instead of detecting those features using traditional algorithms, they can be learnt by utilizing neural networks, as neural networks can automatically learn an unconstrained number of features [5]. As a result, new algorithms that rely on neural networks and other machine learning techniques have been introduced to achieve more efficient encoding at high compression rates. However, less research has been invested to survey and compare the performance of those algorithms. This paper aims to present a few machine learning-based algorithms, analyze their behavior and compare their results.

The rest of the paper is organized as follows. Section 2 introduces the procedure of traditional image and video encoding and discusses the usage of machine learning to address the challenges of encoding. Section 3 introduces the usage of various types of neural networks for image compression, including deep, convolutional and recurrent neural networks. Section 4 describes a real-time approach to perform adaptive image compression [17]. Section 5 presents an algorithm that combines neural networks and temporal subsampling to compress both images and videos [3]. Section 6 reports the results of the previously mentioned algorithms and discusses their performance. Finally, Section 7 provides the conclusion to the paper.

2 Background information

2.1 Image and video compression

Image compression is the process of altering an input image to reduce its size prior to transmission or storage. Two main schemes are used to achieve this, lossless and lossy schemes. In lossless encoding schemes, the receiver retrieves the whole image, but it achieves lower compression rates in comparison to the compression rates achieved by lossy alternatives. Lossless compression can be used in applications where the quality and the level of detail of the reconstructed images is important, like medical imaging. A few known image formats adopt lossless compression, such as Portable Network Graphics (PNG). On the other hand, lossy compression is present in applications where image quality can be compromised to achieve higher compression rates, such as streaming media. Lossless compression typically achieves compression rates varying from 1:1 to 1:3, whereas lossy alternatives can reach compression rates up to 1:300 [15]. Therefore, lossy encoding algorithms are more commonly implemented and aim to minimize the level of data loss or distortion in the encoded image, as long as these algorithms guarantee that the difference between the original image and the compressed one is not noticeable to the human eye. One example for lossy compression is the Discrete Cosine Transform algorithm, which is a main building block of the Joint Photographic Experts Group (JPEG) standard [9]. In this section, JPEG encoding is used as an example for illustrating the steps performed by a traditional image encoding algorithm to compress an input image.

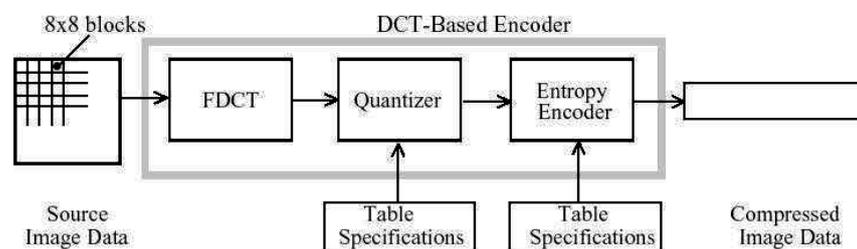


Figure 1. JPEG encoding block diagram. Adopted from [9].

The fundamental building blocks of an image encoder are shown in Figure 1. These components include splitting an image into smaller building blocks, known as macroblocks, transformation, e.g., Discrete Cosine

Transformation (DCT), quantization and entropy coding. Transformation is a linear and reversible process that compacts the representation of an input set into a fewer number of coefficients. The transformation phase results in a matrix that is passed to the quantization phase, which divides the transformed matrix by a quantization matrix and rounds the values of the resultant matrix. The aim of quantization is to minimize Psycho-visual redundancy by removing information in the image that is not detected by the human visual system. Finally, entropy coding removes duplicated bit patterns from the output of the quantizer. The decoding process simply reverses the previous operations to reconstruct the image from the encoded data [1]. Video encoding has the same components of image encoding. However, it has one extra step, known as prediction. After dividing the image into macroblocks, the encoder tries to predict each macroblock based on information from the current frame, known as intra-frame prediction, or from other frames that were already coded in previous iterations, known as inter-frame prediction.

The performance of encoding algorithms can be measured by a few metrics, such as Bits per pixel (BPP), Peak signal-to-noise ratio (PSNR) and Multi-Scale Structural Similarity Index Metric (MS-SIM). BPP has a direct correlation to the compression rate, hence a lower BPP represents a higher compression ratio. The PSNR is defined by Eq. (1),

$$PSNR = 20 \cdot \log_{10}(MAX_I) - 20 \cdot \log_{10}(MSE) \quad (1)$$

where MAX_I represents the maximum possible value of a pixel and MSE is the mean squared error. However, MS-SSIM provides the most powerful metric as it combines the luminance, contrast and structure of an image at multiple scales resulting in a more accurate measure for image quality [13]. The three mentioned metrics will be referenced throughout the paper when comparing the performance of the surveyed algorithms.

2.2 Machine Learning based compression

Although JPEG has been prevalent for the past decades, it has the disadvantage of utilizing linear transformation which can cause artifacts in form of blurriness or pixelation when decoding natural images. These artifacts appear because linear transformation is a predefined process, which may not be optimal for all inputs. In contrast to JPEG or other traditional algorithms, machine learning based algorithms can optimize the encoder and decoder models to reduce the resulting artifacts in re-

constructed images and achieve optimal results. However, two key challenges face machine learning based algorithms. First, correctly modelling the encoder and decoder in an efficient manner to outperform engineered compression schemes, which may require adding more components to the image compression pipeline [12]. Second, most machine learning algorithms are computationally expensive, which poses another challenge as it is difficult to design a machine learning algorithm that can be deployed on devices of limited processing speed, battery life or memory [17]. In the next sections, different machine learning based approaches are discussed that successfully address these challenges.

3 Neural networks and image compression

The algorithms presented in [4], [6] and [8] discuss various machine-learning approaches for image compression that adopt artificial neural networks (ANNs). An artificial neural network is an interconnected network formed of numerous processing nodes known as artificial neurons. An artificial neuron receives multiple input signals and performs non-linear computations to calculate a single output signal. A neuron multiplies each signal with its predefined weight, sums these weighted inputs and finally passes the result through an activation function to calculate the output signal. These neurons are organized into a number of layers, including input, output and several hidden layers [4].

In the next subsections, three different approaches are explained that utilize deep, convolutional and recurrent neural networks respectively.

3.1 Deep Neural Networks

The algorithm proposed in [4] provides an image compression scheme using deep neural networks (DNNs). Deep neural networks comprise numerous hidden layers, allowing more efficient modelling for non-linear relations as they can detect more features of the input [4].

The proposed solution presents two different DNNs, which have the same number of hidden layers n , but each uses a different activation function. The first function is the sigmoid function, which is defined as $\delta(x) = 1 / (1 + e^{-x})$ and limits the range of the output from 0 to +1; whereas the second one is the hyperbolic tangent function, which is defined as $\tau(x) = (e^x - e^{-x}) / (e^x + e^{-x})$ and limits the range of the output from -1

to +1. To achieve compression and decompression, two conditions have to be satisfied. First, the input and output layers must comprise the same number of neurons N to be able to retrieve the original image from its compressed form. In addition, the number of neurons in the last hidden layer K is strictly smaller than N . Both K and N are determined based on the target compression ratio which is represented as $K : N$ [4].

The DNN is trained using supervised learning since the desired outcome is known beforehand. The algorithm starts by splitting the image into $W * W$ macroblocks, then applies a normalization function f to these macroblocks and passes them through the network. Both the input layer and the hidden layers aim to compress the image by performing an orthogonal transformation V , hence the compressed image is present at the last hidden layer. On the other hand, the output layer acts as the decompressor and aims to reconstruct the image by performing a different transformation T followed by an inverse normalization f^{-1} . The network is trained by back propagation, hence the mean squared error is propagated back to readjust the weights of the DNN. The whole process is repeated until the mean square error drops below a specified threshold or the number of specified iterations is reached. In addition, the transformation V and T are adjusted by training on various training sets.

3.2 Convolutional Neural Networks

The algorithm proposed in [6] is inspired by the previous work of using convolutional neural networks (CNNs) in image compression and computer vision, which is thoroughly discussed in [2] and [19]. Convolutional neural networks are neural networks that have at least one layer of convolution, which is achieved by having shared weights between the neurons of the same layer making the forward propagation of data in the network equivalent to applying a filter to the image to result in a new image [16].

The suggested algorithm utilizes an image codec to perform the encoding and decoding process, such as JPEG, JPEG2000 or BPG. Furthermore, the architecture includes two CNNs, a compact representation CNN (ComCNN) and a reconstruction CNN (RecCNN). The input image passes through ComCNN, the image codec and RecCNN respectively. ComCNN constitutes three layers, each layer applies 64 filters to the image. Moreover, the first two layers use ReLu as an activation function which is defined as $y = \max(x, 0)$. Each layer has a specific goal, the first layer aims to generate feature maps so that the second layer can use these maps to

downscale the image and improve its features. Finally, the third layer constructs a condensed representation of the original image. The output of ComCNN is encoded and decoded by the image codec, then upsampled using interpolation and passed through to RecCNN. RecCNN aims to reconstruct the image and constitutes 20 layers. The first layer applies 64 filters to the image and also uses ReLu to generate feature maps. The next layers, except for the last, apply a similar operation, however, these layers perform batch normalization before applying ReLu. Batch normalization is a technique used to normalize the input of each layer to enhance the training process. Since the distributions of activation functions constantly vary during training, each layer must learn to adapt to these changes. However, batch normalization enforces an approximately identical distribution for each training step to overcome this problem. As a result, the whole model converges in considerably lower time [18]. Finally, the last layer of RecCNN reconstructs the image.

Back propagation of the loss function is used to train both ComCNN and RecCNN, and this function is calculated as the mean squared error between the reconstructed image and the original image. The training process is repeated for a predefined number of iterations T .

3.3 Recurrent Neural Networks

The methods proposed in [8] provide several alternatives to compress images regardless of their quality and size using recurrent neural networks (RNNs). These methods extend the previous work done by the same research group, in which an algorithm that utilizes recurrent neural networks was designed to outperform state of art compression methods [7]. However, that algorithm was limited to images of size $32 * 32$, hence these researchers developed a more advanced method to compress images of arbitrary size, which utilizes RNNs as well. Recurrent neural networks differ from other types of neural networks as they have a notion of order in time, since these networks consider both the current input and the recent past using feedback loops which make the information persistent [10].

The algorithm modifies the encoder and decoder models to include recurrent components, including long short-term memory (LSTM) or gated recurrent units (GRU). Both LSTMs and GRUs address the vanishing gradient problem which occurs in RNNs when gradients diminish while back propagating through the network. Both components aim to keep relevant information and discard non-relevant data through internal mechanisms

called gates, which regulate the flow of information [11]. LSTMs achieve this by passing information through three types of gates, a forget gate, an input gate and an output gate. First, the forget gate decides which information to keep by using sigmoid activation, hence the values close to 0 are discarded and the ones close to 1 are kept. Second, The input gate aims to calculate the new cell state by applying both sigmoid and hyperbolic tangent activations to the current input and the previous hidden state, then multiplying the result. Finally the output gate computes the new hidden state, which contains information from previous inputs, through applying a sigmoid activation to the previous hidden state and the current input, applying a hyperbolic tangent activation to the newly computed cell state and multiplying both results. On the other hand, GRUs comprise only two gates, an update gate and a reset gate. The update gate combines both the forget and input gates of LSTMs, whereas the reset gate decides on how much past information to eliminate [11].

In every iteration, the image is first encoded, using one of the previously discussed recurrent units, then transformed into binary codes by the binarizer. Finally, the decoder aims at estimating a reconstructed image based on these binary codes. The loss function used is the sum of weighted residuals of each iteration, where the residual is the difference between the original image and the reconstructed image. The proposed reconstruction schemes include one-shot reconstruction and additive reconstruction. One-shot reconstruction implies that the decoder aims to predict the whole image in each iteration then passes the residual to the next iteration, while in additive reconstruction the decoder intends to reconstruct only the residual of previous iterations and the whole image is retrieved by adding the outputs of all iterations [8].

The network is trained for around one million steps by two sets of images, a set of 32 x 32 images previously gathered to test the algorithm developed in [7] and a second set that includes random tiles from images that had the lowest compression ratio when using the PNG algorithm [8].

4 Real-time adaptive image compression

The algorithm proposed in [17] alters the traditional compression pipeline by adding a few main components, including feature extraction and a Generative Adversarial Network (GAN) as shown in Figure 2. Moreover, it replaces traditional coding algorithms with adaptive arithmetic coding.

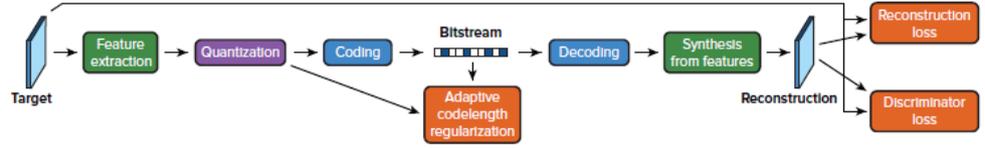


Figure 2. Block diagram of the model proposed in [17].

4.1 Feature extraction

Traditional compression algorithms use Discrete Cosine Transform (DCT) or Discrete Wave Transform (DWT) as a first step of the encoding process, which is a linear operation performed by the convolutions of two predefined filters independent of scale or the number of channels. On the other hand, the idea of this algorithm is to replace this phase with individual non-linear operations for each scale to achieve optimal transformation. This operation is known as pyramidal decomposition. Assuming we have M scales and x_m inputs for each scale and $x_m = x$ for $M = 1$ initially. First, coefficients are extracted according to a function f_m that has parameters, such as the height H , width W and the number of channels C of the input image. Then, the input x_m is down-sampled and passed to the next scale $x_m + 1$. For each scale, the feature extractor computes multiple convolution operations. Second, a process called interscale alignment is performed to gather the coefficient matrices c_m from multiple scales while ensuring the output size is constrained by the number of channels, height and width defined for f_m . This is achieved by mapping each matrix x_m to the required dimensions according to a another function g_m , adding each $g_m(c_M)$ and performing a final non-linear transformation. The final output is a single matrix of the dimensions $C * H * W$. The value of $M = 6$ was used in the implementation of the algorithm.

4.2 Generative Adversarial Network

The general idea of Generative Adversarial Networks is to build two networks, a generator and a discriminator. The generator aims to generate outputs based on the ground truth provided, which is a set of actual images in this algorithm whereas the discriminator aims to differentiate between two given images, one sample from the ground truth and a sample from the output of the generator [17]. The goal of this process is to maximize the efficiency of the generator, so that it can generate images that the discriminator cannot differentiate from the ground truth. Ac-

According to the suggested architecture, the whole encoding-decoding process represents the generator and is trained according to a loss function, called reconstruction loss. In addition, the discriminator is trained by another loss function, the discriminator loss. The discriminator is trained to solve the problem of deciding which of the two input images is the real one. The procedure is defined as follows, first the input images are randomly swapped according to a uniform distribution then the images are propagated to the discriminator's network. Second, the discriminator accumulates outputs at different scales, since image artifacts can appear at different layers of the network, and applies a final sigmoid function. To train both networks, an error matrix that represents the performance of the generator, is propagated to the generator and the accuracy a of the discriminator is compared against two bounds, a lower bound $L = 0.8$ and an upper bound $U = 0.95$. Then, the decision is made as follows:

- if $a \leq L$: Train the discriminator.
- if $L < a < U$: Propagate the error matrix and train the discriminator.
- if $a \geq U$: Propagate the error matrix.

5 Neural Networks and Temporal Subsampling for video compression

The algorithm proposed in [3] aims to compress videos in real-time by using a neural network based still-image compressor and decompressor, motion detection and temporal subsampling. To begin with, the neural network comprises a single hidden layer, where the weights between the input layer and this hidden layer represent the compression process, whereas the weights between the output layer and the intermediate layer represent decompression. The network uses a similar approach to the one explained in Section 3.1, where the image is also split into macroblocks and the sigmoid function is used for activation. However, the network is trained on a single image hence, the quality of the output images is just adequate. The algorithm then proceeds to perform motion detection, which is achieved by calculating the difference between the same macroblocks in two subsequent frames $F_i - 1$ and F_i and comparing it to a

predefined threshold d . The decision is made as follows: if the difference is greater than d , motion is detected and the block is compressed and transmitted, otherwise the block is considered unchanged and is neither compressed nor transmitted. The block is then compressed by distinct concurrent networks and the decompressor decides which output to consider from these networks to reconstruct the image based on the user-defined quality factor Q .

Even though this compression scheme is not computationally expensive, it cannot perform real-time compression over low bit-rate networks. Therefore, the algorithm discards a few frames and reconstructs them during decompression without altering the frame rate. The algorithm achieves that by perceiving the video as a set of 1-D temporal functions instead of 2-D spatial mappings with motion vectors for each pixel. Temporal subsampling is then applied as follows, each S^{th} frame is compressed by the neural network and transmitted, while the remaining $S - 1$ frames are discarded. Hence, $1/S$ of the frames are retrieved at the decompressor, and the missing frames are reconstructed pixel by pixel using cubic spline interpolation [3].

6 Results and Discussion

The first algorithm [4] does not provide any comparison to traditional encoding schemes, however, it shows that the hyperbolic tangent variant surpassed the sigmoid alternative with roughly +3db in $PSNR$ on average at various compression rates, varying from 4 : 1 to 16 : 1. In addition, the DNNs using the hyperbolic tangent as an activation function converged much faster. The three other image encoding algorithms were able to outperform JPEG. Regarding the second algorithm [6], the experiments proved an average gain of +3.4db in $PSNR$ compared to JPEG and 0.8535 in SSIM and average gain of +1.985db in $PSNR$ across bit rates varying from 0.1bpp to 0.4bpp. Third, a few of the recurrent units presented in [8] surpassed JPEG, including GRU and residual GRU with one shot reconstruction, showing roughly +1.9db gain in $PSNR$ and 0.04 in $MS - SIM$. Finally, the images compressed by the real-time algorithm presented in [17] are almost half the size of the images compressed by other alternatives, including JPEG, JPEG2000 and WebP across different compression ratios. Moreover, this algorithm encoded and decoded the images in 8.6ms and 9ms respectively, whereas JPEG required more

computation time on average, 18.6ms and 13ms respectively.

The video encoding algorithm discussed in [3] provides an efficient alternative for traditional algorithms at low bit rates. Furthermore, the performance of the proposed algorithm degrades at a much slower rate compared to H.261, even without using temporal subsampling, and can achieve compression rates up to 500 : 1, while H.261 can only achieve 150 : 1.

Despite the outstanding performance of a few algorithms, a few issues remain unaddressed. First, providing a metric that measures perceptual differences since *PSNR* and *MS-SIM* cannot fully match human visual perception or address all kinds of distortion. In addition, both metrics could be contradicting, for example in comparing the results of recurrent units presented in [8]. Moreover, a few of the algorithms require numerous training steps which is computationally expensive, e.g., the algorithm presented in [8].

7 Conclusion

Image and video compression are extremely important for various applications including streaming, medical imaging and entertainment. Over the years, various traditional standards have been developed and implemented. However, neural networks provide an exceedingly strong alternative for the currently used image and video compression algorithms, as they have the capability of adapting to the given input and achieving high performance results against to well-defined metrics. Several algorithms have been surveyed, including four algorithms that focus on still image compression and one algorithm for video compression. Results show that a few approaches are able to outperform traditional alternatives, such as [7], [17] and [6]. In addition, the video compression approach explained in [3] provides a more robust method to perform real-time video encoding compared to the H.26x family. However, finding an effective metric that resembles human visual perception and a robust method to train complex neural networks remain open challenges, which can set the foundation for future work.

References

- [1] A.M.Raid, W.M.Khedr, M. A. El-dosuky and Wesam Ahmed. Jpeg Image Compression Using Discrete Cosine Transform. *arXiv:1405.6147, International Journal of Computer Science & Engineering*, 2014.
- [2] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang. Compression Artifacts Reduction by a Deep Convolutional Network. In *IEEE International Conference on Computer Vision*, pages 576–584, 2015.
- [3] Christopher Cramer, Erol Gelenbe and H. Bakircoglu. Low bit-rate video compression with neural networks and temporal subsampling. In *Proceedings of the IEEE 84, no. 10*, pages 1529–1543, 1996.
- [4] Farhan Hussain and Jechang Jeong. Exploiting deep neural networks for digital image compression. In *Web Applications and Networking (WSWAN), 2015 2nd World Symposium on*, pages 1–6. IEEE, 2015.
- [5] Fei Hu, Changjiu Pu, Haowei Gao, Mengzi Tang and Li Li. An Image Compression and encryption scheme based on deep learning. In *arXiv preprint arXiv:1608.05001*, 2016.
- [6] Feng Jiang, Wen Tao, Shaohui Liu, Jie Ren, Xun Guo and Debin Zhao. An End-to-End Compression Framework Based on Convolutional Neural Networks. *arXiv:1708.00838, IEEE Transactions on Circuits and Systems for Video Technology*, 2017.
- [7] Sung J. Hwang Damien Vincent David Minnen Shumeet Baluja Michele Covell George Toderici, Sean M. O'Malley and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. In *ICLR 2016*, 2016.
- [8] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor and Michele Covell. Full Resolution Image Compression with Recurrent Neural Networks. In *CVPR*, pages 5435–5443, 2017.
- [9] G.K. Wallace. THE JPEG STILL PICTURE COMPRESSION STANDARD. *IEEE Transactions on Consumer Electronics*, 38, 1992.
- [10] Juergen Schmidhuber. Deep Learning in Neural Networks: An Overview. *arXiv:1404.7828*, 2014.
- [11] Kazuki Irie, Zoltan Tuske, Tamer Alkhouli, Ralf Schluter, Hermann Ney. LSTM, GRU, Highway and a Bit of Attention: An Empirical Overview for Language Modeling in Speech Recognition. In *INTERSPEECH 2016*, 2016.
- [12] Ken Nakanishi, Takeru Miyato and Daisuke Okanohara. Neural Multi-scale Image Compression. In *arXiv preprint arXiv:1805.06386v*, 2018.
- [13] M.Abdel-Salam Nasr, Mohammed F.AlRahmawy, A.S.Tolba. Multi-scale structural similarity index for motion detection. *Journal of King Saud University - Computer and Information Sciences*, 29:399–409, 2017.
- [14] Manoj Kumar and Dr. Manish Verma. Digital Image Processing Tool and Imaging Technique through Crossbreeding Wavelet and Cosine Transformation. *International Journal of Electrical Electronics and Computer Science Engineering*, 4, 2017.

- [15] Michael Thierschmann, Uwe Martin and Reinhard Rosel. New Perspectives on Image Compression. *Photogrammetric Weeks*, 1997.
- [16] Nal Kalchbrenner, Edward Grefenstette and Phil Blunsom. A Convolutional Neural Network for Modelling Sentences. *arXiv:1404.2188*, 2014.
- [17] Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In *arXiv preprint arXiv:1705.05823*, 2017.
- [18] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv:1502.03167v3*, 2015.
- [19] Zhangyang Wang, Ding Liu, Shiyu Chang, Qing Ling, Yingzhen Yang, and Thomas S. Huang. D3: Deep Dual-Domain Based Fast Restoration of JPEG-Compressed Images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2764–2772, 2016.

Automated testing of Deep Neural Networks

Daniel Saxén

daniel.saxen@aalto.fi

Tutor: Jukka K. Nurminen

Abstract

Most deep neural network (DNN) training data sets are manually labeled, which leads to certain corner cases not being taken into account. In safety-critical systems these corner cases may have fatal consequences. There are some proposals for testing these corner cases automatically. In this seminar paper, we discuss the use of such automated testing frameworks, and in particular focus on DeepXplore, a testing framework developed in 2015. We compare how DeepXplore and other automated DNN testing frameworks work, what their strengths and weaknesses are, and what kind of applications these systems are most applicable for.

The comparison showed that the testing frameworks are getting increasingly sophisticated, but are still not comprehensive enough to cover all possible perturbations. The frameworks aimed for more specific problems show more sophistication, but DeepXplore is the most comprehensive out of the four chosen frameworks.

KEYWORDS: *deep neural network, deep neural network testing, automated testing, neuron coverage*

1 Introduction

Artificial intelligence (AI) is often defined as intelligence displayed by machines. It has gained increased popularity during the past decade. There are several reasons why AI has become such a popular research topic, but some of the key enablers for AI can be viewed to be big data, high-speed internet, increased CPU performance and the digitization of data. There are several focus areas in scientific research which belong under AI, and in this paper we focus on some aspects within Machine Learning (ML), which is a subgroup of AI.

ML concepts were already discovered during the 20th century. ML has gained lots of publicity because researchers have been able to apply ML techniques in several real-world applications, including end-to-end autonomous vehicles [1], image classification [2] and online content personalization [3]. An important subgroup of ML is called deep learning (DL). In this paper, we investigate an important part of deep learning called deep neural networks (DNNs) and how these DNNs can be tested in an efficient way.

A deep neural network mimics the infrastructure of interconnected neurons in our brains. When speaking of DNNs, we usually refer to several hundreds of thousands, if not millions of neurons which all have a particular weight. These neuron weights describe how strongly the neurons are interconnected. An activation function decides how big an input (multiplied by the weight) is required in order for the neuron to be activated. This limit value is referred in literature as activation threshold. The neuron weights are assigned during the training phase of the DNN, usually by minimizing a certain cost function. Cost functions measure how wrong the model is compared to the nowadays mostly manually labeled true answers.

There are several layers of neurons in a DNN. Usually a DNN consists of an input layer in which inputs are fed, an output layer which displays the answers the DNN has come up with many hidden layers which consist of weighted neurons. All neurons have connections to at least one, but often several other neurons in other layers. These DNN neurons are activated if a certain condition is met and this activation criteria is unique for each neuron. For example when classifying hand-written numbers, a DNN neuron would be activated if a certain shape of a number is present in the image or not [4]. Automated classification works because the DNN

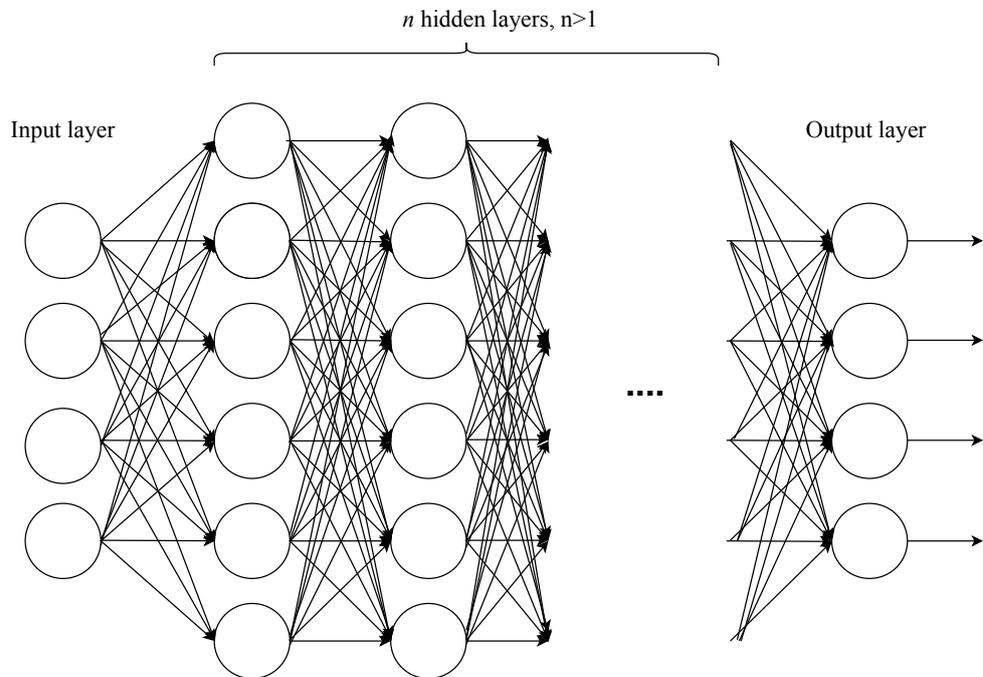


Figure 1. A deep neural network, consisting of an input layer, several hidden layers and an output layer.

has been trained with labeled training data. In the case of a hand-written number data set, a human has to label these images with what actual number the handwriting should represent. This kind of labeling is slow because it requires manual work for huge amounts of data. In general, the manual labeling is imperfect because in real-world situations no amount of data can cover every possible option of events.

There are some well-known cases where sensor detection errors and/or insufficient DNN testing have caused serious injuries. For example, past self-driving car accidents have caused fatal injuries: in two cases a Tesla driver suffered fatal injuries and in one case a self-driving system caused a pedestrian fatality, partly because of an Uber driver not paying attention to the road. These cases are interesting because it shows that autonomous cars, in which no driver would have to be behind the wheel, is not yet realistic.

Car manufacturers share in general little or no technical data with the public regarding their self-driving car development, so our data is limited to the car manufacturer's public blog posts, statements and other non-scientific news sources. The were different reasons for these accidents, usually the key reason being that the driver was not paying attention to the traffic. For example, the first fatality linked to Tesla's autopilot included very challenging lighting conditions [5], which caused the car camera not to detect a nearby tractor-trailer. The Tesla driver in this accident

ignored audible warnings about the nearby tractor-trailer. The reason for the Uber accident, which involved the first pedestrian fatality caused by a self-driving car, is believed according to the preliminary report [6] by the National Transportation Safety Board to have been caused by the car software system misclassifying the pedestrian as an "unknown object, as a vehicle, and then as a bicycle". It is possible that there are some critical edge cases which were not taken into account in the development of the DNNs in the self-driving software system used by Uber. Even though the systems are autonomous to some degree, they are designed by humans who make mistakes. The fact that humans make design mistakes is a well-motivated reason for researching and developing automated DNN testing frameworks.

2 DeepXplore

DeepXplore [7] is a white-box framework designed for DNN testing. By white-box we mean that DeepXplore also knows the internal structure of DNNs, but cannot edit them in any way. The developers of DeepXplore claim that their testing framework can "automatically identify erroneous behaviors" in the DNN. The main focus of the DeepXplore system is to gain insight into how you can test a DNN more thoroughly, and what kind of inputs cause erroneous behaviors.

The DeepXplore system tries to find erroneous behaviors, such as wrong self-driving car steering angle decisions, by testing the DNNs in several ways. The first method focuses on maximizing the amount of differences in outputs when using different DNNs. The goal of this method is to find which inputs cause the greatest difference between DNNs in classification. If several DNNs give different outputs, you know that at least some of them exhibit erroneous behaviors. This kind of testing can be compared with differential testing used by classical software systems. For example, if two DNNs label a picture to have a clown in it and a third DNN labels the picture to have a car in it instead, we could assume that the third DNN is not working correctly. In a real test setting you would naturally use more than three DNNs and lots of different pictures of clowns, because otherwise it would be difficult to exactly pinpoint which DNNs display wrong behavior. The key notion with a DNN comparison like this is that the DNNs should be trained independently and with different data parameters. The training of these DNNs requires manual work and might

cause a bottleneck for DeepXplore testing, especially if there are not several DNNs already available.

What the DeepXplore system also tries to cover is activating a maximum amount of neurons by using gradient ascent, because an increased amount of activated neurons tests the neural network more thoroughly. The neurons which are not activated are detected and handled by the DeepXplore system in an iterative way. In the DeepXplore article, the researchers call this metric neuron coverage (NC). NC can be compared to code coverage used by classical software systems, but code coverage is not suitable for DL systems because DL systems are not written by humans. The researchers of DeepXplore also point out that similar to code coverage, a high NC is more difficult to achieve when the amount of neurons increases. The aim of NC is to be a metric of how comprehensive the provided test data is.

The testing framework is based on two DL libraries, TensorFlow and Keras. The whole DeepXplore test system is coded in Python and it is publicly available on Github. Both TensorFlow and Keras are open-source which is preferable for individual developers and small companies. The use of TensorFlow in DeepXplore is motivated by the fact that TensorFlow has efficient gradient computation properties and sub-DNN creation support. Keras is a open-source deep learning library which is designed to enable developers to quickly prototype DNNs.

In the DeepXplore article, the authors discuss that they have tried the testing framework on five real-life data sets: Drebin containing both clean and malicious Android applications, Driving with Udacity self-driving car challenge [8] data sets, ImageNet with image data, MNIST also with image data and PDF malware data collected by Contagio/VirusTotal. These data sets vary a bit in nature, three of them focus mainly on visual data and two on detecting malware. The researchers of DeepXplore then design the constraints, which are for the image data sets perceptible and simulate natural road conditions.

DeepXplore managed to find several erroneous behaviors in each of the 15 DNNs. For example, with the malicious PDF DNNs when selecting 2,000 random seeds from test sets, DeepXplore found around 1,000 differences for each of the three Contagio/VirusTotal DNNs. DeepXplore found over a thousand of difference-inducing inputs in at least two of three DNNs used in each of the five data sets. The neuron coverage metric showed that using 10 randomly selected seed inputs, the DeepXplore sys-

tem measured very varying numbers. The worst neuron coverage (under 5%) was present in the ImageNet and Driving test sets, while the more sophisticated and simpler MNIST data set got over 30% code coverage with the 10 randomly selected inputs.

One of the major imitations of DeepXplore is that it requires you to use several different DNNs in order to find the difference-inducing inputs. This is also pointed out in another article, which we will discuss in the next section. The authors of DeepXplore also highlight that if the used DNNs for a particular data set are too similar, the time to find difference-causing inputs increases.

Another shortcoming with DeepXplore is that the difference-inducing inputs, for example when testing the Driving dataset, do not describe several different image conditions. The authors used image seeds with only varying lighting conditions and occlusions with one big or multiple small rectangles. Researches of other frameworks, which will be discussed below in section 3, have made their framework solutions generate and test more sophisticated image perturbations.

3 Comparison between DeepXplore and other automated DNN testing solutions

In this section, we compare the DeepXplore testing framework to other related research proposals. We also discuss what strengths and weaknesses each of the proposed frameworks have, and which testing framework are the most suitable for certain DL applications. The comparison is summarized in Table 1, at the end of this section.

3.1 DLFuzz

DLFuzz [9] is another DL testing framework with focus on fuzzing testing. Fuzzing testing is about manipulating the inputs in a way to make the system deem them invalid. With fuzzing testing, the aim is to find as many inputs as possible which could cause a system to crash or express odd behavior. The researchers who developed DLFuzz argue that fuzzing is faster than using several different DNNs for differential testing. DLFuzz uses a mutation algorithm, which includes neuron selection strategies. The main strategies for selecting neurons include selecting neurons which appear more infrequently and neurons which have more weight.

DLFuzz was tested with the ImageNet and MNIST data sets and was found to get a higher neuron coverage (NC) and also generate more difference-inducing inputs than DeepXplore. DLFuzz seems in other words to be more efficient than DeepXplore considering image classification. A key difference to the DeepXplore tests is that DLFuzz focused on imperceptible, i.e. adversarial input modifications. This kind of approach is not as broad because we can assume that self-driving cars have extensive security platforms in order to prevent adversarial images making it into the image detection system. These kind of security modules are unfortunately, but for good reasons, not public knowledge. DLFuzz uses and implements the same definition of NC as DeepXplore, but uses fewer hyperparameters than DeepXplore in the joint optimization problem, which consists of both maximizing the NC and differences in output based on input. A shortcoming, which the DLFuzz developers also mentioned, was that when the amount of neurons in a DNN model grew bigger, the DLFuzz framework did not scale as good as DeepXplore. Although with a smaller amount of neurons, DLFuzz was faster to generate adversarial inputs than DeepXplore.

3.2 DeepTest

DeepTest [10] is another example of a DNN testing framework. DeepTest focuses on test automation concerning the input images fed into the DNNs which make steering angle decisions for self-driving cars. DeepTest has another approach to the problem: The DeepTest system applies transformations to the seed images, i.e. creating synthetic images in order to find the corner cases. The developers of DeepTest take several environmental conditions into account.

In the DeepTest article, the main research regards making a prototype which could be built upon for real self-driving car testing. The DeepTest system, much like DeepXplore, focuses on natural road conditions which may be challenging for DNNs. The researchers who developed DeepTest alter the input images in several ways, e.g. by rotating, shearing, blurring, changing contrast and adding fog and rain effects. The DeepTest framework uses metamorphic testing in order to see how much the transformed images differ from the original ones. Metamorphic testing is based on the idea that if we know the metamorphic relations between multiple inputs, the outputs should also be related and thus able to be tested.

DeepTest also defines their own neuron coverage, but for several sub-

groups of DNNs, including recurrent neural networks, RNNs. The framework does not use differential testing, but varying the image transformations seem to activate different neurons, which contributes to a higher neuron coverage. By combining different transformations and coordinating the image transformations, the framework was able to achieve higher neuron coverage. They also achieved a more accurate DNN by retraining it with the DeepTest-transformed images.

The main problem with DeepTest is that while it tries to address the shortcomings of manual test situations, you still have to manually choose what kind of basic input image disturbances are possible, still leaving out potential real-life road conditions. The same problem persists as with DeepXplore: there are even more different weather and lighting conditions which could affect the decision making which are not taken into account with DeepTest.

3.3 DeepRoad

The final testing framework discussed more in depth in this paper is called DeepRoad [11], which is a Generative Adversarial Network (GAN)-based metamorphic testing framework implemented with use of Pytorch and Python. A GAN consists of two unsupervised competing neural networks. The aim of a GAN framework is that one of the neural networks (generator) creates new images based on training data, while the other neural network (classifier) tries to make a guess whether or not the image belongs to the training images or the images created by the generator neural network. If a GAN is well designed, both of the networks should become more and more sophisticated because they are "competing" against one another - the generator network tries to trick the classifier network with increasingly more realistic images, while the classifier network becomes progressively better at detecting the real and generated images.

The DeepRoad article can be seen as an upgrade to the DeepTest work, because the authors of DeepRoad motivate that DeepTest "cannot accurately reflect the real-world driving scenes", for example snowy conditions. Instead of developing the neuron coverage metric like DeepTest and the other testing frameworks do, DeepRoad makes use of metamorphic testing. In the case of DeepTest, metamorphic testing is used in order to determine whether similar input images produce the same steering output signal.

The authors used road scenery images extracted from YouTube to check

how well their system works. DeepRoad focuses only on generating snow and rain conditions with UNIT, and the DeepRoad system is evaluated against three self-driving car Udacity challenge DNNs. DeepRoad, as every other system, detected thousands of erroneous behaviors in the Udacity data sets. DeepRoad also worked as a metric for inconsistency in DNNs, because it detected clear differences in the amount of inconsistent behaviors between the three Udacity DNNs. The weakness of DeepRoad is the already mentioned limitation of weather conditions, and that the framework is only tested on three DNNs.

	DeepXplore	DLFuzz	DeepTest	DeepRoad
Year of publishing	2015	2018	2017	2018
Sophistication	Satisfactory, an early draft with new concepts	Sufficient, develops further on DeepXplore ideas but does not delve deep enough	Good, addresses the shortcomings of the system appropriately	Very good, improves the image generation compared to DeepTest
Amount of application areas	Very good, not only for DNNs classifying images	Good, not limited to images but alternative areas not discussed in depth	Sufficient, focuses on very specific image alterations	Good, takes overall road scenery into account
Focuses on	Generating as different input images as possible for DNN systems	Fuzzing testing without the use of multiple DNNs, adversarial testing	Making affine transformations and using filters to make test images	Synthesizing road scenes with a GAN and metamorphic testing

Table 1. Comparison summary of the four chosen DNN testing frameworks. The grading from worst to best: Sufficient, Satisfactory, Good, Very good, Excellent.

4 Discussion

What is common for three of these testing frameworks, namely DeepXplore, DLFuzz and DeepTest, is that they developed the neuron coverage metric. DeepRoad uses metamorphic testing instead of neuron coverage. When several articles make the same kind of research, it is evident that the term neuron coverage (NC) should be standardized and not be pre-

sented as a new concept in every other article regarding the subject. Each of the automated frameworks managed to detect several erroneous behaviors, in the case of DeepXplore and DeepTest over a thousand of them. A problem with the NC metric is that if there are any oversights in the training phase of the DNNs, the frameworks might still give great neuron coverage even though the DNN itself might have been poorly designed.

What is different in these four frameworks is that they all focus on somewhat different areas: DeepXplore focuses on developing the term neuron coverage, and proving that it is a good metric to automatically measure how good the DNN test data is. DeepXplore also shows that a high neuron coverage leads to an increased number of found error-inducing inputs. DLFuzz focuses on adversarial aspects and how these imperceptible perturbations could be tested in an automated way. DeepTest is a more specific solution which could perhaps be used with modifications and extensions by self-driving car manufacturers. DeepRoad does not even develop neuron coverage as such, but instead focuses on the metamorphic testing of the generated images.

These testing frameworks could be used to test real-life applications in an early development stage. None of these testing frameworks have been used by car manufacturers using DL systems to our knowledge. An important thing to mention is that only looking at deep learning errors is not be enough for example to ensure that a self-driving vehicle will not crash. Like mentioned, in the first ever self-driving car fatal incident the problem was not related to the deep neural network, but to the sensory systems. The different self-driving car sensors also pose a security threat, as discussed in [12].

5 Related work

Apart from the aforementioned four deep learning testing frameworks (DeepXplore, DLFuzz, DeepTest and DeepRoad), there is lots of related work being done in subjects revolving around deep neural network security and classification problems. Below we present two closely related topics to the testing frameworks.

Deep neural network image classification problems. A widely discussed problem with DNNs is that they do not recognize images as humans. For example in [13], authors demonstrate how DNN models can be tricked by producing images which humans cannot recognize, but which

the DNNs are able to recognize. This problem is also present the other way around; in some cases a DNN might not be able to recognize something we humans easily do, for example facial recognition while a human is wearing accessories such as glasses [14]. These kind of problems are also present in self-driving car systems, which could be seen in the aforementioned Uber incident where, allegedly, the DNN could not classify a human as a recognizable object.

Attacks and defenses for deep neural networks. When speaking of DNNs, it is important to note that there has been lots of discussion about the image classification problem can pose a real security threat, because of the fact that you cannot monitor every decision a large DNN makes. Articles such as [15] claim that you can fool deep neural networks even by modifying only a single pixel of an image. Car manufacturers must use extensive security modules in order to prevent adversarial attacks such as these. For example [16] contains more general discussion about deep neural network vulnerabilities and defense mechanisms.

6 Conclusion

In this paper, we reviewed some of the most recent automated deep neural network testing frameworks. The main focus of this paper was to look at how DeepXplore is related to some more recently published frameworks. Compared to other work, DeepXplore has indeed a broader spectrum of application areas which goes beyond image-classifying DNNs, while the three other frameworks focus on image classification and automated testing. The frameworks are most probably being developed further as we conclude this seminar paper, so we can expect some more sophisticated solutions in the upcoming years.

All of the frameworks have unique testing solutions, but the end result is similar in each of the articles; by mutating the seed images, the DNNs showed thousands of wrong decisions. This is something which can have fatal consequences for example in self-driving car systems, where misclassification may affect the steering angle, braking and/or throttling of the car. Further work could be done in the field of self-driving car DNNs to test what erroneous inputs cause wrong throttling and braking decisions.

References

- [1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Praseon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, Karol Zieba. End to End Learning for Self-Driving Cars. <https://arxiv.org/pdf/1604.07316.pdf>, April 2016 (accessed January 30, 2019).
- [2] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS 2012 - Advances in Neural Information Processing Systems 25*, volume 26, December 2012.
- [3] Ruslan Salakhutdinov, Andriy Mnih, Geoffrey Hinton. Restricted Boltzmann Machines for Collaborative Filtering. In *Proceedings of the 24th international conference on Machine learning*, volume 24, June 2007.
- [4] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278 – 2324, November 1998.
- [5] The Tesla Team. A Tragic Loss. <https://www.tesla.com/blog/tragic-loss>, June 2016 (accessed April 2, 2019).
- [6] National Transportation Safety Board. Preliminary report- HWY18MH010. <https://www.nts.gov/investigations/AccidentReports/Reports/HWY18MH010-prelim.pdf>, June 2018 (accessed April 4, 2019).
- [7] Kexin Pei, Yinzhi Cao, Junfeng Yang, Suman Jana. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In *SOSP 2017 - Symposium on Operating Systems Principles*, volume 26, pages 1–18, October 2017.
- [8] Udacity. Udacity self-driving car challenges. <https://github.com/udacity/self-driving-car/tree/master/challenges/>, 2016 (accessed April 6, 2019).
- [9] Jianmin Guo, Yu Jiang, Yue Zhao, Quan Chen, Jianguang Sun. DLFuzz: differential fuzzing testing of deep learning systems. In *ESEC/FSE 2018 - ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, volume 26, pages 739–743, November 2018.
- [10] Yuchi Tian, Kexin Pei, Suman Jana, Baishakhi Ray. DeepTest: Automated Testing of Deep-Neural-Network-driven Autonomous Cars. In *International Conference on Software Engineering*, volume 40, May-June 2018.
- [11] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, Sarfraz Khurshid. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems. In *ACM/IEEE International Conference on Automated Software Engineering*, volume 33, pages 132–142, September 2018.
- [12] Alexander M. Wyglinski, Xinming Huang, Taskin Padir, Lifeng Lai, Thomas R. Eisenbarth, Krishna Venkatasubramanian. Security of Autonomous Systems Employing Embedded Computing and Sensors. 33:80–86, January 2013.

- [13] Anh Nguyen, Jason Yosinski, Jeff Clune. Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images. In *The IEEE Conference on Computer Vision and Pattern Recognition*, volume 28, pages 427–436, June 2015.
- [14] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, Michael K. Reiter. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. In *CCS '16 Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, volume 23, pages 1528–1540, October 2016.
- [15] Jiawei Su, Danilo Vasconcellos Vargas, Sakurai Kouichi. One Pixel Attack for Fooling Deep Neural Networks. In *IEEE Transactions on Evolutionary Computation*, 2019.
- [16] Nicolas Papernot ; Patrick McDaniel ; Somesh Jha ; Matt Fredrikson ; Z. Berkay Celik ; Ananthram Swami. The Limitations of Deep Learning in Adversarial Settings.

Mobile Edge Computing Assisted Internet of Things

Héctor Bállega Fernández

hector.ballegafernandez@aalto.fi

Tutor: Abbas Mehrabi

Abstract

With the fast spread of the Internet of Things and its multiple applications, e.g., smart cities, wearable devices or smart grids, new requirements including optimization of resources, ultra-low latency connections and high availability bandwidth must be faced to offer an optimal quality of service. To fully use the computing capabilities of smart devices and reduce the latency, the edge computing paradigm has been introduced to localize the computation / communication at the edges of the network and move the computationally-intensive tasks from the IoT devices to the closest edge servers. In this paper, we examine the transition from the Mobile Cloud Computing paradigm to the Mobile Edge Computing with special emphasis on the use cases of IoT applications where the Mobile Edge Computing is enabled in the future 5G networks. Finally, we analyze two application scenarios where MEC in combination with 5G will bring important benefits, the automotive IoT and the adaptive mobile video streaming.

KEYWORDS: *Mobile Cloud Computing, Mobile Edge Computing, IoT, 5G Networks, Mobile Edge Caching, Device-to-Device, Vehicle-to-Vehicle.*

1 Introduction

The network infrastructure that powers the Internet of Things (IoT) has moved to the cloud. Companies allocate their resources in different parts of the world; however, Recently, emerging technologies such as, IoT, Virtual Reality (VR) or Augmented Reality (AR) introduce the requirement of a very low latency. These latency sensitive applications have created a new paradigm: Mobile Edge Computing where the servers of the network are deployed as close as possible to the clients.

The Mobile Edge Computing paradigm brings data and computation power closer to the end users, reducing the latency and increasing the communication efficiency. This is achieved by deploying the resources in small data centers operated by the service providers [25].

This concept has brought new difficulties compared with the Mobile Cloud Computing paradigm, for example, the coordination over all the nodes of the system, and new challenges related with the virtualization of the network infrastructure.

This paper is organized as follows, Section 2 presents an overview of the Mobile Cloud Computing paradigm, Section 3 describes the Mobile Edge Computing and compares its advantages and disadvantages with Mobile Cloud Computing paradigm in the mobile context. Section 4 analyses the use cases and characteristics of the Internet of Things and its technical requirements for Mobile Edge Computing applications. Section 5 discusses the role of Mobile Edge Computing in the Internet of Things, presents the benefits of using 5G as the communication model and analyzes two application scenarios. Finally, Section 6 presents our conclusions.

2 Mobile Cloud Computing (MCC)

The popularity of smartphones has increased rapidly in a broad range of domains: communications, e-learning, gaming, health-care or entertainment are some of the areas boosted by the rapidly development of *Mobile Computing*. This term, refers to the platform supported by wireless communication that enables mobile users access to online services [10].

Mobile Cloud Computing (MCC) is the application of Cloud Computing to mobile devices, enabling on demand network access and storage capabilities which can be used to deliver different services to mobile users [7].

In MCC the device intensive computations, storage and data processing

have been transferred to a cloud-service, and therefore the mobile devices do not need a powerful hardware because the hard computing tasks can be processed in the Cloud. The Figure (Fig. 1) below illustrates the MCC architecture [15]. The mobile devices are connected to the network using a mobile network base such as, an access point, satellite or base transceiver station (BTS). The information is transmitted to a central processor unit which is connected to different servers that provides the network capabilities. Later, the requests are delivered to a cloud platform to be processed and brings the corresponding cloud services to the end users.

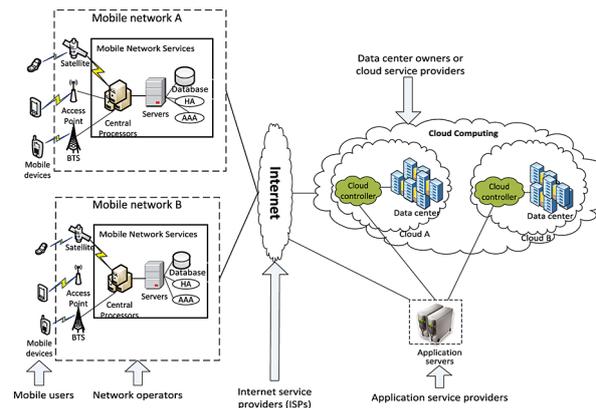


Figure 1. MCC Architecture [15]

The major challenges that MCC has to face are related with the ubiquitous nature of the mobile devices [17]. Rapid changes in the quality of wireless communication implies the need for highly adaptive applications, mobility implies address migration and location dependency, and portability implies limitations on the mobile device hardware characteristics.

These physical limitations are well solved by the possibility to scale the cloud infrastructure depending on the demand. Cloud service-providers, can easily expand a service without constraints because the hardware resources are virtualized and shared between a huge amount of customers [29].

3 Mobile Edge Computing (MEC)

According to the Cisco Visual Networking Index, in 2022 there will be 12.3 billion mobile connected devices, representing the 20 percent of the total Internet traffic [2].

The increasing trend in the number of mobile devices has constituted new scenarios that MCC is not able to solve. For example, when it is neces-

sary to ensure real-time communications or guarantee a high Quality of Service (QoS) [8].

To overcome this limitations, the Mobile Edge Computing (MEC) paradigm offers cloud computing resources such as, processing capabilities or storage within the Radio Access Network (RAN) [19]. MEC connects the client directly to the closest cloud service in the edge network, improves its computational capabilities by offloading the computation efforts and reduces the latency and network bottlenecks [27].

According to the European Telecommunications Standards Institute (ETSI) the service environment around the MEC is characterized by [19]:

1. Proximity

Being close to the source of information enables MEC to capture and analyze key information [13]. It is beneficial for resource-hungry applications, augmented reality or video analytics.

2. Ultra-low latency

The cloud services are deployed at the nearest location to the end user, segregating network data movements to the Core Network. Therefore, MEC is able to minimize congestion in other parts of the network and improve the overall quality of experience.

3. High bandwidth

Directly exposing the information to the edge network avoid unnecessary transmissions and bandwidth consumption of routing messages to the consumers via the Core Network [?].

4. Real-time access to radio network information

The applications that make use of real-time network information can estimate the congestion and bandwidth of the network to offer context-related and smarter services to their customers.

5. Location awareness

The local services can leverage low-level signals to discover the exact location of each connected device [6] [22].

There are multiple proposals to define an Edge Computing architecture [26], which indicates, that the concept of the edge network is not clearly defined and the terminology varies depending on the categories and functionalities to evaluate. In [24] the authors propose a categorization based on the common deployment features of the Edge Computing paradigm. The first category is *resource-rich servers deployed close to the end-device*,

the second one is based on exploiting resources from *heterogeneous nodes at the edge* and the last category is a *federation of resources at the edge and centralized data centers*. The MEC architecture belongs to the first category (*resource-rich servers deployed close to the end-device*) since its approach consists of deploying the cloud services within the Radio Access Network (RAN).

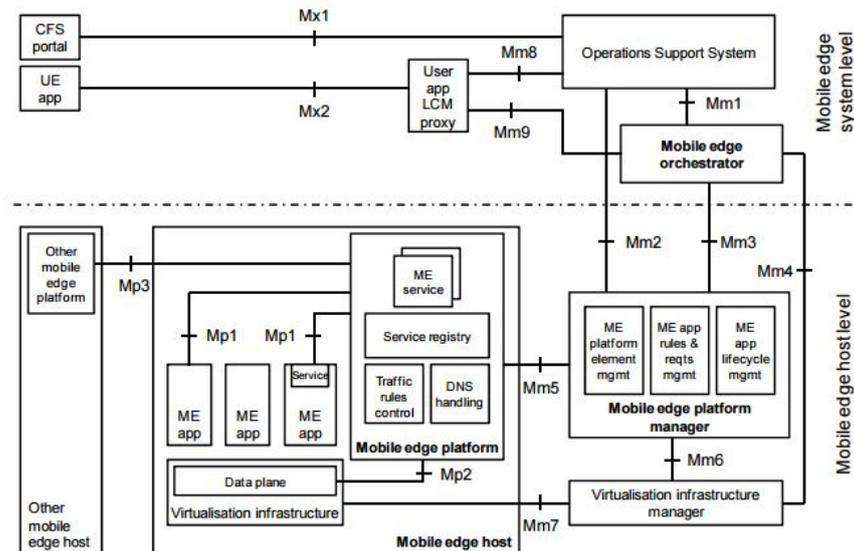


Figure 2. MEC Reference Architecture [5]

The figure above (Fig. 2), provides a wide view of the entities and functions that form the MEC reference architecture [5]. It focus on two different levels, the mobile edge host and the mobile edge system.

The **mobile edge host level** contains a mobile edge platform (responsible of enabling applications to initialize and consume their services) and a virtualisation infrastructure to provide compute, storage and network resources to run mobile edge applications.

The **mobile edge system level** contains a mobile edge orchestrator responsible of maintaining a general view of the mobile edge system based on the number of hosts, available resources and edge services. The orchestrator selects the most adequate mobile host to instantiate a mobile edge application based on constraints, e.g, number of connected devices, available memory or network latency.

The performance of MEC relies heavily on the capabilities of the service orchestrator as well as its interaction with the network architecture. The fluctuation of resources as well as the conditions of the network (due to the mobile capabilities of the devices) introduce new challenges to solve such

as, elastic resource allocation [18], service placement [12], edge selection and reliability [33].

3.1 MCC vs MEC

Under the MCC paradigm, the mobile devices exploits the benefits of a cloud, such as faster processing capabilities or a higher storage capacity, through internet access via a Core Network (CN). Although there are notorious advantages, e.g, enabling computational demanding applications to end-users or extending the battery life of the mobile devices, it also introduces an increment in the latency of the communications due to the remoteness of the cloud resources. Certain scenarios that involve the Internet of Things require an ultra-low latency (less than 1 ms) and high reliability. MEC promise to overcome this limitations by placing the cloud services in close proximity with the mobile devices [34].

	MCC	MEC
Network Infrastructure	Private/Public Cloud	Mobile Network Operator
Location	Anywhere	RAN
Deployment	Centralized	Distributed
Accessibility	via Internet	Direct to the Edge Network
Context Awareness	High	High
Latency	High	Low
Storage capabilities	High	Low
Computational power	High	Low

Table 1. Comparison of MCC and MEC models

A high level comparison of the most important aspects of the MCC and the MEC is outlined in Table I. In the MEC paradigm, the network infrastructure belongs to the mobile network operator, meanwhile in the MCC it might be part of a public or private cloud platform. This implies that in the MCC model the deployment is centralized around the cloud-service provider whereas in the MEC it is distributed. The location of the nodes must be within the radio network access in MEC, while in MCC the cloud resources can be geolocated in multiple locations. Because the accessibility in the MEC is established directly through the edge network and not via the Internet, the cloud resources are closer to the mobile devices, which immediately reduces the latency. In MCC, the clients can exploit computing and storage resources of a powerful centralized cloud

in contrast with the MEC, where the close proximity also limits the cloud capabilities. Finally, both models offer context awareness to discover the exact location of each mobile device and gather information from their environment.

4 Internet of Things (IoT)

The Internet of Things (IoT) is a communication paradigm where physical objects are able to interact with each other, share information and coordinate decisions to reach common goals. Essentially, IoT combines two concepts: *Internet*, that appeals to a network oriented vision, and the *Things*, that are the generic objects integrated in the ecosystem [11]. The IoT aims to provide connectivity to anything at any time and place, being a key technology in enabling Machine to Machine (M2M) communications. The IoT in combination with M2M brings the possibility of systems monitoring themselves and responding to changes in the environment with no human interaction.

In the literature [31][9], the IoT architecture is represented as a set of layers representing a group of modules with a cohesive set of services [28]. The layers from a bottom-up perspective are: The Device Layer, Network Layer, Session Layer, Cloud Layer, Application Layer, Management Layer, and Security Layer. It is possible to distributed each layer in different parts of the system, as well as dispense of those that do not fit the requirements of the design.

A further vision correlated with the IoT is the Web of Things [32], where web standards like HTTP or WebSockets are used to integrate into the web everyday-life objects that contain an embedded device or computer.

4.1 IoT Enabling Technologies

In this section we discuss the most relevant technologies that enhanced the IoT [30].

1. Identification

RFID systems are one of the key technologies used in the IoT. They are composed by at least one reader and several RFID tags. The tags have a unique identification in a small microchip attached to an antenna (used for receiving the external signal from the reader and transmitting the tag ID). The RFID tags are passive, which means

that they do not have any kind of power supply and the energy needed to transmit their ID is taken by induction from the query signal of the reader [21].

2. Sensor Networks

Sensor networks contribute to the increase in context awareness in the system by tracking the status of the objects, such as, the location, proximity, temperature or pressure. The sensor networks contain sensing nodes that report their measures to gateways called sinks. The main sensor network solutions are based on the IEEE 802.15.4 standard, which defines the physical and MAC layers for low-power, low bit rate communications in wireless personal area networks (WPAN) [1].

3. Wireless Communications

The IoT wireless communication technologies connect heterogeneous objects together to share information. Some examples of communication protocols involved in IoT are NFC, Bluetooth, WiFi, Z-wave and 4G LTE. Usually, the IoT nodes operate with low power through noisy communication channels.

4. Semantics

Semantics consists on the ability to extract knowledge, model information and analyze the obtained data to provide the right decision for the required service. The main technologies used to model semantics are the Resource Description Framework (RDF) and the Web Ontology Language (OWL). The recommended format to exchange data in IoT is the Efficient XML Interchange (EXI), which reduces the bandwidth needs without detriment of the energy consumed for processing or the required storage size [9].

5 Role of MEC in IoT

The MEC ecosystem not only provides benefits for the Mobile Network Operator, such as, a better Quality of Experience (QoE) or faster cloud-resource provisioning. But also to the IoT use cases that require an accurate context awareness, a low latency and high available throughput. According to the IDC global predictions [3], by 2020, 40% of IoT created data will be stored, processed, analyzed and consumed upon close or at

the edge of the network.

The ETSI-ISG group has identified the following technical requirements in MEC applications [4], the most relevant for an IoT scenario are:

A. Generic requirements

- The mobile edge system shall be possible to deploy the mobile edge platform on mobile edge hosts in various locations, including radio nodes, aggregation points, gateways, and in a distributed data centre at the edge of the core network.
- The mobile edge system shall be able to verify the authenticity and integrity of a mobile edge application.
- The mobile edge system shall be able to maintain the connectivity between the IoT device and an application instance when the device performs a handover to another cell associated with the same mobile edge host.

B. Service requirements

- The mobile edge platform shall allow authentication and authorization of providers and consumers of mobile edge services.
- The mobile edge system shall be able to route network traffic to other nodes of the system and influence the DNS resolution.
- The mobile edge system shall be able to support user identity mapping.
- The mobile edge system shall be able to support bandwidth management.
- The mobile edge system shall be able to provide radio network information.

C. Operation and management requirements

- It shall be possible to control the access of a mobile edge application to mobile edge services.
- The mobile edge system shall be able to provide a virtualization environment to run the applications.
- The mobile edge platform management shall be able to collect and expose performance data regarding the virtualisation environment.

D. Security, regulation and charging requirements

- The mobile edge platform shall only provide a mobile edge application with the information for which the application is authorized.
- The mobile edge system shall allow the collection of charging-related information, log it in a secure way and make it available for further processing.

5.1 MEC in 5G Networks

5G is the latest generation of cellular mobile communications. It succeeds the LTE-Advanced generation (4G) and it promises to offer a higher data rate, decrease the latency, save energy, reduce costs and handle a massive device connectivity. The 5G networks based on the 3GPP specification are a target environment for MEC deployments [23]. The Service Based Architecture (SBA) from the 5G specification leverages some characteristics that are also specified in MEC, such as network functions virtualisation (NFV) and Software-defined Networking (SDN) paradigms. Besides, the 3GPP 5G standard define key enablers for Edge Computing, allowing a MEC and 5G system interact together in traffic routing and policy control operations.

The ETSI-MEC standard proposes different scenarios to deploy MEC hosts [23]. The User Plane Function (UPF) plays an important role when MEC is deployed in a 5G network. UPFs can be seen as a distributed and configurable data plane from the MEC system perspective. It takes care of redirecting the traffic through the preferred applications in the network. The location of the UPF is a choice of the MNO based on technical or business decisions, such as, availability, running applications or estimated user load.

In a scenario with multiple IoT devices, some options to reduce the latency and increase the bandwidth at the edge include deploying the MEC collocated with a transmission node and a local UPF, or deploying the MEC and the local UPF with a network aggregation point that is able to collect and digest data from different sources.

5.2 IoT and MEC Application Scenarios

5.2.1 *Automotive IoT Applications: Internet of Vehicles (IoV)*

5G is a key enabler for the autonomous vehicles industry. It enhances the V2X (Vehicle to Everything) paradigm which covers other particular cases, e.g, Vehicle to Vehicle (V2V), Vehicle to Infrastructure (V2I) or Vehicle to Device (V2D). Automotive IoT requires a critical communication infrastructure where reliability and ultra-low latency are decisive factors [16]. The current network technologies are not able to provide a good solution to some features of a vehicular network, including real-time vehicles monitoring, continuous sensing or high reliability. The combination of MEC with 5G will bring improvements in the virtualization of the network infrastructure, that towards the placement of the resources within the RAN will handle the challenges related to real-time monitoring and decision, vehicles orchestration and scalability [14].

5.2.2 *Adaptive Mobile Video Streaming*

MEC together with the future 5G cellular networks will improve the experience of visualizing high quality videos in mobile devices. By allocating the video files at the edge of the network, MEC will reduce the delivery delay and traffic congestion of the network. The quality of the videos may be impaired when the clients are not fixed at one location and they compete simultaneously for shared bandwidth. In [20], the authors explore the impact of Collaborative Edge Caching and QoE-traffic optimization for MEC environments, proving that it is a useful method to alleviate the network traffic and increase the overall Quality of Experience.

6 Conclusion

This paper surveys and summarizes the Role of MEC in IoT. We have introduced the MCC paradigm, its architecture and challenges. Then, we have reviewed the MEC paradigm that overcomes the MCC limitations by bringing the cloud resources to the Edge of the network. We have reviewed the relevant technologies that enable the IoT and the key role that MEC plays on it. Finally, we have identified 5G as a key enabler for MEC and discussed two application scenarios that will benefit from its combination.

References

- [1] Ieee 802.15 working group for wireless personal area networks (wpans). <http://www.ieee802.org/15/>. (Accessed on 04/06/2019).
- [2] Cisco visual networking index: Global mobile data traffic forecast update, 2017–2022 white paper - cisco, 2017.
- [3] International data corporation (idc), futurescape: Worldwide iot 2019 predictions, 2019.
- [4] ETSI GS MEC 002. Mobile edge computing (mec) technical requirements, 2017.
- [5] ETSI GS MEC 003. Mobile edge computing (mec) framework and reference architecture, 2016.
- [6] ETSI GS MEC 013. Mobile edge computing (mec) location api, 2017.
- [7] Nurul Hidayah Ab Rahman, Niken Dwi Wahyu Cahyani, and Kim-Kwang Raymond Choo. Cloud incident handling and forensic-by-design: cloud storage as a case study. *Concurrency and Computation: Practice and Experience*, 29(14):e3868. e3868 CPE-16-0076.R1.
- [8] Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. Mobile edge computing: A survey. *IEEE Internet of Things Journal*, 5:450–465, 2018.
- [9] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials*, 17(4):2347–2376, Fourthquarter 2015.
- [10] Priyanka Asrani. Mobile cloud computing. *International Journal of Engineering and Advanced Technology*, 2(4):606–609, 2013.
- [11] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Comput. Netw.*, 54(15):2787–2805, October 2010.
- [12] M. Bagaa, T. Taleb, and A. Ksentini. Service-aware network function placement for efficient traffic handling in carrier cloud. In *2014 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 2402–2407, April 2014.
- [13] X. Chen, W. Li, S. Lu, Z. Zhou, and X. Fu. Efficient resource allocation for on-demand mobile-edge cloud computing. *IEEE Transactions on Vehicular Technology*, 67(9):8769–8780, Sep. 2018.
- [14] S. K. Datta, J. Haerri, C. Bonnet, and R. Ferreira Da Costa. Vehicles as connected resources: Opportunities and challenges for the future. *IEEE Vehicular Technology Magazine*, 12(2):26–35, June 2017.
- [15] Hoang T. Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless Communications and Mobile Computing*, 13(18):1587–1611.
- [16] Zakaria et al. Internet of things (iot) automotive device, system, and method - google patents. <https://patents.google.com/patent/US9717012B2/pt>, 2017.

- [17] George H. Forman and John Zahorjan. The challenges of mobile computing. *Computer*, 27(4):38–47, April 1994.
- [18] Y. Liu, M. J. Lee, and Y. Zheng. Adaptive multi-resource allocation for cloudlet-based mobile cloud computing system. *IEEE Transactions on Mobile Computing*, 15(10):2398–2410, Oct 2016.
- [19] C. Chan N. Sprecher S. Abeta A. Neal et al M. Patel, B. Naughton. Mobile-edge computing introductory technical white paper, 2014.
- [20] A. Mehrabi, M. Siekkinen, and A. Ylä-Jääski. Qoe-traffic optimization through collaborative edge caching in adaptive mobile video streaming. *IEEE Access*, 6:52261–52276, 2018.
- [21] Ahmed Mahmoud Mostafa. Iot architecture and protocols in 5g environment. In *Powering the Internet of Things With 5G Networks*, pages 105–130. IGI Global, 2018.
- [22] Monica Paolini. Mec and edge computing. the importance of location, 2016.
- [23] ETSI GS MEC White paper No. 28. Mec in 5g networks, 2018.
- [24] G. Premsankar, M. Di Francesco, and T. Taleb. Edge computing for the internet of things: A case study. *IEEE Internet of Things Journal*, 5(2):1275–1284, April 2018.
- [25] Mario; Taleb Tarik Premsankar, Gopika; Di Francesco. Edge computing for the internet of things. 2018.
- [26] Dario Sabella, Alessandro Vaillant, Pekka Kuure, Uwe Rauschenbach, and Fabio Giust. Mobile-edge computing architecture: The role of mec in the internet of things. *IEEE Consumer Electronics Magazine*, 5:84–91, 10 2016.
- [27] Dimas Satria, Daihee Park, and Minho Jo. Recovery for overloaded mobile edge computing. *Future Generation Computer Systems*, 70:138 – 147, 2017.
- [28] Bedir Tekinerdogan and Ömer Köksal. Pattern based integration of internet of things systems. In Dimitrios Georgakopoulos and Liang-Jie Zhang, editors, *Internet of Things – ICIOT 2018*, pages 19–33, Cham, 2018. Springer International Publishing.
- [29] K. Tsakalozos, H. Kllapi, E. Sitaridi, M. Roussopoulos, D. Paparas, and A. Delis. Flexible use of cloud resources through profit maximization and price discrimination. In *2011 IEEE 27th International Conference on Data Engineering*, pages 75–86, April 2011.
- [30] Andrew Whitmore, Anurag Agarwal, and Li Da Xu. The internet of things—a survey of topics and trends. *Information Systems Frontiers*, 17(2):261–274, Apr 2015.
- [31] L. D. Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, Nov 2014.
- [32] Deze Zeng, Song Guo, and Zixue Cheng. The web of things: A survey (invited paper). *JCM*, 6:424–438, 2011.
- [33] H. Zhang, F. Guo, H. Ji, and C. Zhu. Combinational auction-based service provider selection in mobile edge computing networks. *IEEE Access*, 5:13455–13464, 2017.

- [34] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong. An air-ground integration approach for mobile edge computing in iot. *IEEE Communications Magazine*, 56(8):40–47, August 2018.

Mobile Crowdsensing in Internet of Things

Arnab Rahman Chowdhury

arnab.chowdhury@aalto.fi

Tutor: Abbas Mehrabidavoodabadi

Abstract

The 'Internet of Things', a buzzword in the modern era of Information Technology. It is a paradigm where every object has the capability of classifying, sensing, communicating and computing processes and activities that enable them to communicate and exchange valuable information with each other as well as provide services to achieve objectives. IoT consists of small, low-powered, sophisticated, portable devices which have limited life-span and computational power and are scattered within a large deployment area. Efficient data collection from these IoT devices is a challenging task. In the advancement of Information and Communication Technology, different emerging techniques, including Mobile Cloud Computing (MCC) and Mobile Edge Computing (MEC) are proposed by researchers to mitigate the challenges. This survey paper focuses on different Mobile Crowdsensing (MCS) strategies for data collection and also examines the Information and Communication Technology, such as MCC, MEC which are deliberately assisting MCS to be a trustworthy medium of data collection and processing. The goal of the survey is not only confined to the analysis of MCS but also explore the necessity of different types of data collection, data collection process and the impact of ICT technologies on MCS.

KEYWORDS: *Internet of Things (IoT), Mobile Crowdsensing (MCS), Mobile Cloud Computing (MCC), Mobile Edge Computing (MEC).*

1 Introduction

The 'Internet of Things' where mobile and portable objects of everyday life are embedded with sensors for electronic communication and applicable protocol suits for reliable communication between the objects, thus making it an integral part of the Internet [1]. Every IoT object is equipped with capabilities of identifying, sensing, networking and processing . In foreseeable future, appropriate Information and Communication Technology will connect all the digital and physical entities with each other, which fosters the future of application and services.

IoT consists of sophisticated, low-powered, portable and wearable devices which have limitations, including shorter life-span and limited communication power. Moreover, devices are scattered within a large deployment area, which poses a great challenge for efficient data collection. In addition, the challenge becomes more critical because of the abandoned and inhospitable deployment environment, unreliable and critical radio frequency links, frequent changes of network topologies and heterogeneity of sensor nodes as well as multi-hop communication. Collecting time-varying, time-sensitive, heterogeneous, ubiquitous data has made the whole process more complicated and challenging. The advanced Information and Communication Technology researchers propose different emerging techniques and methods, including Mobile Cloud Computing (MCC) and Mobile Edge Computing (MEC) for data collection and analysis to alleviate the challenges.

In recent years, Mobile Crowdsensing (MCS) has acquired significant attention as a popular sensing approach. MCS is a technique where users of sensing and computing capable IoT devices collectively sense and share data using their devices as well as perform computation based on the sensed data for extracting information of common interest. Mobile devices, such as smart cellular phones, wearable and portable devices have powerful computation as well as communication capabilities and are enriched with multi-functional sensors, for example Proximity, Accelerometers, Gyroscope, Air humidity and Temperature. These sensors are accountable for data collection and transmission to the central processing server. As a result, these sensing devices consume many resources, such as frequencies, power and storage as well as the service quality of the ap-

applications degrade due to extensive amount of data generation. Moreover, they have low transmission power and a narrow communication range.

This paper surveys data collection process in different application domain, different MCS data collection strategies, their limitations and how researchers overcome these challenges by initiating cost as well as energy effective and self-participatory data collection techniques.

The rest of the paper is organized as follows: Section 2 describes data classification and collection techniques in different application domain, Section 3 presents emergence of MCS techniques for collecting data and the challenges of MCS approaches, Section 4 focuses on advanced ICT approaches such as Mobile Edge Computing to mitigate the MCS challenges and finally Section 5 discusses future direction and conclusion.

2 Data classification and collection techniques in different application domain

In the advancement of Information and Communication Technology, everyday physical objects are equipped with miniature sensors and processors. These sensor enable connected objects work collaboratively to collect captured data which makes human live easy and comfortable. This interweave environment is referred to as Smart Environment. The definition of Smart Environment originates from the ubiquitous computing. According to Mark Weiser, every object in our everyday lives are associated with invisible and a large number of sensors, actuators which is connected widely through a continuous network [2]. IoT can integrate with the smart environments based on different application domain and industry. Adoption of IoT in industry is known as Industrial IOT (IIOT). The data that are generated from different application domain and industry vary in terms of data uses and collection techniques.

2.1 Smart Healthcare

Current healthcare system utilizes modern Information and Communication Technologies for providing healthcare facilities which is referred to as e-health. This term encompasses a range of services and facilities, including electronic health record, emergency clinical support, telemedicine and personalised devices for diagnosis. These services are implemented with the help of rapid advancement of sensing technologies, exponential

growth of commercial wearable devices and promising machine learning techniques and cutting-edge artificial intelligent applications. Different low-cost and discreet inertial sensors, including Accelerometers, Gyroscopes, Biometric pressure, Magnetic field, location aware sensors, such as Global Positioning System (GPS), and physiological sensors, for example temperature, image sensor, for instance SenseCam as well as ambient sensors (e.g. Thermometer, Hygrometer, Infra-red, Active RFID tags, NFC tags) are utilized for measuring daily physical activities, blood sugar level, temperature, respiration rate, blood pressure, health data, including weight, user-context information [3]. Moreover, many wearable and mobile devices, such as Fitbit, smart watches, mobile camera are used to log the daily activities to keep a regular person healthy and fit.

2.2 Smart Home

Our homes are equipped with electronic devices, including air-condition, television, thermostat, refrigerator, lightweight entertainment devices. IoT-based smart home allows to control all the electronic devices remotely and increases the standard of human's comfort at home. To ensure a sustainable environment, some of the sensitive data, such as temperature, humidity, luminosity, air quality are collected and monitored [4]. Smart home appliances, including home assistant, such as Google Assistant, Amazon Echo, security assistant Amazon Cloud Cam, smart smoke detector, such as Nest Protect are used for home automation and increase the living standard of dwellers. Collected data are transmitted using communication technology, including Bluetooth, ZigBee, Wi-Fi.

2.3 Smart Transportation

In the advancement of communication technology and vehicular network, automobiles are becoming smarter with enriched sensors embedded in them. Smart transportation incorporates some of the key technologies, such as identification and tracking, communication and networking, service management for rapid development in transportation system [5]. Sensors are responsible for collecting perceptive information of the environment, machine learning and artificial intelligent algorithms are making decision based on the perceived surroundings and control system execute each decision timely. There are two categories of sensors, namely Proprioceptive sensors which collect information of internal state of the vehicle

and Exteroceptive sensors which are responsible for observing outside environment of the vehicle. Smart sensors that are embedded in automobiles are mostly cameras, radar, lidar, infrared, ultrasonic. Ultrasound sensors are used in parking assistant, traffic sign recognition. Lane departure warning are controlled by smart camera, lidar is responsible for emergency braking, pedestrian detection, collision avoidance. Short range radar sensors are useful for collision warning, traffic alert, and adaptive cruise is controlled by long range radar [6]. VANETs (Vehicular Ad hoc Networks) is proposed for exchanging messages via vehicle-to-vehicle communication protocol, and vehicle to roadside access point [7].

2.4 Smart Grid

In modern age, energy management is one of the challenging task due to rising energy cost, fragile energy supplies, regulated emission and climate change. Smart sensor network gives immense opportunities for smart grid applications, including energy management, power control and monitoring, de-centralized storage facilities and renewable energy production [8]. Smart sensor based applications have some benefits, such as lower cost and easy deployment. These applications can be integrated in smart home environment for future activity prediction. Integration of IoT with the smart power grid improves the quality of service (QoS) of power management by continuously monitoring transmission lines and sending periodic report to the control unit. Integration of smart meter improves the Demand-Side Energy Management which reduces electricity bill and operational cost of the power grid based on the usage of the consumer [9].

3 Mobile Crowdsensing

Data acquisition techniques can be categorized in three defined approaches, such as conventional approach where data are collected by the IoT devices and transmitted to cloud infrastructures through base stations, MCS approach where users compute and contribute data and personalize information for common interest and emerging ICT based solution, for example edge computing for low-latency and high-reliable data collection. This survey paper mainly focuses on Mobile Crowdsensing (MCS) data collection approach but also discusses the benefits of using the emerging ICT technologies for implementing MCS.

In the context of ubiquitous computing, Mobile Crowdsensing is a technique where data acquisition and collective data sharing is done by sensing and computing capable mobile and locomotive devices (e.g. smartphones, tablet computers, wearables, smart vehicles) of individuals for extracting information to measure, estimate, predict, mapping processed data of common interest. The term Mobile Crowdsensing was first introduced by Ganti et. al. in the year of 2011 [10].

3.1 Data Generation Platform

User participation for collecting data is one of the key features of MCS. The authors in [11] have mentioned two platforms which are used as the source of user participant data.

3.1.1 Mobile Sensing

Sensor rich smart devices of individual participate in context-based sensing task and contribute their collected and computed information for further transformation.

3.1.2 Mobile Social Networking

Social platforms act as a source of sensor data where group of individual with common interest cooperate and share sensed data for greater interest. This sensing and collection strategy is also referred to as cluster/community sensing.

3.2 MCS Applications

MCS applications can be categorized in three major area of interest, such as environmental, infrastructure, and social.

3.2.1 Environmental MCS

In environmental MCS applications, the natural environmental factors are considered. For example, measuring the growth of world's population, uses of water and pollution in water substance, air and climate conditions, waste management techniques, earth core resource mining, monitoring wildlife habitat in forest and desert environment are the main concern for mapping comprehensive environmental circumstance by ensuring the participation of the active contributor.

3.2.2 Infrastructure MCS

Infrastructure applications involve the basic public infrastructures which are accountable for well-living condition in the society. Some of the major infrastructures are transportation, water resource management, energy, information for cheapest and easy communication, critical institutions and government sectors. MCS applications can be involved in measuring the large-scale infrastructural phenomena, such as water blockage, weather condition, traffic congestion in highway, nearby parking facilities and tracking products [10].

3.2.3 Social MCS

Social MCS applications relate with the individual's sharing their experience, activities, responsibilities to community for collective development of social well-being. Individual can share places they have visited, food they have tested, public resources they have used, social awareness against adultery activities and information about services they have taken from socio-cultural organisations and government which will foster the living standard of human being.

3.3 MCS Architecture

MCS architecture depends on the application domain. The general architecture of MCS in all application domain consists of five fundamental levels. The service of MCS reaches to the user in the form of application which goes through the process of data sensing and processing. Figure 1 illustrates the overall architecture model of MCS.

3.4 MCS Techniques for Data Collection

In the context of user participation and involvement, MCS is classified into two categories, such as Participatory and Opportunistic Crowdsensing. Participatory Crowdsensing refers to the group of enthusiastic people contributing information using their sensing and computing capable handhold devices whereas Opportunistic Crowdsensing collects information automatically without the intervention of the participants. The main Participatory and Opportunistic Crowdsensing data collection techniques are discussed in the following subsections.

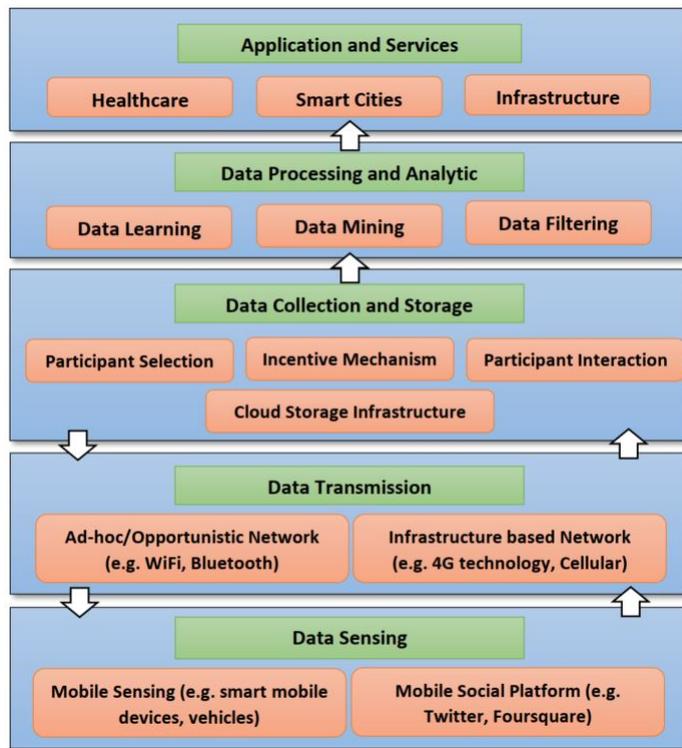


Figure 1. MCS Architecture

3.4.1 User Recruitment and Task Allocation Techniques in MCS

MCS techniques is mainly focused on users and selecting appropriate and well-suited participants from a massive pool of users for data collection and performing specific task is one of the challenging work. Fiandrino et. al. [12] have proposed a novel policy for user recruitment based on user sociability which refers to the estimation of user participation in sensing and contributing willingly and the spatial distance between participating users and tasks. The first study on dynamic user recruitment is proposed in [13] with diverse sensing tasks and handles the problem by proposing three greedy algorithms and illustrates that less number of participants can achieve stable task coverage. In user participation in MCS, contributing information should be kept secret from other users or to the platform itself. To figure out this problem, Xiao et. al. [14] have proposed a Basic User Recruitment protocol that recruits minimum participant while ensuring sensing quality of each task will always remain above a given threshold. They have also proposed a secret sharing scheme called ‘Secure User Recruitment’.

3.4.2 Incentive Mechanism Techniques for MCS

For collecting extensive amount of data from user, incentive mechanism will encourage the user for participating in data collection process. The

authors in [15] have proposed an incentive mechanism based on quality-driven auction where the participants will be paid based on the quality of sensed data instead of working hour. Zhang et. al. [16] have categorised the incentive mechanism into three border domains, such as entertainment where crowdsensing task will be represented as playable games and user will contribute by playing games; service where mutual-benefit is considered as service and to get benefit from the service, user needs to contribute to the system; monetary where participants will be paid for performing and contributing to the system. Monetary incentive technique is helpful to increase the number of potential participants and encourage them as well but the quality of sensing data may not meet the desired expectation. To address the issue, ‘TaskMe’ [17], a location-based social network model is proposed for participant selection. A multi-payment-enhanced reverse action scheme is proposed to improve the quality of sensing.

3.4.3 Social Sensing and Participant Interaction techniques in MCS

In the emergence of social platform, social relationships, responsibilities and activities can be inferred by the process of social sensing which is composed of social sensors (e.g. person who participate in sharing information about traffic situation, he/she is considered as traffic sensors) and social sensor receiver platform (e.g. social media). Rahman et. al. [18] have proposed social crowdsensing approach by forming an ad-hoc social network (authors consider one of the holy event of Muslim Ummah ‘Hajj’ as ad-hoc social network) and providing personalized as well as context aware services such as location information about point of interest, health care facilities, traffic and weather conditions, religious rituals and formalities. In [19], authors introduce community-based task where a group of user participate with high mutual attendance and sensing amplification mechanism to encourage near proximity devices to participate in resource sharing. Involving suitable participants to form groups or clusters is an important part of crowdsensing. To reduce delay for data collection and foster message delivery performance, a method is proposed in [20] for group/cluster formation based on the radio connectivity between the users. Guo et. al. [21] have proposed a group formation and recommendation technique, ‘GroupMe’, based on large scale mobile sensing data and social graphs mining such as group extraction and group abstraction in real world environment.

3.5 Challenges of MCS Approach

Implementation of MCS application requires individual or community participation, data aggregation and mining techniques, fast data transmission network infrastructure, remote data storage facilities and proper services of context-aware application. To assemble all the required components in one place, researchers have encountered with numerous challenges. These challenges are catalogued as follows.

3.5.1 Human participation and sensing

Participants of MCS application use their sensor enriched portable devices, such as smartphones and automobiles in data collection as well as sharing which consumes resources (e.g. battery power) and risks security as well as privacy threats. Encouraging users to participate is a challenging task.

3.5.2 Redundant and low quality data

Collected data from the anonymous participants can be faked, redundant, incorrect, inconsistent and abnormal. Moreover, sensed data can differ due to different sensing condition even if the data is collected using same sensor of same event. To extract necessary data from the poll of extraneous data, researchers introduce data filtering and selection techniques which imposes a great challenge in Mobile Crowdsensing.

3.5.3 Data accuracy and completeness

Distinguishing between false and accurate data is a challenging task. MCS is a continuous process and the completeness of data flow depends on successful accomplishment of all the stages involved. Moreover, MCS data are collected from both offline and online environment. Combining two forms of environments under one umbrella and facilitating heterogeneous cross space data mining is an important challenging issue.

3.5.4 Security and Privacy risk

Participant's in Mobile Crowdsensing shares sensitive, personal information which imposes security and privacy threats in data collection process. Some of the threats and security vulnerabilities of MCS are fake sensing attacks, malware, spoofing attacks, denial of service attacks and jamming [22]. Anonymization and cryptographic approaches are used for unveiling the identity of the participants but still faces challenges as cryptographic approach adds extra overhead to the sensing process.

3.5.5 Data authentication, reliability and integrity

Identity authentication, reliable data communication and data integrity ensures MCS system more secure and robust. To prevent security threats, validating every sensed data with proper user authentication is a challenging task. Some research articles suggest anonymous identity authentication mechanism based on pseudonym. Maintaining integrity of sensed data must be addressed to ensure security of the system. According to some research works, data reliability issue can be handled by identifying originated location of sensed data.

4 Edge-enabled MCS Approach

Traditional MCS is implemented based on centralized and cloud infrastructure. This architectural approach is inefficient and consumes excessive resources due to many concurrent connection from the users to back-end server for real-time data processing and storage. Moreover, maintaining additional device management increases the resource consumption. In [23], authors have performed a comparative study on Edge-enabled MCS and Cloud-based MCS. Authors have shown the tradeoff between centralized MCS and edge-assistant MCS in terms of computational offloading, cost, latency and privacy issue. A future MCS marketplace is proposed where participant can contribute and initiate sensing task. This survey summarizes MCS ecosystem enabled by Mobile Edge Computing in two network domain: 1. Mobile Networks, 2. Vehicular Networks.

4.1 Mobile Networks

4.1.1 Participant Selection and Quality of Sensed Data

Legitimate participant selection and maintaining the quality of sensed data are major challenges in typical centralized MCS architecture because of mobility of user and duplication of sensed data. To overcome the challenges, Jianbing Ni et. al. [24] have proposed Fog-assisted MCS framework which uses fog nodes to perform participant selection observing the mobility pattern of the user. Authors have also designed a duplication sensitive application named Fog-assisted Secure Data Duplication scheme to protect against probable attacks such as brute-force, duplicate-linking and duplicate-replay. Fiandrino et. al. [25] have proposed a fog-computing based user recruitment policy named DSE considering three

criteria, such as Distance between users and tasks, Sociability of the user and Energy consumption of sensor devices.

4.2 Vehicular Networks

4.2.1 Real-time Traffic Management

In recent times, smart vehicles are equipped with sensors to establish vehicle-to-vehicle communication which is known as Social Internet of Vehicles (SIoV) and according to the statistics, 150 millions smart vehicles will be connected by 2020. Due to limited wireless bandwidth, mobility of the vehicles, deployment cost of roadside units (RSUs) and expensive communication as well as computation capabilities of mobile cloud computing, real-time traffic management is facing a great challenge. Wang et. al. have designed a real-time traffic management framework CREAM based on crowdsensing in heterogeneous Social Internet of Vehicles [26]. Hou et. al. have considered the communication and computation capacity problem in vehicular network and proposed a novel system which utilizes mobile and parking vehicles as infrastructure [27]. A three-layer Vehicle Fog Computing (VFC) model (Vehicle-Cloudlet-Cloud) is proposed by [28] investigating offloading scheme in VFC for real-time traffic management.

5 Future Direction and Conclusion

This survey presents Mobile Crowdsensing (MCS) as a medium of data acquisition, it's challenges and the impact of ICT techniques on MCS. Different data collection techniques in terms of different application domains such as healthcare, transportation, industry are also discussed throughout the paper. MCS exploits the advantage of mobility of the participants to collect data from abandoned and inhospitable environment where static sensor networks face excessive difficulties.

As Mobile Crowdsensing is an emerging research area, many opportunities are needed to be addressed. In recent times, researchers are focusing on facilitating data collection and contributing in both Infrastructure based and Opportunistic networks known as Hybrid network. Data aggregation from online and offline community platform (cross-community platform) is becoming one of the challenging issues and future opportunities for MCS.

References

- [1] Luigi Atzori et al. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.
- [2] Mark Weiser et al. The origins of ubiquitous computing research at parc in the late 1980s. *IBM systems journal*, 38(4):693–696, 1999.
- [3] Jun Qi et al. Advanced internet of things for personalised healthcare systems: A survey. *Pervasive and Mobile Computing*, 41:132–149, 2017.
- [4] Warren Fincher et al. Standards of human comfort: relative and absolute. 2009.
- [5] Li Da Xu et al. Internet of things in industries: A survey. *IEEE Transactions on industrial informatics*, 10(4):2233–2243, 2014.
- [6] Sensors give smart cars a sixth sense, <https://www.futurecar.com/1516/sensors-give-smart-cars-a-sixth-sense>.
- [7] Hannes Hartenstein et al. *VANET: vehicular applications and inter-networking technologies*, volume 1. John Wiley & Sons, 2009.
- [8] Manar Jaradat et al. The internet of energy: smart sensor networks and big data management for smart grid. *Procedia Computer Science*, 56:592–597, 2015.
- [9] Iordanis Koutsopoulos et al. Control and optimization meet the smart power grid: Scheduling of power demands for optimal energy management. In *Proceedings of the 2nd International Conference on Energy-efficient Computing and Networking*, pages 41–50. ACM, 2011.
- [10] Raghu K Ganti et al. Mobile crowdsensing: current state and future challenges. *IEEE Communications Magazine*, 49(11):32–39, 2011.
- [11] Bin Guo et al. From participatory sensing to mobile crowd sensing. In *2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS)*, pages 593–598. IEEE, 2014.
- [12] Claudio Fiandrino et al. Sociability-driven user recruitment in mobile crowdsensing internet of things platforms. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2016.
- [13] Hanshang Li et al. Dynamic participant recruitment of mobile crowd sensing for heterogeneous sensing tasks. In *2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems*, pages 136–144. IEEE, 2015.
- [14] Mingjun Xiao et al. Secret-sharing-based secure user recruitment protocol for mobile crowdsensing. In *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pages 1–9. IEEE, 2017.
- [15] Yutian Wen et al. Quality-driven auction-based incentive mechanism for mobile crowd sensing. *IEEE Transactions on Vehicular Technology*, 64(9):4203–4214, 2015.
- [16] Xinglin Zhang et al. Incentives for mobile crowd sensing: A survey. *IEEE Communications Surveys & Tutorials*, 18(1):54–67, 2016.

- [17] Bin Guo et al. Taskme: Toward a dynamic and quality-enhanced incentive mechanism for mobile crowd sensing. *International Journal of Human-Computer Studies*, 102:14–26, 2017.
- [18] Md Abdur Rahman et al. A location-based mobile crowdsensing framework supporting a massive ad hoc social network environment. *IEEE Communications Magazine*, 55(3):76–85, 2017.
- [19] Stefano Chessa et al. Empowering mobile crowdsensing through social and ad hoc networking. *IEEE Communications Magazine*, 54(7):108–114, 2016.
- [20] Takamasa Higuchi et al. A neighbor collaboration mechanism for mobile crowd sensing in opportunistic networks. In *2014 IEEE International Conference on Communications (ICC)*, pages 42–47. IEEE, 2014.
- [21] Bin Guo et al. Groupme: Supporting group formation with mobile sensing and social graph mining. In *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 200–211. Springer, 2012.
- [22] Djallel Eddine Boubiche et al. Mobile crowd sensing—taxonomy, applications, challenges, and solutions. *Computers in Human Behavior*, 2018.
- [23] Martina Marjanović et al. Edge computing architecture for mobile crowd-sensing. *IEEE Access*, 6:10662–10674, 2018.
- [24] Jianbing Ni et al. Providing task allocation and secure deduplication for mobile crowdsensing via fog computing. *IEEE Transactions on Dependable and Secure Computing*, 2018.
- [25] Claudio Fiandrino et al. Sociability-driven framework for data acquisition in mobile crowdsensing over fog computing platforms for smart cities. *IEEE Transactions on Sustainable Computing*, 2(4):345–358, 2017.
- [26] Xiaojie Wang et al. A city-wide real-time traffic management system: Enabling crowdsensing in social internet of vehicles. *IEEE Communications Magazine*, 56(9):19–25, 2018.
- [27] Xueshi Hou et al. Vehicular fog computing: A viewpoint of vehicles as the infrastructures. *IEEE Transactions on Vehicular Technology*, 65(6):3860–3873, 2016.
- [28] Zhaolong Ning et al. Vehicular fog computing: Enabling real-time traffic management for smart cities. *IEEE Wireless Communications*, 26(1):87–93, 2019.

Formal Verification using Tamarin Prover

Eric Cornelissen

eric.cornelissen@aalto.fi

Tutor: Aleksi Peltonen

Abstract

The use of formal verification systems in the analysis of cryptographic protocols has become increasingly common in recent years. Such systems provide a robust way to guarantee security properties of a protocol. In a world where cryptographic protocols have become increasingly commonplace, e.g. for digital communication, their security also becomes increasingly important. In this paper the formal verification system TAMARIN PROVER is examined through the modelling and analysis of the NAXOS protocol. This results in an overview of TAMARIN's syntax and capabilities.

KEYWORDS: Tamarin Prover, Formal Verification, Cryptographic Protocols

1 Introduction

A cryptographic protocol guarantees security properties, e.g. end-to-end encryption, by applying cryptographic methods. Such protocols are used increasingly in day-to-day activities such as electronic communication, remote authentication, electronic payments, and signing digital contracts. These protocols need to be secure against attacks by third-parties or even

participating parties. Verifying security properties can be accomplished manually, by means of rigorous mathematical analysis, but it is becoming increasingly common to use formal verification systems as well [6].

Once a cryptographic protocol is designed and published, the security community will try to break it. In response to the vulnerabilities found by the community, the protocol is patched and the update is published. This vicious cycle continues until there is sufficient trust from the security community that the protocol is secure, at which point it is standardised and may be used in practice.

Analysts utilise formal verification systems in an attempt to limit the number of vulnerabilities in a protocol that attackers can exploit. Such a system does this by proving or disproving the correctness of statements about the protocol model. The model is written by a modeller in a formal syntax and includes details such as protocol definitions and desired security properties.

Manual verification of cryptographic protocols is perilous as humans are bound to make mistakes. However, formal verification is not the be-all-end-all solution to guaranteeing security in protocols for two reasons. Firstly because the model can be incomplete or contain mistakes and secondly because protocol analysis is an undecidable problem [17]. Hence, combining manual and formal verification of cryptographic protocols is a good practice to achieve the highest level of security.

This paper provides a model and analysis of the NAXOS protocol using the formal verification system TAMARIN PROVER [16] originally created by Schmidt et al. [19] in 2012. TAMARIN has been used to verify the security of a number of protocols, most notably TLS 1.3 [8, 9].

This paper is organised as follows. Section 2 provides background information on formal verification systems and TAMARIN. Section 3 uses the background information to present a model of the NAXOS protocol in TAMARIN, as well as the analysis of a security property. Finally, Section 4 offers concluding remarks.

2 Background

TAMARIN PROVER was originally presented with an analysis of Diffie-Hellman protocols by Schmidt et al. [19] in 2012 and has been improved over time. For example, support for convergent equational theories [11] and exclusive OR [12] were added in 2017 and 2018 respectively. Since

the analyses of Diffie-Hellman protocols, TAMARIN has also been used to analyse, among others, TLS 1.3 [8, 9], 5G authentication [2], Attack Resilient Public-Key Infrastructure (APRKI) [1], and the voting protocol Alethea [3].

As a formal verification system, TAMARIN aims to analyse and proof the security of cryptographic protocols. The model of the protocol, as well as the security properties, such as *the secret key is unknown to the adversary at the end of the protocol*, are defined by a human modeller.

Although the problem of verifying security properties in general is undecidable [17], TAMARIN will halt in most analyses [19]. TAMARIN typically halts in a matter of seconds for simple protocols, but may take several hours or days for complex protocols [14]. If it halts, it provides either a proof of correctness or a counterexample, i.e. an attack, on the security properties [16].

If TAMARIN does not halt, there are two potential causes. The first cause is that there are partial deconstructions, meaning that TAMARIN cannot resolve where a fact came from. The second cause is that the protocol, or the specific model of the protocol, is too complex. In both scenarios, the interactive mode offered by TAMARIN can be utilised to manually guide the proving process and potentially terminate the proof.

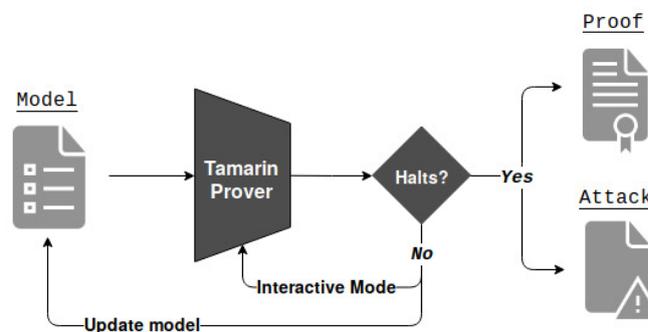


Figure 1. TAMARIN PROVER modelling workflow.

2.1 Related work

There exist a few alternatives to TAMARIN PROVER in the literature, among which are SCYTHÉ (the predecessor of TAMARIN) [10], PROVERIF [4], CRYPTOVERIF [5], and MAUDE-NPA [13]. TAMARIN distinguishes itself with visualisation capabilities, providing both attack and proof visualisation, as well as an interactive mode where analysts can inspect the generated proof step-by-step (also available in PROVERIF). However,

both PROVERIF and CRYPTOVERIF have superior performance on many protocols analysed with TAMARIN as well. [14, 18].

2.2 Modelling syntax

A protocol model in TAMARIN is defined in terms of rules. These rules operate on terms, which model cryptographic messages, and facts, which can be thought of as predicates. Section 3.1 builds upon the content of this section to model a real-world protocol.

Rules A rule defines a transition from one state, the premise, to another state, the conclusion. Each *state* is an intermediate step in the protocol where all agents involved, including the adversary, have a certain amount of knowledge. The rule itself can be thought of as either a legitimate agent or an adversary performing some action, such as sending a message over the network.

The premise describes the state that is required for the rule to take effect. The conclusion describes the state resulting from applying the rule to the premise. Algorithm 1 shows the syntax of a rule.

Algorithm 1 Rule example

```
rule: ExampleRule
    [ premise ]  $\longrightarrow$  [ conclusion ]
```

The third, optional, element of a rule is the action fact. An action fact is similar to a fact (discussed later in this section) but is only visible in the protocol trace and not to the adversary. An action fact might, for example, declare that Alice knows the secret key without revealing the secret key to the adversary. Algorithm 2 shows the syntax of a rule with an action fact.

Algorithm 2 A rule with an action fact

```
rule: ExampleRuleWithActionFact
    [ premise ]
    -[ action fact ]  $\rightarrow$ 
    [ conclusion ]
```

There are two keywords important specifically to rules, namely In and Out. The In keyword specifies that a value is received from the public channel, and the Out keyword specifies that a value is send out to the public channel. Algorithm 3 shows these two keywords in use.

Algorithm 3 A rule using In and Out

rule: ExampleRuleWithInAndOut
[In(a)] \longrightarrow [Out(b)]

Lastly, TAMARIN provides a syntax called let bindings that allow a modeller to define a value in terms of other values, much like variables in programming languages. Algorithm 4 shows the syntax for let bindings.

Algorithm 4 Let binding syntax

rule: ExampleRuleWithLetBinding
let b = f(a) in
[In(a)] \longrightarrow [Out(b)]

Terms Terms are used by rules to model cryptographic messages in the protocol. A term can be either a variable or a function. A term that is a variable can be a value or a constant. A value can be thought of as a regular variable as it is seen in programming languages. A constant can be either an atomic value, such as an agent identifier or a label, or a fresh and random value, such as a secret key or a nonce. Algorithm 5 shows how variables can be defined.

Algorithm 5 Value syntax

x \triangleright a value
'ConstantValue' \triangleright an atomic constant

There exist four modifiers for variables as Algorithm 6 shows: fresh variables are prefixed by \sim or suffixed by `:fresh` and can be used for example to create nonces; public variables are prefixed by $\$$ or suffixed by `:pub` and are publically available to all agents, including the adversary; temporal variables are prefixed by $\#$ or suffixed by `:temporal` and represent a point in time; persistent variables are prefixed by $!$ and are never removed from the state.

Algorithm 6 Variable modifiers syntax

\sim a / b: fresh \triangleright a and b are fresh values
 $\$$ a / b: pub \triangleright a and b are public values
 $\#$ i / j: temporal \triangleright i and j are temporal values
!y \triangleright y is a persistent value

A term that is not a variable is a function. This paper only considers functions provided by TAMARIN as built-in functions. These include

3 Modelling in TAMARIN

In this section the NAXOS protocol is introduced, followed by the construction of its model in TAMARIN. Finally, an analysis in TAMARIN of a security property desired from the NAXOS protocol is provided.

The NAXOS protocol is an Authenticated Key Exchange (AKE) protocol published in 2007 with the goal of improving existing AKE protocols against stronger adversaries [15]. The goal of an AKE protocol is to establish an ephemeral key for a communication session, as well as authenticate the communicating agents. The latter is usually achieved through public keys certificates. Some security aspect of an AKE protocol include that an adversary cannot impersonate any agent or that it cannot read any messages sent by any agent after the key exchange.

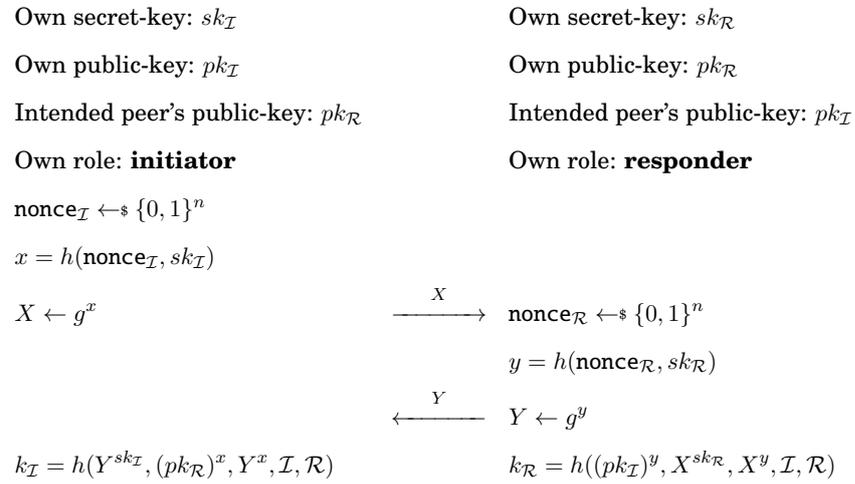


Figure 2. Schematic model of the NAXOS protocol.

NAXOS is a two-step protocol as shown in Figure 2. It is based on a Diffie-Hellman generator g and a hash function h . Both the initiator, \mathcal{I} , and responder, \mathcal{R} , have a Diffie-Hellman key pair (pk, sk) , and for each session both agents also generate a fresh nonce. The secret share of the key of each agent is derived from their secret Diffie-Hellman key (sk) and nonce. The agents exchange their public share of the key and based on that calculate the session key.

This paper evaluates the NAXOS protocol on much weaker properties than it was originally designed for.

3.1 Encoding the NAXOS protocol

Based on the syntax described in Section 2.2, this section constructs a TAMARIN model for the NAXOS protocol described before. This model is derived from the NAXOS model by Cremers and Schmidt [7].

A protocol model in TAMARIN begins with a theory name denoted by theory [NAME] which is used as a human-readable reference to the model. This is followed by the keyword begin, informing TAMARIN the model description starts here.

Algorithm 11 Model prerequisites

```
theory NAXOS
begin
builtins:  diffie-hellman, hashing
```

As shown in Algorithm 11, the NAXOS protocol depends on two built-ins, namely: a Diffie-Hellman generator, exposed through the constant 'g', and a hash function, exposed through the function h.

Following the prerequisites are the rules defining the model of the protocol. Not all rules define the protocol directly, for example the NAXOS model contains a rule defining key management, Algorithm 12 shows this rule. The generate_key_pair rule states that given a fresh value sk, a secret key (!Sk(...)) and public key (!Pk(...)) can be created for an agent X. The facts about the keys are persistent (denoted by !) as they are kept over time. The action fact for this rule records that a key was registered for the agent X.

Because the premise of this rule can always be satisfied, TAMARIN can use this rule at any time to generate a Diffie-Hellman key pair for an agent in the protocol.

Algorithm 12 Model key management

```
rule generate_key_pair:
  let pkX = 'g'^skX:fresh in
  [ Fr(skX:fresh) ]
-[ KeyRegistered(X:pub) ] →
  [ !Sk(X:pub, skX:fresh)
  , !Pk(X:pub, pkX)
  , Out(pkX) ]
```

Algorithm 13 to 15 show the rules modelling the protocol description

directly. More specifically, 13 models the initiator, \mathcal{I} , sending the first message, 14 models the responder, \mathcal{R} , receiving and responding to that message, and finally 15 models \mathcal{I} receiving the message from \mathcal{R} .

The first rule (Algorithm 13) models the initial message from \mathcal{I} to \mathcal{R} . In the premise of this rule a fresh nonce is generated ($\text{Fr}(\text{nonceI}:\text{fresh})$), as well as a Diffie-Hellman key pair ($\text{Sk}(\text{I}:\text{pub}, \text{skI}:\text{fresh})$). The key pair will be generated by TAMARIN using the rule in Algorithm 12 in order to satisfy the premise. The conclusion of this rule records the protocol was initiated ($\text{Initiated}(\dots)$) and sends the public share of \mathcal{I} to the public channel ($\text{Out}(X)$). The fact recording the protocol was initiated is not persistent because \mathcal{I} will only use the nonce once.

Algorithm 13 Model \mathcal{I} 's initial message

```

rule Initiator_initiate:
  let exp = h(<nonceI:fresh, skI:fresh>)
      X = 'g'^exp
  in
  [ Fr(nonceI:fresh)
    , !Sk(I:pub, skI:fresh) ]
  →
  [ Initiated(I:pub, nonceI:fresh, skI:fresh, R:pub)
    , Out(X) ]

```

The next rule (Algorithm 14) models \mathcal{R} receiving a value X from the public channel ($\text{In}(X)$). Note that this message may come from \mathcal{I} but also from an attacker. Similar to Algorithm 13, the rule also generates a fresh nonce ($\text{Fr}(\text{nonceR}:\text{fresh})$) and Diffie-Hellman key pair ($\text{Sk}(\text{R}:\text{pub}, \text{skR}:\text{pub})$). Lastly, it looks up a public key as registered by the rule in Algorithm 12. This should be the key of \mathcal{I} , but may be the key of an attacker as well. The conclusion of the rule only sends the public share of \mathcal{R} to the public channel ($\text{Out}(Y)$).

The action fact of this rule records that \mathcal{R} has accepted a session for nonceR with the key ($\text{Accept}(\dots)$) and the session identifier ($\text{SessionId}(\dots)$).

Algorithm 14 Model \mathcal{R} 's response message

```
rule Responder_respond:
  let KI = 'g'^skI:fresh
      exp = h(<nonceR:fresh, skR:fresh>)
      Y = 'g'^exp
      key = h(<KI^exp, X^skR:fresh, X^exp, I:pub, R:pub>)
  in
  [ In(X)
    , Fr(nonceR:fresh)
    , !Sk(R:pub, skR:fresh)
    , !Pk(I:pub, KI) ]
-[ Accept(nonceR:fresh, R:pub, I:pub, key)
  , SessionId(nonceR:fresh, <R:pub, I:pub, Y, X, 'Resp'>)] →
[ Out(Y) ]
```

The last rule (Algorithm 15) models the last step of the protocol where \mathcal{I} receives a value Y from \mathcal{R} (or an attacker) over the public channel ($\text{In}(Y)$). This rule requires the rule in Algorithm 13 has been applied as it consumes the Initiated fact. Lastly, the premise looks up a public key as registered by the rule in Algorithm 12, which should be the key of \mathcal{R} . The conclusion is empty as the protocol has ended.

Similar to the rule for \mathcal{R} , the action fact of this rule records that \mathcal{I} has accepted a session for nonceI with the key ($\text{Accept}(\dots)$) and the session identifier ($\text{SessionId}(\dots)$).

Algorithm 15 Model \mathcal{I} 's key calculation

```
rule Initiator_finish:
  let KR = 'g'^skR:fresh
      exp = h(<nonceI:fresh, skI:fresh>)
      X = 'g'^exp
      key = h(<Y^skI:fresh, KR^exp, Y^exp, I:pub, R:pub>)
  in
  [ In(Y)
    , Initiated(I:pub, nonceI:fresh, skI:fresh, R:pub)
    , !Pk(R:pub, KR) ]
-[ Accept(nonceI:fresh, I:pub, R:pub, key)
  , SessionId(nonceI:fresh, <I:pub, R:pub, X, Y, 'Init'>)] →
[ ]
```

3.2 Analysing the NAXOS protocol

With the protocol model described in Section 3.1 this section will illustrate how TAMARIN is used to analyse a protocol model. The analysis of a model is based on the lemmata defined for that model. Algorithm 16 shows the lemma that will be analysed for the NAXOS model in this paper.

On a high level the lemma in Algorithm 16 specifies that: *if no agent has registered more than one Diffie-Hellman key pair, then all sessions with the same session id have accepted the same key*. The antecedent of this implication is encoded in the first parentheses and translates to (based on the facts in Algorithm 12): for all (All) agents (A) and temporal values i and j, if the agent has a key registered at time i and j (KeyRegistered(A)@x), then i and j must be the same temporal value.

The consequent of the implication in the lemma is encoded in the second parentheses and translates to (based on the facts in Algorithm 14 and 15): there does not exist (not (Ex ...)) two nonces (nI nR), two keys (keyI keyR), two agents (I R), and a session id (sid) such that I and R have the same sid for their respective nonce (SessionId(nI, sid) & SessionId(nR, sid)) and the key I and R accept for that nonce (Accept(nI, I, R, keyI) & Accept(nR, R, I, keyR)) does not equal the key of the other agent (not(keyI = keyR)).

Algorithm 16 Security property "same key"

lemma Same_key:

$$\begin{aligned}
 & \text{"(All A \#i \#j . KeyRegistered(A)@i \& KeyRegistered(A)@j} \\
 & \qquad \qquad \qquad \implies (\#i = \#j))} \\
 & \implies (\text{not (Ex \#i1 \#i2 \#i3 \#i4 nI nR keyI keyR I R sid .} \\
 & \qquad \text{SessionId(nI, sid)@i1 \& SessionId(nR, sid)@i2} \\
 & \qquad \text{\& Accept(nI, I, R, keyI)@i3 \& Accept(nR, R, I, keyR)@i4} \\
 & \qquad \text{\& not(keyI = keyR)}} \\
 & \text{))"}
 \end{aligned}$$

TAMARIN proves a lemma by finding *traces* in the protocol which are allowed by the model's rules. A trace is an ordered set of rules applied to available terms. Fresh terms are always available, other terms can be made available by applying a rule on available terms. If a term is not persistent it becomes unavailable after a rule consumed it.

For the lemma in Algorithm 16, TAMARIN will try to construct traces where the consequence is not satisfied (but the antecedent is) in order to

show an attack is possible. However, as the protocol is secure it will prove by contradiction that no attack traces can be constructed.

For example, TAMARIN may assume the keys are not equal and work backwards from the other facts in the consequence. Based primarily on the rules and facts in Algorithm 14 and 15, it will conclude that the model does not allow for a scenario where the keys differ but the `SessionId` and `Accept` facts are correct.

4 Conclusion

In this paper the formal verification system TAMARIN PROVER was examined by modelling and analysing the NAXOS protocol. An extensive overview of the relevant syntax was provided, as well as insights into how to model a real-world protocol. Lastly, the analysis of a protocol by TAMARIN was discussed with a specific security property for the NAXOS protocol.

The analysis of NAXOS takes less than 1 second on a user-grade system (i7-6700HQ CPU @ 2.60GHz \times 8) which is in line with the figures reported by Lafourcade et al. [14]. Modelling a protocol in TAMARIN is approachable provided one has sufficient prior knowledge of cryptographic protocols and discrete mathematics.

Based on the research for this paper, future work for TAMARIN PROVER should focus primarily on documentation. Although the existing manual is a useful resource, TAMARIN lacks proper documentation. Explaining syntax through generic examples with little syntactic sugar, as well as graphics, may help newcomers understand TAMARIN more easily.

References

- [1] David Basin, Cas Cremers, Tiffany Hyun-Jin Kim, Adrian Perrig, Ralf Sasse, and Pawel Szalachowski. Arpki: Attack resilient public-key infrastructure. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 382–393. ACM, 2014.
- [2] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. A formal analysis of 5g authentication. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1383–1396. ACM, 2018.
- [3] David Basin, Saša Radomirovic, and Lara Schmid. Alethea: A provably secure random sample voting protocol. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 283–297. IEEE, 2018.

- [4] Bruno Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *Proceedings 14th IEEE Computer Security Foundations Workshop, 2001.*, pages 82–96. IEEE, 2001.
- [5] Bruno Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing*, 5(4):193–207, 2008.
- [6] Paolo Camurati and Paolo Prinetto. Formal verification of hardware correctness: Introduction and survey of current research. *Computer*, 21(7):8–19, 1988.
- [7] Cas Cremers and Benedikt Schmidt. Naxos eCK. https://github.com/tamarin-prover/tamarin-prover/blob/develop/examples/ake/dh/NAXOS_eCK.spthy, 2012. Online; accessed 27 March 2019.
- [8] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe. A comprehensive symbolic analysis of tls 1.3. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1773–1788. ACM, 2017.
- [9] Cas Cremers, Marko Horvat, Sam Scott, and Thyla van der Merwe. Automated analysis and verification of tls 1.3: 0-rtt, resumption and delayed authentication. In *Security and Privacy (SP), 2016 IEEE Symposium on*, pages 470–485. IEEE, 2016.
- [10] Casimier Joseph Franciscus Cremers. *Scyther: Semantics and verification of security protocols*. Eindhoven University of Technology Eindhoven, Netherlands, 2006.
- [11] Jannik Dreier, Charles Duménil, Steve Kremer, and Ralf Sasse. Beyond subterm-convergent equational theories in automated verification of stateful protocols. In *International Conference on Principles of Security and Trust*, pages 117–140. Springer, 2017.
- [12] Jannik Dreier, Lucca Hirschi, Sasa Radomirovic, and Ralf Sasse. Automated unbounded verification of stateful cryptographic protocols with exclusive or. In *31st IEEE Computer Security Foundations Symposium (CSF’2018)*, 2018.
- [13] Santiago Escobar, Catherine Meadows, and José Meseguer. Maude-npa: Cryptographic protocol analysis modulo equational properties. In *Foundations of Security Analysis and Design V*, pages 1–50. Springer, 2009.
- [14] Pascal Lafourcade and Maxime Puy. Performance evaluations of cryptographic protocols verification tools dealing with algebraic properties. In *International Symposium on Foundations and Practice of Security*, pages 137–155. Springer, 2015.
- [15] Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger security of authenticated key exchange. In *International conference on provable security*, pages 1–16. Springer, 2007.
- [16] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. The tamarin prover for the symbolic analysis of security protocols. In *International Conference on Computer Aided Verification*, pages 696–701. Springer, 2013.

- [17] John Mitchell, Andre Scedrov, Nancy Durgin, and Patrick Lincoln. Undecidability of bounded security protocols. In *Workshop on Formal Methods and Security Protocols*, 1999.
- [18] Sonia Santiago Pinazo and Santiago Escobar Román. Advanced features in protocol verification: theory, properties, and efficiency in maude-npa. *Escobar Román Santiago*, 2015.
- [19] Benedikt Schmidt, Simon Meier, Cas Cremers, and David Basin. Automated analysis of diffie-hellman protocols and advanced security properties. In *Computer Security Foundations Symposium (CSF), 2012 IEEE 25th*, pages 78–94. IEEE, 2012.

A. Full TAMARIN model of the NAXOS protocol

```
theory NAXOS
begin

builtins: diffie-hellman, hashing

rule generate_key_pair:
  let pkX = 'g'^~skX in
    [ Fr(~skX) ]
  --[ KeyRegistered($X) ]->
    [ !Sk($X, ~skX)
      , !Pk($X, pkX )
      , Out(pkX) ]

rule Initiator_initiate:
  let exp = h(<~nonceI, ~skI>)
      X   = 'g'^exp
  in
    [ Fr(~nonceI)
      , !Sk($I, ~skI) ]
  -->
    [ Initiated($I, ~nonceI, ~skI, $R)
      , Out(X) ]

rule Responder_respond:
  let KI = 'g'^~skI
      exp = h(<~nonceR, ~skR>)
      Y   = 'g'^exp
      key = h(<KI^exp, X^~skR, X^exp, $I, $R>)
```

```

in
  [ Fr(~nonceR)
    , !Ltk($R, ~skR)
    , !Pk($I, KI)
    , In(X) ]
--[ AcceptR(~nonceR, key), SessionId(~nonceR, <$I, $R, Y, X, 'Resp'>)]->
  [ Out(Y) ]

rule Initiator_finish:
  let KR = 'g'^~skR
      exp = h(<~nonceI, ~skI>)
      X   = 'g'^exp
      Y   = 'g'^ h(<~nonceR, ~skR>)
      key = h(<Y^~skI, KR^exp, Y^exp, $I, $R>)
in
  [ Initiated($I, ~nonceI, ~skI, $R)
    , !Pk($R, KR)
    , In(Y) ]
--[ AcceptI(~nonceI, key), SessionId(~nonceI, <$I, $R, X, Y, 'Init'>)]->
  [ ]

lemma Same_key:
"
  (All A #i #j . KeyRegistered(A)@i & KeyRegistered(A)@j ==> (#i = #j))
==>
(not
  (Ex #i1 #i2 #i3 #i4 nonceI nonceR keyI keyR sid .
    AcceptI(nonceI, keyI)@i1
    & AcceptR(nonceR, keyR)@i2
    & SessionId(nonceI, sid)@i3
    & SessionId(nonceR, sid)@i4
    & not(keyI = keyR)
  )
)
"
end

```

How deep is the learning used in wireless communications? Can we go deeper?

Amaanat Ali

amaanat.ali@aalto.fi

Tutor: Prof. Dr. Risto Wichmann

Abstract

Futuristic 5G wireless networks will cater to an extremely diverse range of applications and thus demand flexible network architecture with massive number of control parameters leading to intractable mobile network control problems. Deep Learning (DL) enables experimental data research by uncovering correlations and feature extraction on Big Data allowing hierarchical feature extraction from raw data with vast amounts of extremely heterogeneous data exhibiting complex correlations saving tremendous effort of hand-crafted feature engineering. Three potential areas, advanced positioning, resource scheduling and signal processing, relevant to wireless communication that may benefit from cross-functional application of DL principles are discussed. This report improves the state-of-the-art in two ways. Firstly, a conceptual model of the software architecture of a DL augmented radio access scheduler is presented. Secondly, an conceptual architecture of a DL enabled 5G radio access network is outlined. Lastly, the challenges that lay ahead for research and testing of such augmented radio networks are also discussed.

KEYWORDS: *Machine Learning, Deep Learning, Wireless Communications, Big Data, Augmented Intelligence based 5G Radio Access Network Framework.*

1 Introduction

Next generation wireless technologies are demand high flexibility requiring new paradigms to meet conflicting requirements. Traditional approaches to build networks need to be augmented by algorithms that allow continuous learning of the environment and reorganize network behavior around it. Tools employing deep learning may solve problems in this area. This report surveys the crossover between two functional areas involving deep learning and mobile network design and specifically explores how deep learning may be applied to improve radio access performance in next generation wireless networks.

2 Background

2.1 Intelligent 5G Networks

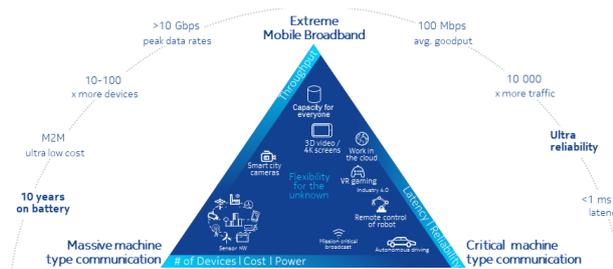


Figure 1. 5G Service Triad [1].

5G (or NR) is expected to fulfil the stringent requirements set by the International Telecommunication Union (ITU) by supporting an extremely flexible range of services [2]. Figure 1 summarizes three vertical areas, viz. Enhanced Mobile Broadband (eMBB), Massive Scale Communication (Massive IoT) and Ultra-Reliable Low Latency Services (URLLC) [1] [2]:

A 5G cellular network comprises of a Radio Access Network (RAN) and a Core Network (CN) part. The Core Network is the switching centre, that links to the RAN and mediates access to the Internet and Public Switched Telephone Network (PSTN). The RAN consists of a set of Base Stations (BSs) connected to the CN through an interface (A) that provides Control and User Plane functions. The BS implements the wireless transceiver function, i.e. access technology by providing the modulation and demodulation roles. The BSs may be connected in a point-to-point manner with an interface (B) which allows user mobility. The BS also performs quasi-real

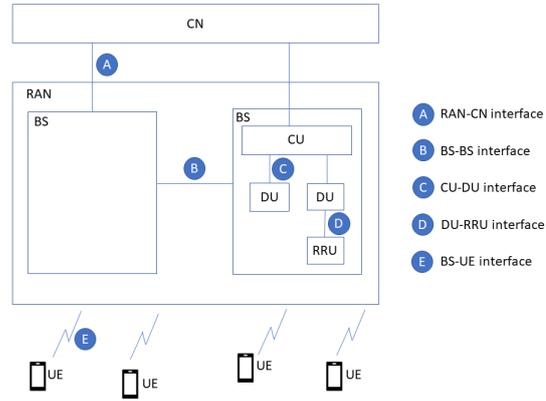


Figure 2. Overall Architecture of a Cellular Network [26].

time network control functions (radio resource allocation, measurements and mobility management) and real-time dynamic resource allocation using an air interface scheduler [26].

The Controller Unit (CU) centrally implements functions to manage the Data Units (DUs) that implement the latency critical, baseband, radio transmit and receive processing functions via. an interface (C) which typically uses high capacity transport connections. A DU may be split to isolate the baseband processing unit and the Remote Radio Unit (RRU) that transmits and receives the physical signals over the air (E). A UE may be capable of transmitting and receiving data simultaneously from multiple BS's.

Meeting the service requirements in Figure 1 demands a very flexible network architecture and implementation with detailed planning and tuning which are error prone. Artificial Intelligence (AI) is expected to help automate these processes by augmenting the networks with actionable intelligence. This paper focuses of the BS relevant for wireless communication. A more comprehensive list of applications applicable to the other network components (e.g. the Core Network) are identified in section IV of [4].

Traditional mobile communication networks rely on complex network algorithms. Although these algorithms routinely perform non-linear tasks (e.g. resource scheduling, handover optimization and mobility robustness optimizations to deal with link failures using SON techniques, etc.), their ability to self-learn and self-optimize is quite limited. Traditional networks built using these algorithms are not capable to detect anomalies (e.g. spurious packet injection by malicious users, jamming of uplink channels, etc.). There are several reasons for this. Firstly, the amount

of heterogeneous data available from a mobile network is rather limited. Secondly, the computing platforms for analyzing and extracting relevant features are non-existent. Furthermore, most of the algorithms and software do not have the potential to scale to address the wide spectrum of service requirements for novel 5G services as they are built using traditional programming languages [4] [11] [12].

Successful ML practices can be applied to the Self-Organizing Network (SON) [7] [8], energy efficiency [9] and traffic offloading [10]. A concrete example of SON is 4G Multi-band load balancing wherein subscribers are served across multiple frequencies based on an adaptive algorithm relying on daily KPIs relative to the available spectrum. The output parameters from the dynamic SON algorithm improves network performance in all areas, viz. capacity, coverage, accessibility and mobility with minimal manual work to set the control parameters in every cell of the network. DL based approaches may further augment SON algorithms using data analytics in the areas of intrusion detection, anomaly detection and enhancing Quality of Experience (QoE) [11] [12]. Deep sensing [13] is another area.

2.2 Potential for Augmented Intelligence Based Radio Access Network Design

A computer cannot perform subjective and intuitive tasks using traditional programming languages. A Machine Learning (ML) system can learn features from data by extracting patterns (or uncovering correlations). Applying artificial intelligence to solve real-life problems needs feature extraction followed by a relatively simpler task of supplying these features to a ML algorithm essentially creating a model [3].

Feature engineering is tedious, time-consuming and requires manual intervention with system and domain level knowledge. DL allows feature extraction and predictions (or classifications) in one-shot without being explicitly programmed. However, the training time for DL models is very high and is computationally very intensive [6]. To mitigate this, parallel processing architectures with distributed memory are needed to execute DL software models.

Systematic mining of “actionable” information allows solving problems specific to the mobile networking domain [4]. The performance of DL improves significantly with larger data. Features learned may be reused by transferring (or reapplying) the learning to related tasks. [4].

3 Deep Learning Driven Wireless Communication Networks

3.1 State of the Art Artificial Intelligence Models

Generally, there are three well-known ML paradigms which all need human involvement [3] [11].

- **Supervised learning:** allows to infer a non-linear function that maps a given input to an output i.e. curve-fitting and useful in models performing either regression or classification.
- **Unsupervised learning:** allows to uncover correlations and patterns when the underlying data structure is heterogeneous and unstructured and useful in anomaly or fault detection (e.g. spam detection). The insights gathered may be converted to a supervised learning once the model is verified (i.e. reliable).
- **Reinforcement learning:** In this approach, the focus is on maximizing the reward (i.e. return) over a given set of input parameters to find a balance between exploration (from what is possible) and exploitation (from what is available). Radio resource management algorithms in wireless base stations can exploit this paradigm.

Deep learning models In modern literature there are several deep learning models discussed. However, the most salient ones are summarized here.

- **Multilayer Perceptron (MLP)** An MLP performs a non-linear transformation using an activation function fed with a weighted version of an input with an optional bias and are mostly used for assisting with feature extraction or at the final stage of a complex neural network due to their simplicity [4].
- **Restricted Boltzmann machine (RBM)** This is a shallow neural network comprising of a visible(outer) and inner(hidden) layers. Nodes in the visible layer are connected to the hidden layer with restricted connectivity. RBM's use forward and backward propagation to learn features and may be cascaded to create Deep Belief Networks (DBN).

RBM's are useful in anomaly detection and hence are very suitable for mobile networks as well [4].

- **Autoencoders** An autoencoder is trained to copy the output to the input with a high amount of fidelity and while doing so, an autoencoder learns to represent the input data in a compact form (entropy preservation). The succinct representation is useful because it implies dimensionality reduction (i.e. feature extraction). [4].
- **Convolutional Neural Network** To optimize the number of computations performed, one approach could be to employ partial connection between layers and employ locally connected filters to identify correlations on specific areas of the input data. The identical nature of images and spatial mobile data (traffic snapshot on a geographical area, mobility of users), i.e. represented in a map which is another way of expressing an image with pixels [4].
- **Recurrent Neural Network** Just as a CNN is specialized for processing an image (or a grid of values), an RNN is specialized for analyzing a time-series (or a sequence of values). Most RNN's can process sequences of variable length as well. This permits easier searching for patterns with different levels of time granularity in time series analysis in mobile networks [4].
- **Generative Adversarial Network (GAN)** This comprises of a framework comprising of a Generative model (G) and a Discriminative model (D) and depends on a corroboration process forcing the learning. GAN's are mostly used to generate synthetic training data mimicking mobile networks when appropriate datasets cannot be shared by network operators due to privacy and competitive reasons [4].
- **Deep Reinforcement Learning (DRL)** relies on reward signals as a feedback to learn from the past mistakes. A group of methods approximate a value function (deep Q learning) or policy function (policy gradient approach) and can help solve mobile networking problems when formulated as Markov Decision Processes (MDP). Furthermore, DRL algorithms allow handling high dimensionality (i.e. large set of input parameters or variables) which are very typical of next generation wire-

less mobile networks [4].

3.2 Mobile Big Data as a Pre-requisite for Deep Learning Based RAN

The most important pre-requisite to intelligently augment a wireless communication network is the ability to gather large volumes of heterogeneous data to allow online and offline analysis; enabling AI model training and feature extraction. As such mobile Big Data may be gathered at two levels:

- **Network level data:** obtained as part of the daily of the network infrastructure which includes call logs, performance counters and operations management, network measurements etc.
- **User or application level data:** obtained in a crowd-sourced manner which includes sensors, specific application packets, user profiles etc.

Any wireless communications network which seeks to augment its operation with "actionable" intelligence must be able to access large amounts of heterogeneous Big Data and store them for processing and analysis. A large suite of Big Data analysis platforms are available and are listed in [4] [8] [9] [12].

3.3 Applications of Deep Learning to Mobile and Wireless Communication Networks

There are several areas identified in section IV of [4] which are applicable to several network components in a mobile wireless network. As mentioned earlier this paper only focuses specifically on AI based augmentation of the functional components in the BS that implements the air interface (i.e. the physical layer) of the wireless communication network.

3.3.1 Deep learning driven mobility analysis and advanced positioning

As discussed in section 2.1, URLLC services demand high throughput and low-latency and rely on precise real-time positioning [14] [15]. Mobile network resource management e.g. handover and admission control algorithms may be improved allowing the network to add or remove radio links ensuring true multi-connectivity by knowing the precise location of

a user by learning the environment [16]. In addition, sound knowledge of the radio propagation environment is needed with mmWave networks, which inherently suffer from beam blocking, to ensure high data rate at cell-edge with optimal measurement reporting loads on the network up-link [18].

To support the above requirements, DL allows improved trajectory prediction by coupling time series analysis and spatial data (e.g. geo-location maps). User localization can be improved by combining heterogeneous information from different sources (sensors, mobile measurements reporting receives signal strength or SINR, Channel State Information, geo-location maps, etc.). In some cases, a combination of supervised and unsupervised learning techniques may be employed.

3.3.2 Deep learning driven network control

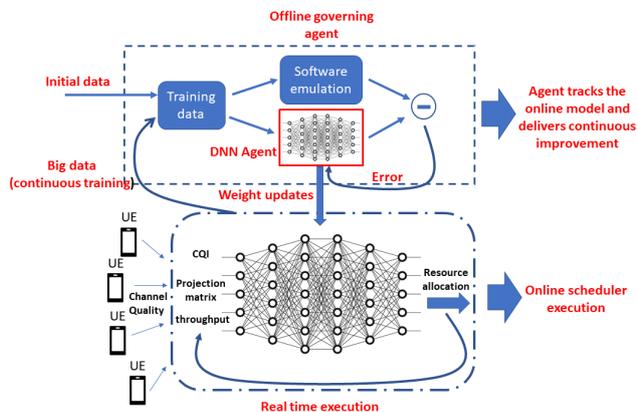


Figure 3. Machine Learning applied to a radio access network problem [2].

URLLC and eMBB applications in section 2.1 benefit from low scheduling latency and improved throughput. DNN's can be employed for non-linear function approximation, especially as the number of control parameters in 5G are ≤ 2500 . A reinforcement learning approach is described in the conceptual scheduler model in Figure 3. A combination of online and offline learning tunes the DNN weights providing feedback data back to the offline process that re-validates the weights. The entire process can be monitored using performance counters [20] [21].

Other practical examples of using the above approach are as follows:

- Improving the downlink Signal To Interference and Noise ratio (SINR) in mmWave beamformed network using deep reinforcement learning towards a targeted value [18].

- Optimizing handover performance in cellular networks compared to traditional handover signalling [19].
- Wireless radio resource management and power control [21].
- Optimizing energy efficiency in base station sleep patterns [23].

3.3.3 Deep learning driven signal processing

DL has promising applications in the 5G BS physical layer and applicable to all the services listed in section 2.1. These are quickly summarized as follows:

- **MIMO detection** Multi-user detection can be performed for various constellations providing reliable detection. However, the aspects of hardware complexity, reliability and general purpose fitness to commercial communication systems [24] need to be dealt with.
- **Modem** allows processing of baseband I/Q samples in the presence of channel impairments (power delay profile, frequency selective fading, local oscillator offset and Additive White Gaussian Noise (AWGN)). Long-Short Term Memory approach (LSTM) provides superior performance compared to the other neural network approaches [17].

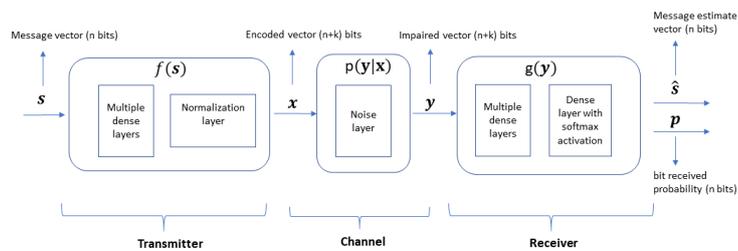


Figure 4. Autoencoder applied to channel coding [2].

- **Channel coding** Autoencoders allow for unsupervised learning illustrated in Figure 5. It comprises of a transmitter (modulation), a communication channel (distorting the transmitted signal in many ways) and a receiver (equalizing the impacts of the channel). The communication blocks are organized to ensure that the output message vector \hat{s} (which is the approximation) follows the input message vector s , an autoencoder function is achieved. The autoencoder an encoder and de-

coder neural network. The encoder block provides x that is resistant to the effects of the communication channel. CNN's may further be able learn the noise correlation and improve channel noise estimation errors (using the property of feature extraction and uncovering correlations). Autoencoders suffer from higher computational overhead [5] [17].

3.4 Conceptual Architectural Framework of Machine Learning Enabled 5G Radio Access Network

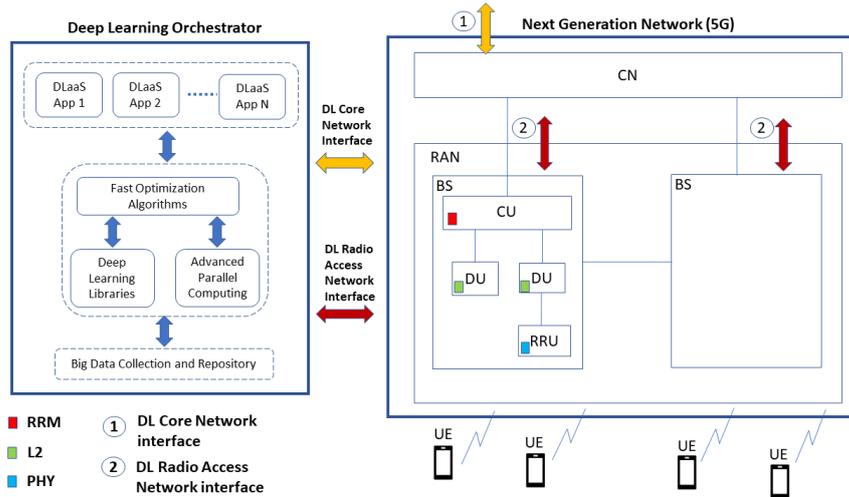


Figure 5. ML focussed RAN architecture [2].

Applying the learning from the previous sections, a conceptual architectural model of a DL enabled radio network architecture is shown in Figure 5 that may be applied for next-generation wireless communication networks (including 5G). The architecture comprises of a centralized DL Orchestrator (DLO) collecting data from different network nodes using the new DL Network Interface (RAN and CN). Augmented intelligence is fed back to the network to alter their behavior. Each of the functional components within the BS are expected to be upgraded with the corresponding Radio Resource Management (RRM), Layer 2 (L2) and Physical Layer (PHY) DL enabled components executing on a hardware platform supporting DL accelerators (those components were already discussed in the previous section 3.3). Each of the DL aware components may further participate in continuous improvement (or upgrades) by transmitting information about their execution using the already available interfaces and finally aggregated towards the DLO. In 3GPP, one of the key topics in the future of AI/ML enabled radio access network is one of the potential standardization topics for the 5G in Release 17.

3.5 Challenges in Applying Deep Learning to Wireless Communication Networks

Big-Data enabled DL poses a problem of data availability for training purposes. Firstly, there are security and privacy issues hindering collection of data. Secondly, only operators have access to diverse amounts of such data and are averse to share due to competitive or legal reasons. Thirdly, latency and overhead of data transfer to an orchestrator will matter if on-line analysis is needed. Fourthly, heterogeneous forms of data require basic labelling which is effort-wise prohibitive when dealing with tons of data. Finally, mobile terminals with limited energy and computational complexity cannot participate in the learning process and software upgrades of the DNN model to mobile terminals are required [4] [8] [9].

Polar codes suffer from sequential high decoding complexity resulting in decoding latency. For small block lengths, MAP performance can be matched but exponentially increases as codewords become longer and requiring training on the entire codebook. They only provide good performance at high SINR, limiting the decoder dynamic range. [17] [25].

Current approaches to communication receivers assume bandwidth normalized datasets which is unrealistic and requires adaptation by resampling signals to normalize the bandwidth or by learning features with different channel bandwidths. Models that can seamlessly resample, synchronize and equalize non-linear channel impairments require more research [24] [25].

Applications performing MIMO detection need to factor in hardware complexity, reliability, robustness and ease of integration. Current proposals do not exhibit the flexibility to adapt to the constellation used nor the number of users. A change to either one necessitates a new learning behavior (resulting in a new network) [17] [24] [25].

From the modelling perspective, a simple model cannot perfectly match input data. Debugging a non-optimal model is non-trivial as the complexity is abstracted at the hidden nodes rendering debugging extremely tedious and time-consuming (even impossible). In the absence of real field data, simulated data may be used to train the neural network. However, one must be prepared that the testing distribution may not match the training distribution and in some cases even fatally mismatching. In many cases, the testing effort may render the whole activity useless [4].

4 Conclusion

This report explored whether at all traditional mobile communication networks have benefited from embedded artificial intelligence and how DL based approaches may be applied to mobile and wireless communications systems, especially focusing futuristic radio access technologies like 5G. This analysis is vital to understand which of the functions of a 5G base station may benefit by augmented intelligence to deliver the stringent requirements for a wide range of 5G services, e.g. eMBB, URLLC and mMTC.

The first topic introduced enablers for a 5G mobile network. The report then discussed the reasons why traditional mobile networks have been largely unable to employ DL principles, further elaborating what it means to apply DL to the 5G wireless physical layer.

The second topic provided a background of the most salient DL models relevant to mobile wireless communication and it was found that there are not just one but several interesting neural network architectures that are relevant for solving different problems (feature extraction, anomaly detection, auto-encoding, spatial data analysis, time series analysis, data with huge input dimensions).

The third topic delved deeper on applying deep learning to the most relevant potential areas relevant to the area of wireless communication, viz. advanced positioning, resource scheduling and signal processing, critical to 5G base station wireless functionality and implementation.

The fourth topic highlights two main contributions of the report. Firstly, a conceptual software architecture of an augmented intelligence based radio scheduler was described. The model uses a combination of online and offline reinforcement learning approaches to balance the short and long term objectives of a radio resource scheduler to enhance its performance. Secondly, a conceptual architecture of a 5G DL enabled radio access network is discussed with interface requirements to an orchestrator.

The fifth and final topic of the report summarized the main challenges and research opportunities that lay ahead in front of us in exploiting the interconnect between AI and mobile and wireless communication network design. It may be concluded that DL enabled mobile communication networks herald a huge number of research and engineering opportunities and futuristic mobile networks will be enhanced with ML/DL enabled features.

References

- [1] Nokia, "5G Use Cases and requirements", White paper, 2014. Available: www.resources.alcatel-lucent.com.
- [2] ITU, "Minimum requirements related to technical performance for IMT-2020 radio interface(s)", ITU-R M.2410-0, Nov 2016. Available: www.itu.int.
- [3] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning", MIT Press, 2016.
- [4] C. Zhang, P. Patras, H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey", 2018.
- [5] Timothy O'Shea and Jakob Hoydis, "An introduction to deep learning for the physical layer", *IEEE Transactions on Cognitive Communications and Networking*, 3(4):563–575, 2017.
- [6] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep learning. Nature", 521(7553):436–444, 2015.
- [7] Ali Imran, Ahmed Zoha, and Adnan Abu-Dayya, "Challenges in 5G: how to empower SON with big data for enabling 5G", *IEEE Network*, 28(6):27–33, 2014.
- [8] Kan Zheng, Zhe Yang, Kuan Zhang, Periklis Chatzimisios, Kan Yang, and Wei Xiang, "Big data-driven optimization for mobile networks toward 5G", *IEEE network*, 30(1):44–51, 2016.
- [9] Wu Jinsong and Guo Song and Li Jie and Zeng Deze, "Big data meet green challenges: big data toward green applications", *IEEE Systems Journal*, 10(3):888–900, 2016.
- [10] Chen, Xianfu and Wu, Jinsong and Cai, Yueming and Zhang, Honggang and Chen Tao, "Energy-efficiency oriented traffic offloading in wireless networks", *IEEE Journal on Selected Areas in Communications*, 33(4):627–640, 2015.
- [11] Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo, "Machine learning paradigms for next generation wireless networks", *IEEE Wireless Communications*, 24(2):98–105, 2017.
- [12] Rongpeng Li, Zhifeng Zhao, Xuan Zhou, Guoru Ding, Yan Chen, Zhongyao Wang, and Honggang Zhang, "Intelligent 5G: When cellular networks meet artificial intelligence", *IEEE Wireless Communications*, 2017.
- [13] Nicholas D Lane and Petko Georgiev, "Can deep learning revolutionize mobile sensing?", *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pages 117–122. ACM, 2015.
- [14] J Venkata Subramanian and M Abdul Karim Sadiq, "Implementation of artificial neural network for mobile movement prediction", *Indian Journal of science and Technology*, 7(6):858–863, 2014.
- [15] Joao Vieira, Erik Leitinger, Muris Sarajlic, Xuhong Li, and Fredrik Tufvesson, "Deep convolutional neural networks for massive MIMO fingerprint-based positioning", *28th Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, IEEE, 2017.

- [16] Kai Zhao, Sasu Tarkoma, Siyuan Liu, and Huy Vo, "Urban human mobility data mining: An overview in Big Data", IEEE International Conference on, pages 1911–1920, 2016.
- [17] Timothy J. O’Shea, Johnathan Corgan, T. Charles Clancy, “Deep Architectures for Modulation Recognition”, IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN) 2017.
- [18] Faris B Mismar and Brian L Evans, "Deep reinforcement learning for improving downlink mmwave communication performance", arXiv:1707.02329, 2017.
- [19] Wang, Zhi and Li, Lihua and Xu, Yue and Tian, Hui and Cui, Shuguang, "Handover optimization via asynchronous multi-user deep reinforcement learning", International Conference on Communications (ICC), pages 1–6. IEEE, 2018.
- [20] Sandeep Chinchali, Pan Hu, Tianshu Chu, Manu Sharma, Manu Bansal, Rakesh Misra, Marco Pavone, and Katti Sachin, "Cellular network traffic scheduling with deep reinforcement learning", National Conference on Artificial Intelligence (AAAI), 2018.
- [21] Haoran Sun, Xiangyi Chen, Qingjiang Shi, Mingyi Hong, Xiao Fu, and Nikos D Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management", Signal Processing Advances in Wireless Communications (SPAWC), 18th International Workshop on, pages 1–6. IEEE, 2017.
- [22] Luo, Changqing and Ji, Jinlong and Wang, Qianlong and Yu, Lixing and Li Pan, "Online power control for 5G wireless communications: A deep Q-network approach", International Conference on Communications (ICC), pages 1–6. IEEE, 2018.
- [23] Liu, Jingchu and Krishnamachari, Bhaskar and Zhou, Sheng and Niu, Zhisheng, "DeepNap: Data-driven base station sleeping operations through deep reinforcement learning", IEEE Internet of Things Journal, 2018.
- [24] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, “On deep learning based channel decoding”, in Proc. IEEE 51st Annu. Conf. Inf. Sciences Syst. (CISS), 2017, pp. 1–6.
- [25] Neev Samuel, Tzvi Diskin, and Ami Wiesel, “Learning to Detect”, IEEE Transactions on Signal Processing, Year, 2019.
- [26] 3GPP TSG RAN, “NG-RAN Architecture description”, 3rd Generation Partnership Project, V15.4.0, Dec 2018.

Survey on machine learning for signal detection

Ville Viinikka

ville.viinikka@aalto.fi

Tutor: Verónica Toro-Betancur

Abstract

This survey is motivated by the signal detection possibilities in cognitive radios. We discuss signal detection as it relates to narrowband and wide-band spectrum sensing. Very few machine learning methods are available for signal detection. We also discuss prominent machine learning methods used in automatic modulation classification, which is a relatively similar task. These methods include neural networks, support vector machines and decision trees.

KEYWORDS: machine learning, signal detection, automatic modulation classification

1 Introduction

The rise of multimedia applications has increased the demand for the radio frequency spectrum [24]. Traditionally licensed spectrum has been allocated exclusively to the primary users, leading to inefficient use of the spectrum [7]. Therefore, new methods for spectrum sharing have been developed based on cognitive radios (CR) and spectrum sensing. Cognitive radios are radios that adapt their operating parameters after sensing

their environment [24]. Spectrum sensing allows cognitive radios to detect the presence of spectrum holes, where the primary users of the spectrum are not transmitting [19]. This potentially allows the cognitive radio to utilize this licensed spectrum without interfering with the primary user.

Cognitive radios use signal detection algorithms for primary user detection, which is a part of spectrum sensing [15]. Signal detection can also be the first step before demodulation of the signal [17].

After a signal has been found by using signal detection, further classification may sometimes be necessary. Cognitive radios may want to detect whether the signal is from a primary user or a secondary user [4]. Automatic modulation classification (AMC) can be used to detect the modulation of the signal. This can also be useful if the signal has to be demodulated but the modulation is unknown [17].

Recent advances in machine learning have made solving more and more complex machine learning problems possible. In this survey we want to see how machine learning methods have been applied to the problem of signal detection. Signal detection methods that are not based on machine learning have been well covered in literature [24, 10, 1, 2]. A survey that includes machine learning methods for spectrum sensing networks also exists [3]. We attempt to extend this research by surveying non-networked, non-cooperative machine learning methods. Because very few such methods were found, we extend the scope to include also machine learning methods for AMC. Machine learning based modulation classification is also well surveyed [11, 6]. We add to this research by providing new results from the year 2017 and later.

Section 2 of this survey discusses different methods for spectrum sensing based on signal detection. Both machine learning and other methods are included. Non-machine learning methods are provided for reference, and also because they are often used in combination with machine learning based modulation classification. Methods for AMC are discussed in Section 3. The final section concludes the survey.

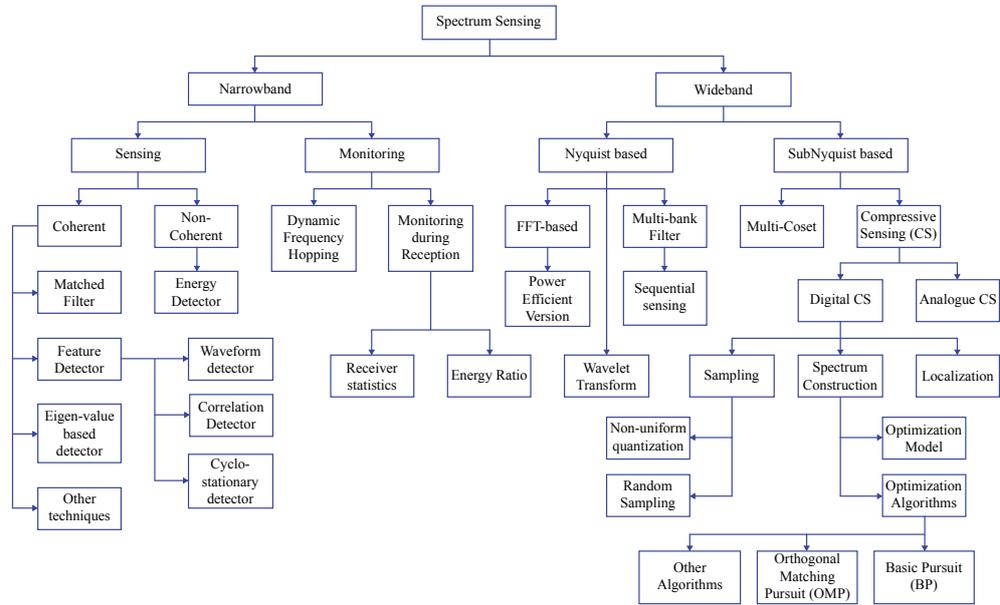


Figure 1. Classification of spectrum sensing approaches (vectorized from the original) [2].

2 Signal detection

In detection theory, signal detection is the process of determining if a signal is present among noise [20]. Signal detection is an elementary part of both narrowband and wideband spectrum sensing. Narrowband spectrum sensing is used when sensing a slice of spectrum that only contains a single signal. The sensing algorithm will classify the whole slice as occupied or unoccupied if a signal is detected or not. Wideband sensing involves splitting the spectrum into multiple slices that are classified separately. This allows wideband sensing to detect spectral opportunities that cover only a part of the whole spectrum [18].

A spectrum sensing algorithm can contain both narrowband and wideband components. For example, the initial scan for spectrum holes can be performed by a wideband algorithm. When a hole has been found and selected for operation, narrowband sensing can be used to monitor if that specific channel becomes occupied [2].

Figure 1 provides an overview of the hierarchy of spectrum sensing approaches. The two main classes, narrowband and wideband, are discussed in their respective sections below. From the narrowband side, only the sensing methods are discussed.

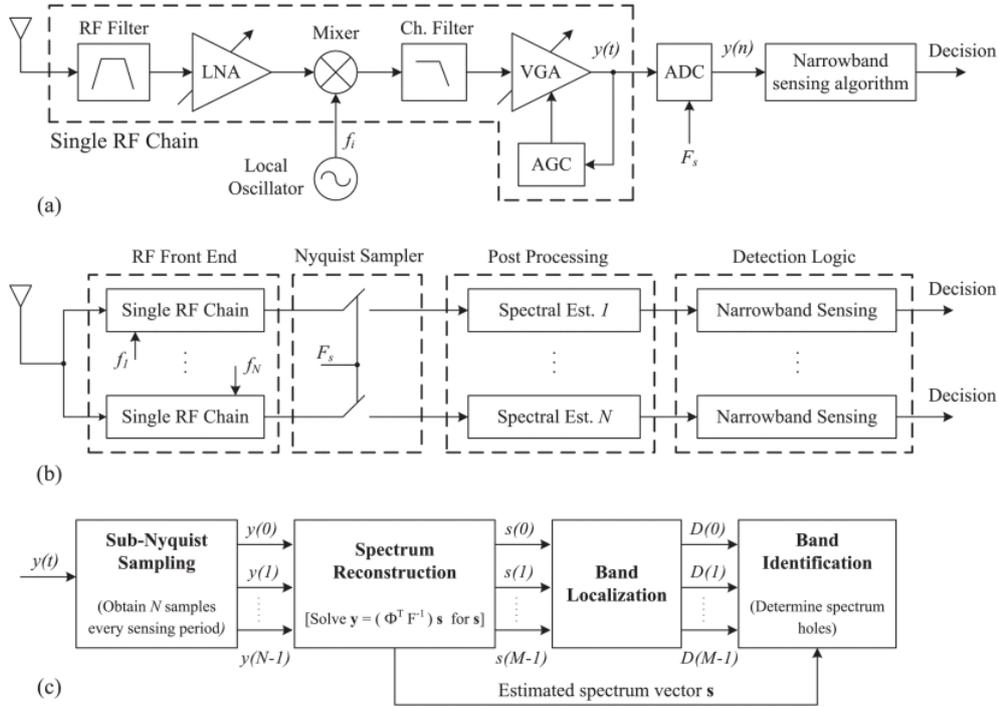


Figure 2. Received diagrams for (a) narrowband sensing, (b) wideband Nyquist sensing and (c) wideband non-Nyquist compressive sensing [2].

2.1 Narrowband sensing

Non-cooperative narrowband sensing can be performed by simply running a known signal detection algorithm on the input spectrum. Traditional methods for signal detection include energy detection, matched filtering and cyclostationary detection [4]. Energy detection works by measuring the signal energy over a certain time period and then asserting a detection if the energy exceeds some predefined threshold. This does not require any knowledge of the type of the detected signal, but it is sensitive to variations in the background noise power [7]. Matched filtering is the optimal signal detector, but it requires precise knowledge of the incoming signal parameters [4]. Cyclostationary detection utilizes the periodicity in most modulation types to get a good performance with low signal levels, but is computationally intensive [4, 18]. A radio receiver using narrowband sensing is illustrated in Figure 2 (a). This illustration contains a typical radio frequency (RF) receiver chain that includes a low noise amplifier (LNA), a signal mixer, a channel filter and a variable gain amplifier (VGA) controlled by automatic gain control (AGC).

Machine learning methods for signal detection are mostly based on cooperative spectrum sensing. In cooperative sensing multiple secondary users gather spectrum information using the classical methods mentioned

above and in Figure 1. This information is then shared among the users over a control channel and the spectrum sensing is performed using machine learning methods such as k-nearest neighbor (KNN), support vector machine (SVM) and decision trees [24, 3].

The nature of cooperative sensing limits its usefulness, because it requires multiple cooperating users and a control channel. Therefore non-cooperative methods would be preferable. Unfortunately very little research on machine learning for non-cooperative spectrum sensing is available. One paper was found that describes blind spectrum sensing using a combination of covariance-based sensing, QR matrix decomposition and support vector machines [5].

2.2 Wideband sensing

As previously mentioned, wideband sensing is the process of splitting the input spectrum into segments based on the activity levels. The corresponding signal detection problem has been called "Narrowband Signal Detection", because it involves detecting narrowband signals within the wideband signal [22].

Two classes of wideband spectrum sensing exist based on the available sampling rate: Nyquist based and sub-Nyquist based. Nyquist based sampling requires sampling the signal at least at the Nyquist sampling rate. In this case the original signal can be faithfully reproduced from the digital samples. This also allows us to perform band-pass filtering of the signal in frequency space. By using multiple band-pass filters at different frequencies, we can split the wideband signal into multiple narrowband channels that can be sensed using narrowband sensing. A radio receiver using wideband Nyquist sensing is illustrated in Figure 2 (b). The figure shows the different channels as having completely separate RF chains, however, equivalent functionality can also be achieved by using the signal processing method explained in this text.

An alternative algorithm detects the edges of the occupied segments in frequency space [2]. Another approach models the signal in frequency domain as a mixture of sinc functions. The algorithm works by iteratively adding new sinc functions to the model and tuning the existing parameters based on the residual errors [22].

Nyquist based sampling of very wideband signals requires also very high sample rates, which can be a disadvantage [18]. Therefore Sub-Nyquist based spectrum sensing has gained increasing attention. Two

methods for sub-Nyquist sensing are compressive sensing and multi-coset sampling. In compressive sensing the signal is first sampled at non-uniform intervals. Then, the signal can be reconstructed from the samples in a base where it is sparse by finding the sparsest solution of an under-determined system of linear equations. In multi-coset sampling multiple synchronized samplers sample the signal at less than the Nyquist rate [2]. A radio receiver using wideband non-Nyquist compressive sensing is illustrated in Figure 2 (c).

3 Automatic modulation classification

AMC is typically used to classify a detected signal. The AMC process is sometimes also called modulation recognition [23]. Traditional methods for modulation classification include likelihood-based (LB) and feature based (FB) methods [17]. Maximum-likelihood methods provide statistically optimal results with the cost of computational complexity [6]. Feature based methods use features extracted from the signal to perform modulation classification. The features are fed to machine learning based classifiers such as artificial neural networks (ANN), support vector machines (SVM) and decisions trees (DT) [11].

For FB methods, the suitable features vary in complexity and usefulness. These have also been extensively catalogued in the literature. Time domain features include features like amplitude and phase, but also simple statistical features like standard deviation. Transform domain features are based on signal transformations like the Fourier transform or the Wavelet transform. Statistical features refer to features calculated based on higher order statistics like moments and cumulans, cyclostationarity measures or the signal autocorrelation function. The features can also be based on the constellation shape of the signal or the number of zero crossings it has [11].

The rest of this Section will focus on the different machine learning methods used for AMC.

3.1 Artificial neural networks

ANNs consist of interconnected neurons that perform calculations. Each neuron is assigned with a set of weights that affect these calculations. The neurons calculate a weighted sum of their inputs, which is then fed to a

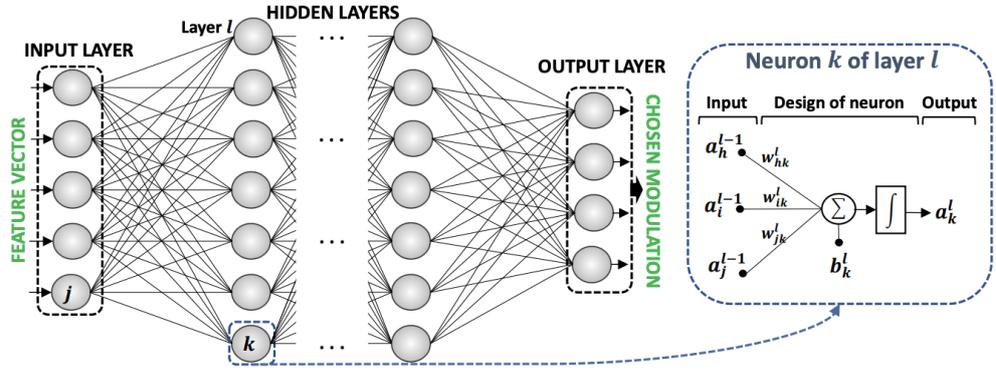


Figure 3. Structure of an artificial neural network [13].

non-linear activation function. The output of this function is the output of the neuron. The inputs and outputs are connected based on the structure of the network. The neurons are commonly structured in layers, where the outputs of layer l are connected to the inputs of layer $l + 1$. When all the neurons of layer l are connected to all the neurons of layer $l + 1$, the layer $l + 1$ is fully connected. A network of fully connected layers is known as a multilayer perceptron (MLP). The structure of an MLP is depicted in Figure 3. An MLP can be trained by updating the weights based on the errors backpropagated from the output layer [8].

Despite their seemingly simple structure, ANNs with suitable properties have been proven to be universal approximators [12]. This means that they can approximate any input function.

Jagannath et al. evaluate the performance of 1, 2 and 3 layer MLPs for AMC using a test setup consisting of two software defined radios (SDRs). Their network uses the sigmoid activation function and a softmax output layer. Five types of features are used. Seven different modulations are used. The network was trained using stochastic mini batch gradient descent using 28000 training samples. The method achieved a probability of correct classification of up to 0.98 [13].

Convolutional neural network can also be used for AMC. Zhou et al. propose an architecture with 16 convolutional layers that classifies the input signal without manual feature extraction, processing the signal directly. They achieve a classification accuracy of 0.9 for signal to noise ratio (SNR) of -10 dB. They also demonstrate how the network can perform automatic feature learning by examining the activations before the output layer [26].

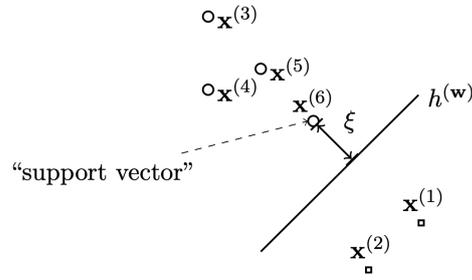


Figure 4. Hyperplane (line) separating two classes with the support vector visible [14].

3.2 Support vector machines

SVMs perform linear classification of the input data by finding a hyperplane that separates the different classes. The hyperplane is chosen in such a way that the distance between the hyperplane and the closest samples from the different classes is maximized. These closest samples are called support vectors. The relation between the support vector and the hyperplane is shown in Figure 4 [14].

As mentioned above, SVMs are linear classifiers. However, most problems are not linearly separable. This can be solved by applying a kernel function to the input data that transforms it into a new feature space in which the classes are more easy to separate [14, 23].

The simple SVM introduced above is a binary classifier, which only works with two classes. To perform multi-class classification, a "one against one" (OAO) strategy can be used. In OAO, a classifier is built for each pair of classes. Each of these classifiers casts one vote, and the class with the most votes wins [16].

Hassanpour et al. evaluate the performance of SVMs for AMC. They use a set of temporal, spectral and wavelet domain features, and 7 different modulation types. They also compare the OAO strategy with their own hierarchical classification structure. The hierarchical structure achieves similar accuracy as OAO but with lower complexity. Their evaluation gives a classification accuracy of 0.98 at SNR of -10dB [9].

3.3 Decision trees

DTs are structures that can also be used for classification. They consist of decision nodes, which correspond to tests performed on input features. The outcome of a test determines which branch in the decision tree is taken. The root node is the first test performed, and the process ends at

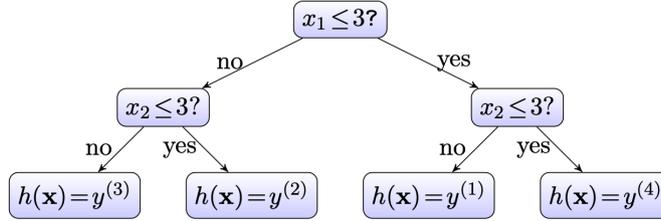


Figure 5. Decision tree classifier [14].

one of the leaf nodes that correspond to different classes. Figure 5 shows a decision tree classifier with two input features x_1 and x_2 and four output classes $y^{(1)} \dots y^{(4)}$ [14].

Subbarao et al. evaluate the performance of DTs for AMC. They use multi-order cumulant features and 6 different modulations. They achieve an accuracy of 0.82 at 0dB SNR [21].

DT classifiers can be extended by constructing a random forest classifier. Random forests consist of multiple DTs that vote on the classification result. The trees are trained using different subsets of the training set. Zhang et al. use a random forest classifier for AMC. They use entropy based features and 7 modulations. The method results in an accuracy of over 0.9 at -4dB SNR [25].

Based on the examples in this section, DTs are worse than ANNs and SVMs for AMC. However, the referenced papers use different input features, different training and validation sets, and different modulations. The chosen SNR can also affect the relative performance of the methods. The SNR of the signals that are used for training should also be taken into account. Therefore, the significance of this result is questionable. To get comparable results for the different methods, all the independent variables should be fixed, potentially requiring a separate study. Still, other research has also concluded that DTs have the worst accuracy of the pattern recognition methods used for AMC [11].

4 Conclusion

The target of this paper was to survey machine learning methods used for signal detection. We find that most existing signal detection algorithms are not based on machine learning. Additionally, most of the machine learning based signal detectors found were designed for co-operative sensing, which requires a network of receivers and a communication channel

between them. Surveys for these methods already exist. Additionally, these methods are practical in only a subset cases where regular signal detectors are used. Only one non-cooperative method was found that uses SVMs for primary user detection.

Lacking dedicated machine learning based signal detectors, we explore the related modulation classification task, hoping to be able to apply the classifiers to signal detection. Many AMC methods are machine learning based. However, we find no evidence that they are actually used for signal detection, only for classification. For the AMC task, SVMs provide especially good accuracy. We were able to find recent studies that show impressive performance of the classifiers. ANNs are also interesting, particularly because they can operate without manual feature extraction, even if the accuracy results were not as good.

We hypothesize that the AMC methods could be extended to also cover the signal detection task. One possible way would be to combine an algorithmic signal detector with a modulation classifier. The signal detector would work as a preselector, filtering out the cases where a signal is very unlikely. The modulation classifier would have a new output category, no signal, to signify that this sample was a false positive from the signal detector, and does not actually contain any signal. This should be a possible addition to the existing classifiers. Unfortunately, we were unable to find any methods that actually implemented this idea, leaving it purely hypothetical. We assume that adding a new category to the classifiers would decrease the classifier performance at low signal to noise levels. However, the size of this effect should be experimentally verified.

References

- [1] I. F. Akyildiz, B. F. Lo, and R. Balakrishnan. Cooperative spectrum sensing in cognitive radio networks: A survey. *Physical Communication*, 4(1):40–62, 2011.
- [2] A. Ali and W. Hamouda. Advances on spectrum sensing for cognitive radio networks: Theory and applications. *IEEE Communications Surveys and Tutorials*, 19(2):1277–1304, 2017.
- [3] Y. Arjoun and N. Kaabouch. A comprehensive survey on spectrum sensing in cognitive radio networks: Recent advances, new challenges, and future research directions. *Sensors (Switzerland)*, 19(1), 2019.
- [4] D. Cabric, S. M. Mishra, and R. W. Brodersen. Implementation issues in spectrum sensing for cognitive radios. In *Conference Record - Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 772–776, 2004.
- [5] Y. Chen, X. Jing, W. Liu, and J. Li. *An Improved Blind Spectrum Sensing Algorithm Based on QR Decomposition and SVM*, volume 473 of *Lecture Notes in Electrical Engineering*. 2018.
- [6] O. A. Dobre, A. Abdi, Y. Bar-Ness, and W. Su. Survey of automatic modulation classification techniques: Classical approaches and new trends. *IET Communications*, 1(2):137–156, 2007.
- [7] A. Ghasemi and E. S. Sousa. Spectrum sensing in cognitive radio networks: Requirements, challenges and design trade-offs. *IEEE Communications Magazine*, 46(4):32–39, 2008.
- [8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [9] S. Hassanpour, A. M. Pezeshk, and F. Behnia. Automatic digital modulation recognition based on novel features and support vector machine. In *Proceedings - 12th International Conference on Signal Image Technology and Internet-Based Systems, SITIS 2016*, pages 172–177, 2017.
- [10] S. Haykin, D. J. Thomson, and J. H. Reed. Spectrum sensing for cognitive radio. *Proceedings of the IEEE*, 97(5):849–877, 2009.
- [11] A. Hazza, M. Shoaib, S. A. Alshebeili, and A. Fahad. An overview of feature-based methods for digital modulation classification. In *2013 1st International Conference on Communications, Signal Processing and Their Applications, ICCSPA 2013*, 2013. Cited By :40.
- [12] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991.
- [13] J. Jagannath, N. Polosky, D. O'Connor, L. N. Theagarajan, B. Sheaffer, S. Foulke, and P. K. Varshney. Artificial neural network based automatic modulation classification over a software defined radio testbed. In *IEEE International Conference on Communications*, volume 2018-May, 2018.
- [14] Alexander Jung. Machine learning: Basic principles, 2018. <https://arxiv.org/abs/1805.05052>.

- [15] J. Ma, G. Y. Li, and B. H. Juang. Signal processing in cognitive radio. *Proceedings of the IEEE*, 97(5):805–823, 2009.
- [16] Jonathan Milgram, Mohamed Cheriet, and Robert Sabourin. “one against one” or “one against all”: Which one is better for handwriting recognition with svms? In *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft, 2006.
- [17] A. O. A. Salam, R. E. Sheriff, S. R. Al-Araji, K. Mezher, and Q. Nasir. Automatic modulation classification in cognitive radio using multiple antennas and maximum-likelihood techniques. In *Proceedings - 15th IEEE International Conference on Computer and Information Technology, CIT 2015*, pages 1–5, 2015.
- [18] H. Sun, A. Nallanathan, C. . Wang, and Y. Chen. Wideband spectrum sensing for cognitive radio networks: A survey. *IEEE Wireless Communications*, 20(2):74–81, 2013.
- [19] R. Tandra, S. M. Mishra, and A. Sahai. What is a spectrum hole and what does it take to recognize one? *Proceedings of the IEEE*, 97(5):824–848, 2009.
- [20] V. V. Veeravalli. *Fundamentals of detection theory*, pages 369–410. Mathematical Foundations for Signal Processing, Communications, and Networking. 2017.
- [21] M. Venkata Subbarao and P. Samundiswary. Automatic modulation recognition in cognitive radio receivers using multi-order cumulants and decision trees. *International Journal of Recent Technology and Engineering*, 7(4):61–69, 2018.
- [22] Curtis M. Watson. *Signal Detection and Digital Modulation Classification-based Spectrum Sensing for Cognitive Radio*. PhD thesis, Boston, MA, USA, 2013. AAI3604639.
- [23] F. Yang, B. Hao, L. Yang, and Q. Han. A method of high-precision signal recognition based on higher-order cumulants and svm. In *2018 5th International Conference on Systems and Informatics, ICSAI 2018*, pages 455–459, 2019.
- [24] T. Yücek and H. Arslan. A survey of spectrum sensing algorithms for cognitive radio applications. *IEEE Communications Surveys and Tutorials*, 11(1):116–130, 2009.
- [25] Z. Zhang, Y. Li, X. Zhu, and Y. Lin. A method for modulation recognition based on entropy features and random forest. In *Proceedings - 2017 IEEE International Conference on Software Quality, Reliability and Security Companion, QRS-C 2017*, pages 243–246, 2017.
- [26] S. Zhou, Z. Yin, Z. Wu, Y. Chen, N. Zhao, and Z. Yang. A robust modulation classification method using convolutional neural networks. *Eurasip Journal on Advances in Signal Processing*, 2019(1), 2019.

Buffer overflow: Attacks and Defences

Roope Lähetkangas

roope.lahetkangas@aalto.fi

Tutor: Thanh Bui

Abstract

Buffer overflow is a software vulnerability that occurs when writing a variable to memory, that is larger than the space allocated to it. This leads to memory locations next to that location to getting overwritten and corrupted. Attackers can leverage this memory corruption in their attacks to alter the execution flow of the program. In worst cases this leads to attacker being able to execute their malicious code on the system, and gaining escalated access privileges. This paper describes heap- and buffer overflow attacks, as well as protection methods against them.

KEYWORDS: *stack, heap, memory, C, buffer, overflow*

1 Introduction

Buffer overflow is an exploit that occurs because of lacking boundary checking. One of the first famous viruses to leverage this was a worm called "Morris" that might have affected one in every twenty computers back in 1988 [1]. Even though buffer overflow exploit has been around for years it is still common today. According to CVE details website, over 2400 new overflow vulnerabilities have been published [2]. In this paper we are going to take a detailed look into buffer overflow and different prevention methods against it.

In the first chapter we lay out the foundations by taking a look at memory structure and management. This is needed to properly understand how buffer overflow exploits occur and how different variations of the exploit differentiate from each other later on. For this we need to go over process and its structure. We are also describing how memory management is done in C and C++ programming languages to better understand how buffer overflow exploits manifest in software.

After laying down the foundation in second chapter we are diving into the exploit. Starting from the general description of buffer overflow and the varying forms that it occurs in. After the general descriptions we are examining how different methods, how buffer overflow attacks can be done.

In the last chapter we discuss about different buffer overflow prevention methods. This will include vulnerability detection and existing solutions for prevention. Lastly we are looking at emerging and researched prevention and detection methods to see new perspectives on buffer overflow prevention.

2 Memory management

Before diving into buffer overflow we must first concepts of memory and processes. To properly understand buffer overflow vulnerability a solid understanding of process memory organization and the related structures are needed. First we explain how processes are handled in memory. After that concepts of stack and heap are explained in more detail.

2.1 Processes and memory

While organization of process's memory differs depending on the operating system and the processor architecture, here we are going to go over it in general level. This makes the concepts applicable in most cases with C and C++ programming languages. Program is essentially a text file that gets loaded into computer memory. Once the program is loaded into memory it is called a process. In other words process is a program in execution. Process in memory is usually described as having three or four distinct regions of information which are called text, data, BSS and heap [3].

The region that hold the program code is called text [3]. This region is read only, its size is fixed by the program and it is located in the lowest memory address [4]. The second segment of a process memory is data. It holds global and static variables that have explicitly been initialized to some value. Next segment in process memory is called BSS (Block Started by Symbol). Like the data segment BSS holds global and static variables but only the ones that are uninitialized or initialized to zero. As the BSS and data regions are right next to each other in the memory, they are often grouped and represented as one section called data. The last region and the highest memory address is stack [3]. Stack is reserved for local variables that are assigned on runtime, passing parameters for functions and return values [4]. Stack also has other important properties and characteristics which we are going to discuss in next chapter. Heap region is also often depicted in process memory layout and it is dynamically allocated memory during runtime. In memory heap is usually located after BSS and grows upwards in memory addresses, thus residing in between stack and BSS. We will describe stack and heap in detail in the following sections.

2.2 Stack

Stack is a common abstract data type often used in computer science. Practically stack is a LIFO (Last In First Out) queue. Objects can be placed into stack with a PUSH operation and taken from a stack with POP. Due the nature of stack the last object pushed into the stack will be the first one to be removed. As mentioned in the previous section temporary variables are stored in stack. If a variable is no longer needed for example after exiting a function or subroutine, memory holding its local

variables will automatically be freed for use [4, 3].

Function calls in C and C++ programs leverage stack to modify the flow of the program execution. When a program calls a function it needs to know where to return after function or subroutine execution ends. This passing the control from one function to another, then returning the execution back to the original caller is done with stack [4]. When we start talking about buffer overflow and stack smashing in the next chapters the importance of return values will become more apparent.

2.3 Heap

Compared to stack memory which gets automatically allocated from stack when temporary values need to be stored, heap is memory that is dynamically allocated during runtime. Heap memory is used when size of a object that needs to be stored varies or the object is so large that it would be inefficient to store it into stack. It also allow more control over how long will the memory be allocated [5]. Heap management is done differently depending on the system and the allocator it is using. In this paper we describe free-list based allocation which is used in many windows and Linux systems and abstracting it to cover the basic principles [6].

In different programming languages and operating systems the functions to control the memory allocations is implemented in different ways. Most of them behave similarly. Memory allocators are often designed to work quickly and had security as after thought [6]. Heap memory consists of memory chunks that are often called "heap blocks" and metadata. Each heap block has a corresponding header block that contains the metadata. The metadata hold the information about the sizes of next and previous chunks. This allow multiple chunks that are next to each other in memory to be considered as one so that the structure does not get fragmented into a large amount of small heap blocks. Chunks and their metadata are visualized in Figure 1. As seen in the figure, the chunks are right next to each other in memory which leads heap to be susceptible to buffer overflow. Since heap memory usually grows up to higher memory addresses it is also possible to affect stack memory trough heap overflow. When a process needs memory it requests a chunk for itself and frees it when memory is no longer needed. Starting from any known chunk it is possible to move to next or previous chunk freely [5]. Information about the free memory chunks are stored in lists grouped by size. This list is searched for free memory addresses when memory gets allocated [5].



Figure 1. Fragment of the heap [7].

3 Buffer overflow

There are two main types of buffer overflows: *stack overflow* and *heap overflow*. As the names suggest, the former corrupts the stack memory, while the latter corrupts the heap memory [5]. In this chapter, we first give an overview of buffer overflow. We then describe stack- and heap corruption approaches describing what weaknesses do they exploit and how are they performed.

3.1 Overview

Buffer overflow attacks are made possible by inadequate or incomplete bounds checking while using memory [8]. What this means in practice is that it is possible to place more content in an allocated memory area than it can fit. This leads to corruption of the memory that is located after the allocated area. Writing to unallocated memory address can lead to overwriting crucial data such as the return address that hold the information where the program execution should continue after function ends [4]. Anything that causes program to write over the allocated memory area can be considered buffer overflow. Figure 2 shows a simple C program that is vulnerable to buffer overflow. Function takes string as a parameter and copies it to the buffer. The problem is that the size of the string is not checked before copying it to buffer and strcpy does no boundary checking [9]. In the following section we are specifying different methods to exploit buffer overflow vulnerabilities.

```
void func (char *string)
{
    char buffer[32];
    strcpy(buffer, string);
}
int main(int argc, char **argv)
    func(argv[1]);
    return 0;
}
```

Figure 2. Vulnerable C code [9].

3.2 Stack buffer overflow

In the past most common buffer overflow attacks have been targeted on stack and are often called stack smashing attacks [10, 11, 12]. Two main objectives in a stack based attacks are code injection and modifying the return addresses of functions. Code injection means that the attacker gives an input for the program that is executable. This could be for example some binary code or something that can natively run on the target machine. Changing the return address means that there is a function being executed and its return address is modified so that instead of returning to the typical flow of the program it returns to a memory location determined by the attacker [8].

Modification of return addresses

As we have already determined both local variables and return address both reside in stack. A process memory layout can be seen on figure 3. The attacker needs to find a function that is susceptible for buffer over-

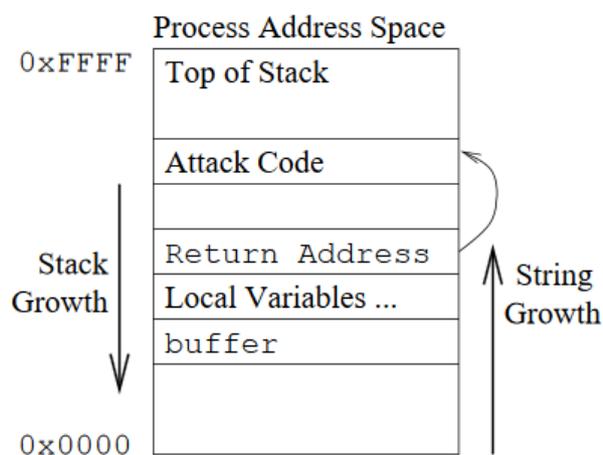


Figure 3. Process stack [8].

flow attacks. After a function has been found there needs to be a local variable where it is possible to input a greater amount of data that was originally allocated to that variable. These variables can be for example C arrays. This given large input will now occupy memory addresses in stack that were not reserved for it. Since this overwritten memory can include the return address the attacker can replace the it and control where the return address points.

Code injection

So now that the attacker can modify the flow of the program how do they execute malicious code that originally was not in the program? The answer is very similar to overwriting the return address. Attacker needs to find a variable that can exceed the memory allocated for it and input executable code in it. In Linux environments this could be a bash script or shellscript in a text format inputted to a C array. After this, we perform the modification of the return address and set it as the memory address where the executable is located. Now instead of returning from function to the normal flow of the program the program will jump to executing the injected code [9].

Programs that are usually targeted in attacks like this are running on privileged access rights. What this means is that the injected executable code is also going to run on privileged rights, thus the usual purpose of attacks are to spawn a shell that has root or similar access rights. Once a shell that has privileged rights is open the attacker can effectively perform any actions in the system with elevated access rights [8].

To successfully pull off a stack smashing attack the attacker needs to have at least some information about the original program. Finding the exact memory addresses of the return address and the part where injected code can be located may be tricky without prior knowledge of the execution flow [8]. There are some tricks that can make process of finding these memory locations easier. For example attacker can duplicate the wanted memory address where the function needs to return multiple times to increase the likelihood that the actual return address gets replaced. The injected code can also be padded with multiple NOP commands that effectively do nothing at all. The point of this is to either hit one of the NOP commands when returning from a function or the executable code. If we return to NOP the program is just going to keep running and hitting the next NOP commands and eventually end up executing the injected code. This basically increases the amount of memory addresses that will end up executing attackers code [9].

3.3 Heap overflow

Buffer overflow attacks against heap can target either the metadata or the allocated memory chunks [6]. Attacker can target a chunk that vulnerable for overflowing and overflow the data on the header of next chunk. The header data holds the information of where the next and previous

chunks resides in memory. Attacker can modify these pointers to next and previous chunks to point on wanted memory locations. Next time the memory allocator tries to use that chunk with modified pointers to adjacent chunks, instead of pointing to a next chunk it will point the program flow to attacker controlled code [5]. Attacker can also modify only the data on the next chunk, so that next time that chunk is used it executes the attackers code [5, 6].

```
#define unlink(P, BK, FD) { \
    FD = P->fd;           \
    BK = P->bk;           \
    FD->bk = BK;         \
    BK->fd = FD;         \
}
```

Figure 4. Unlink macro used by glibc [5].

To further illustrate heap overflow Figure 4 shows unlink macro used by glibc when memory is freed. P is a pointer to chunk that gets unallocated, FD and BK are variables pointing to next and previous memory chunks. If the attacker can control the P->fd and P->bk they can modify where in memory the headers point to once block has been unallocated. Because attacker can point to their own memory location they can also alter the flow of the program.

To successfully perform heap overflow attack similarly to stack overflow the program needs to have buffer overflow vulnerability. Vulnerabilities are often caused by programming error. Compared to stack overflow, executing heap overflow requires more information about the target system and its behaviour. Attacker needs to know how the heap is managed to manipulate program flow. They also need to know or guess when and what memory addresses are used, since just modifying chunk or its header is not enough. That chunk also needs to be used after it get corrupted [5].

4 Mitigation solutions

Even though individual buffer overflow attacks are most of the time quite simple to patch, new attacks happen regularly [8, 2]. One reason for this is that typically fixing a vulnerability in a program is a specific fix against specific attack for just the patched program. This is the case especially with legacy software with programming errors and vulnerable libraries. Fixes are not done systematically and in such way that solution could be expanded to other programs. In this chapter we are taking a look at

different already tried and tested solutions, as well as some researched and theorized solutions [8].

4.1 Existing solutions

Well known solutions for preventing buffer overflow can be grouped to four main categories. There is many existing implementation such as Microsoft's and Linux's gcc compilers, that use these principles [10, 13]. We are going over these principles one by one and exploring their effectiveness and weaknesses. The first method and the most self-explanatory one is to write code that has no buffer overflow vulnerability.

Writing proper code

There is a vast amount of research and good understanding how to write programs that completely negate buffer overflow vulnerabilities [14]. Yet new breaches and vulnerabilities are found frequently. Nature of the most commonly associated languages with buffer overflow such as C adds to this issue with its complexities, such as null terminated strings and many libraries that have unsafe functions. Adding to this the most vulnerable languages are often used in applications that require good performance, leaving security concerns on lower priority. This is why tools are made to do static code analysis on the written code and warn the developers, if for example unsafe function calls are made that could be replaced. Code also needs to be commonly audited and reviewed to spot programming errors and unsafe habits [10]. Programs could be developed in some other more user friendly language which takes security better into account. This would reduce the amount of vulnerabilities caused by programming errors. The problem is that many of the high-level languages rely on C and C++ in their implementation [10].

Operating system level

On the operating system level the buffer can be made to be unable to execute code. This would make injecting malicious code into programs impossible. In older systems it was fairly common to have non-executable buffers, but more modern operating systems have allowed it for performance benefits [10]. Modifying the existing operating systems to have completely non-executable buffers is not feasible since many current programs require it. Yet the stack region of the process could be made to be non-executable without requiring large rewrites. Most of currently existing software could still be compatible with operating system that cannot

execute anything from stack [10]. Even though this would be highly effective against attacks that require code injection to stack it would not prevent other types of buffer overflows such as heap based attacks.

Full boundary checking

Way to eliminate all buffer overflows is to have a complete boundary checking which is done on compiler level. Since all of the memory boundaries are checked no data structure can exceed the memory allocated to it and no buffer overflows can happen. While this would prevent overflowing from happening the implementation can be very costly for performance. Different solutions have been proposed and developed using boundary checking principle [15, 10]. These implementations have struggled to make the solution widely adopted since the performance impact is high even with optimization techniques. Compatibility with the suggested solutions has also been an issue with some solutions. [10].

Pointer integrity checking

Pointer integrity checking is similar to full boundary checking except it does not aim to prevent buffer overflows. It tries to detect if a data structure has been tampered with before using that data. Compared to boundary checking pointer integrity checking does not cover all possible buffer overflow attacks. It covers most of them with little impact on performance. Other benefits compared to boundary checking are that program compatibility is better and implementation of pointer integrity checking is easier [10]. Many existing solutions use this principle in varying ways in their implementations and studies suggest that performance and effectiveness are high. Many of them rely on a "canary" value that is placed on the allocated memory in stack. Before using allocated memory the integrity of the canary is checked. If buffer has overflowed this canary would not be intact and execution would be stopped [8, 16, 17].

4.2 Emerging solutions

In the previous section we introduced typical ways to combat buffer overflow. These solutions are widely known and at least their principles have been implemented in existing systems. In this section we explore some newer and theorized solutions.

Machine learning

With the new leaps in hardware and software machine learning has

risen in popularity. New machine learning approaches have been suggested to detect vulnerabilities in software. Many machine learning solutions leverage symbolic execution, which basically allows program to take undetermined values as input to functions. This allows machine learning algorithms to explore all execution paths. Proposed solutions can be used to detect vulnerable components and or even patterns in the code that might be vulnerable [18, 19]. Machine learning methods have also been used to learn normal system behaviour and detect changes in it. Detecting intrusion early can give users time to react properly. Intrusion detection also help with detecting new attack methods that the attackers are using[20]. In general machine learning tools that perform static analysis on the code and perform symbolic execution show promising result. In the best cases vulnerability detection can be over 90 percent [18, 19].

Hardware solutions

Many hardware based solutions have been developed to prevent buffer overflows from occurring [21, 22, 23]. Implementation between them varies, but commonly they employ the same principles as compiler and operating system based solutions [21, 22]. Using hardware based solution for example to keep function return addresses from being modified solves the performance issues that are often present in typical solutions. As a drawback hardware solutions usually require modifications to the operating system or hardware itself, possibly even both [21].

5 Conclusion

In this paper we explained buffer overflow and concepts required to understand it. First we described how memory management is done and how process is structured. This is required to properly understand what makes different buffer overflow attacks possible. The concepts were abstracted to a level that it can cover many different systems, and be understandable for wide audience.

Second chapter describes buffer overflow vulnerabilities. First in general what is buffer overflow and what makes it possible. Then diving deeper on two main types of buffer overflow attacks explaining how are they performed and what can be achieved with them.

Lastly we took a look at different methods to prevent buffer overflow. We described four main principles on buffer overflow protection and ex-

plored their strengths and weaknesses, as well as why they prevent overflows from happening. Finally we examined some emerging, interesting solutions to counter buffer overflow attack and do they apply new ideas combined with the four main principles.

In conclusion we see that buffer overflow is still a serious vulnerability with no universal solution. A lot of research has been done on buffer overflow and protection against it. Many proposed solutions have been made, and the principles learned from research have been applied to existing products. Development and research of new solutions continues to ensure that future software performs well and securely.

References

- [1] H. Orman, "The morris worm: a fifteen-year perspective," *IEEE Security Privacy*, vol. 99, pp. 35–43, Sep. 2003.
- [2] "Security vulnerabilities published in 2018(overflow)." <https://www.cvedetails.com/vulnerability-list/year-2018/opov-1/overflow.html>. Accessed: 2019-04-03.
- [3] P. Van der Linden, *Expert C programming: deep C secrets*. Prentice Hall Professional, 1994.
- [4] A. One, "Smashing the stack for fun and profit," *Phrack magazine*, vol. 7, no. 49, pp. 14–16, 1996.
- [5] W. K. Robertson, C. Kruegel, D. Mutz, and F. Valeur, "Run-time detection of heap-based overflows.," in *LISA*, vol. 3, pp. 51–60, 2003.
- [6] G. Novark and E. D. Berger, "Dieharder: securing the heap," in *Proceedings of the 17th ACM conference on Computer and communications security*, pp. 573–584, ACM, 2010.
- [7] E. D. Berger, "Heapshield: Library-based heap overflow protection for free," *UMass CS TR*, pp. 06–28, 2006.
- [8] C. Cowan, C. Pu, D. Maier, J. Walpole, P. Bakke, S. Beattie, A. Grier, P. Wagle, Q. Zhang, and H. Hinton, "Stackguard: Automatic adaptive detection and prevention of buffer-overflow attacks.," in *USENIX Security Symposium*, vol. 98, pp. 63–78, San Antonio, TX, 1998.
- [9] K.-S. Lhee and S. J. Chapin, "Buffer overflow and format string overflow vulnerabilities," *Software: practice and experience*, vol. 33, no. 5, pp. 423–460, 2003.
- [10] C. Cowan, F. Wagle, C. Pu, S. Beattie, and J. Walpole, "Buffer overflows: Attacks and defenses for the vulnerability of the decade," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2, pp. 119–129, IEEE, 2000.

- [11] D. Moore, C. Shannon, *et al.*, “Code-red: a case study on the spread and victims of an internet worm,” in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement*, pp. 273–284, ACM, 2002.
- [12] Y.-J. Park and G. Lee, “Repairing return address stack for buffer overflow protection,” in *Proceedings of the 1st conference on Computing frontiers*, pp. 335–342, ACM, 2004.
- [13] P. Silberman and R. Johnson, “A comparison of buffer overflow prevention implementations and weaknesses,” *IDEFENSE*, August, 2004.
- [14] M. Bishop, “How to write a setuid program,” 1987.
- [15] R. Hastings and B. Joyce, “Purify: Fast detection of memory leaks and access errors,” in *In proc. of the winter 1992 usenix conference*, Citeseer, 1991.
- [16] C. Cowan, S. Beattie, J. Johansen, and P. Wagle, “Pointguard tm: protecting pointers from buffer overflow vulnerabilities,” in *Proceedings of the 12th conference on USENIX Security Symposium*, vol. 12, pp. 91–104, 2003.
- [17] A. Baratloo, N. Singh, and T. Tsai, “Libsafe: Protecting critical elements of stacks,” *White Paper* <http://www.research.avayalabs.com/project/libsafe>, 1999.
- [18] Q. Meng, , and and, “Predicting buffer overflow using semi-supervised learning,” in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1959–1963, Oct 2016.
- [19] B. M. Padmanabhuni and H. B. K. Tan, “Predicting buffer overflow vulnerabilities through mining light-weight static code attributes,” in *2014 IEEE International Symposium on Software Reliability Engineering Workshops*, pp. 317–322, Nov 2014.
- [20] J. Singh and M. J. Nene, “A survey on machine learning techniques for intrusion detection systems,” *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 11, pp. 4349–4355, 2013.
- [21] J.-L. Danger, S. Guilley, T. Porteboeuf, F. Praden, and M. Timbert, “Hardware-enforced protection against buffer overflow using masked program counter,” in *The New Codebreakers*, pp. 439–454, Springer, 2016.
- [22] Z. Shao, C. Xue, Q. Zhuge, M. Qiu, B. Xiao, and E.-M. Sha, “Security protection and checking for embedded system integration against buffer overflow attacks via hardware/software,” *IEEE Transactions on Computers*, vol. 55, no. 4, pp. 443–453, 2006.
- [23] N. Tuck, B. Calder, and G. Varghese, “Hardware and binary modification support for code pointer protection from buffer overflow,” in *Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*, pp. 209–220, IEEE Computer Society, 2004.

Barcodes for mobile computing applications

Tanvi Jain

`tanvi.jain@aalto.fi`

Tutor: Mario Di Francesco

Abstract

2D barcodes are ubiquitously used for tagging information to products, making mobile payments and sharing digital data. The encoding process requires basic code generator techniques, while the decoding process requires smart phones, equipped with digital cameras and scanner software. Although they are popular medium for sending information to an interested user, but their black and white appearance is unpleasant to human eyes. Much research has been conducted in order to make these codes beautiful and human-readable. The possibility of human-readability will increase the relevance of these codes for masses. The idea is that the visual channel will carry both human-readable and machine-readable content in the same source. This will involve a considerable amount of computing at the sender as well as receiver end. The process of encoding data into images at sender's end motivates to understand various approaches involving various computing techniques related to barcodes.

KEYWORDS: *2D-barcode, Quick Response codes, aesthetic QR codes, QR codes, Data hiding,*

1 Introduction

Mobile phones contribute towards a half of the total web traffic in the world. According to Ericsson's report [1], in the year 2017, the worldwide mobile data traffic reached nearly 15 exabytes (1 exabyte = 1000000 terabyte) per month, and it is growing by at least 50 percent annually since 2013. This data traffic exploits much of the Radio and microwave frequencies, which has led to much research in the analysis of the use of visible spectrum. The visible spectrum communication offers great theoretical bandwidth potential [5]. For example, 2D-barcodes which exploits the visual spectrum for communication, have gained popularity in the business promotion connecting the offline and online contents.

Two dimensional (2D) barcode is a black and white matrix which encodes various types of data such as promotional web URL link, payment detail link. It facilitates the customer with direct access to the content without actually typing the entire text. On the receiving end, the mobile phone equipped with computational power and imaging capability, captures and decodes the 2D barcode. This process is simple and widely used because it involves a barcode generating software at service provider's end, and a mobile software as scanner/decoder at customer's end.

However, the traditional 2D barcodes lack the visual appeal. They fail to communicate with the user because of their black and white, mechanical and unfriendly appearance. The coded information cannot be understood without appropriate scanners. These codes lack the desirable attributes such as human readability and aesthetics. Moreover, it is important for a user to know the nature of the scan, before that user points the phone camera to scan it. Data matrix and QR codes are some of the examples of 2D barcodes, which are extensively in use.

Due to the inherent error correcting capability of QR codes, it is possible to add human readable content without compromising on the scanner readability. Quick Response codes, popularly known as the QR codes, were first developed by a Japanese company called Denso Wave in 1994 for the automotive industry. QR codes offered quick readability, accuracy, lower cost and data carrying capacity, which made them famous outside the automotive industry. The smart phone industry contributes to the immense penetration of QR codes in people's lives. Figure 1. depicts a QR code which is both artistic and machine-readable.

Another implementation of visible spectrum communication is the trans-



Figure 1. An example of artistic QR code from Wikipedia

fer of digital data such as images, text or videos with some secret data embedded into them. Data hiding is one such technique. Various ways to hide data in images are established [20, 6, 17, 18, 2, 8]. The data hiding image carries the human-readable information, such as artistic images and logos, along with the machine-readable coded data bits, together in the same source over the visual channel [20].

However, little work has been conducted in order to review and compare these techniques with each other. This paper presents a review of the aesthetic QR code formulation techniques as well as data hiding techniques. The aim is to present a broad view which showcases the various aspects and challenges of generating artistic as well as robust visible light communication channels.

In section 2, the structural details of QR code are presented. In section 3, a brief overview of embedding QR code is presented. Section 4 discusses various computing techniques to generate beautiful QR code messages. Section 5 lists few data hiding techniques. In section 6, a brief conclusion is provided.

2 What is QR Code?

The traditional QR code is a pattern of modifiable black and white blocks, which carry data or error correction code. The original message bits are encoded and arranged into data area by applying EX-OR function on data area cells and mask pattern cells. The EX-OR operation is carried out eight times with eight different mask patterns. The result from operation

is assessed for each mask pattern, and the one with highest match is selected.

Along with the modifiable data blocks, QR code comprises of some fixed large blocks which guarantee detection and decoding robustness. The fixed large blocks or patterns are called *finder patterns*, *alignment patterns* and *timing patterns*. The high capacity QR code needs more *fixed patterns* in its interior area. When the capacity increases, the module alignment accuracy becomes more critical and then the numerous *fixed patterns* are utilized to improve its alignment. [8, 10].

The *version number* V determines the size of a QR code, where $1 \leq V \leq 40$. The size increases by a factor of four cells in both directions, for example, 21×21 , 25×25 ,... till 177×177 cells. QR code utilizes error correcting capabilities of Reed Solomon codes, where the codewords appear in consecutive modules. L, M, Q, and H are 4 error correction levels. They recover 7%(L), 15%(M), 25%(Q) and 30%(H) error codewords of the whole QR code [14].

A suitable small version number and error correction level is selected for information accommodation. This selection is based on the coding capability of the data information. The data modes can be numeric, alphanumeric, 8 bit byte and kanji or kana. The Reed Solomon encoding algorithm encodes the data into QR code. For example, in Figure 2, a QR code (version 3, error correction level: L) uses an RS codes (70, 55, 7), which comprises 55 data code words (D1-D55) and 15 error correction code words (E1-E15). Since the error correction level is L, 7 pieces of data can be corrected [14].

3 Picture embedded QR Codes

The traditional QR codes, when incorporated with explicit features such as, faces, letters or logos, generate visually acceptable codes. However, multiple challenges render the embellishment process, complex. The main challenge is to maintain the scanning robustness of the resulting QR code. The scanning robustness is the property of the QR code, which enables easy and correct scanning by universal scanners [19]. The stability of a QR code is affected by the ever-changing conditions, such as lighting, camera specifications, and even perturbation in the QR codes. Poorly documented algorithms which are used to scan these codes, result in more challenges [9]. Much research has been conducted in recent years on

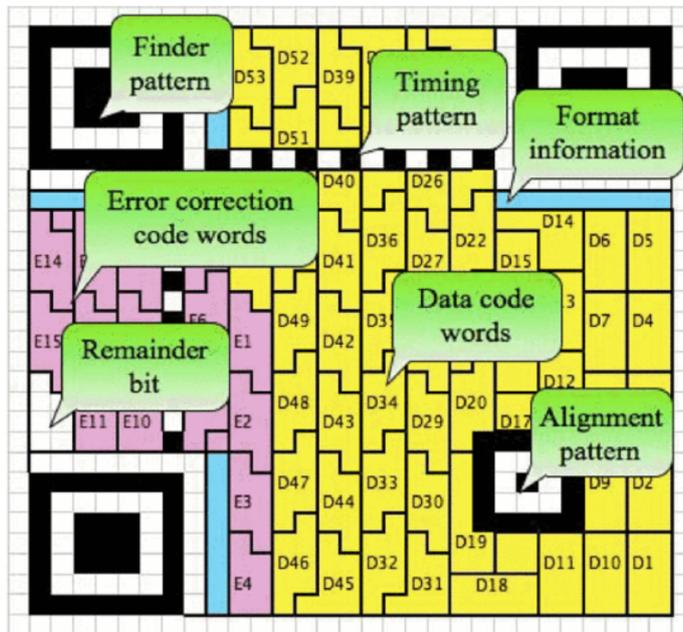


Figure 2. QR code encoding [16] © 2011 IEEE

adding visual aesthetics to the QR codes and making them user-friendly [13]. Various techniques have been devised to beautify the codes by trying and balancing the trade-off between readability and visual quality. The fundamental techniques or algorithms for generating aesthetic codes can be classified into five groups: embedding icons [16, 15, 14, 9], replacing colors [3], changing the module shape [13], adjusting codewords [10] and data hiding [20, 8]. Each technique has its own merits and drawbacks. The paper focuses on processes which involve embedding of data in images and data hiding techniques.

4 Aesthetic QR codes and Embedding Techniques

Few of the most common approaches used for implementing the process of embedding images or icons in QR code are :

4.1 Direct replacement

In this technique, the error correction capabilities of barcode handles the incurred error due to replacement [15, 16]. This method is less popular because the decodability of the QR code relies on the careful selection of embedding region. Samretwit and Wakahara[16] experimented with the size and locations of superimposed image within the QR code. They changed the image size as well as the image position in order to measure

the change in reading performance. They then measured the reading performance of the code for every position and concluded that the center-left position shows the highest reading capability characteristic. Therefore, it is the best position for superimposing the image. In another approach,

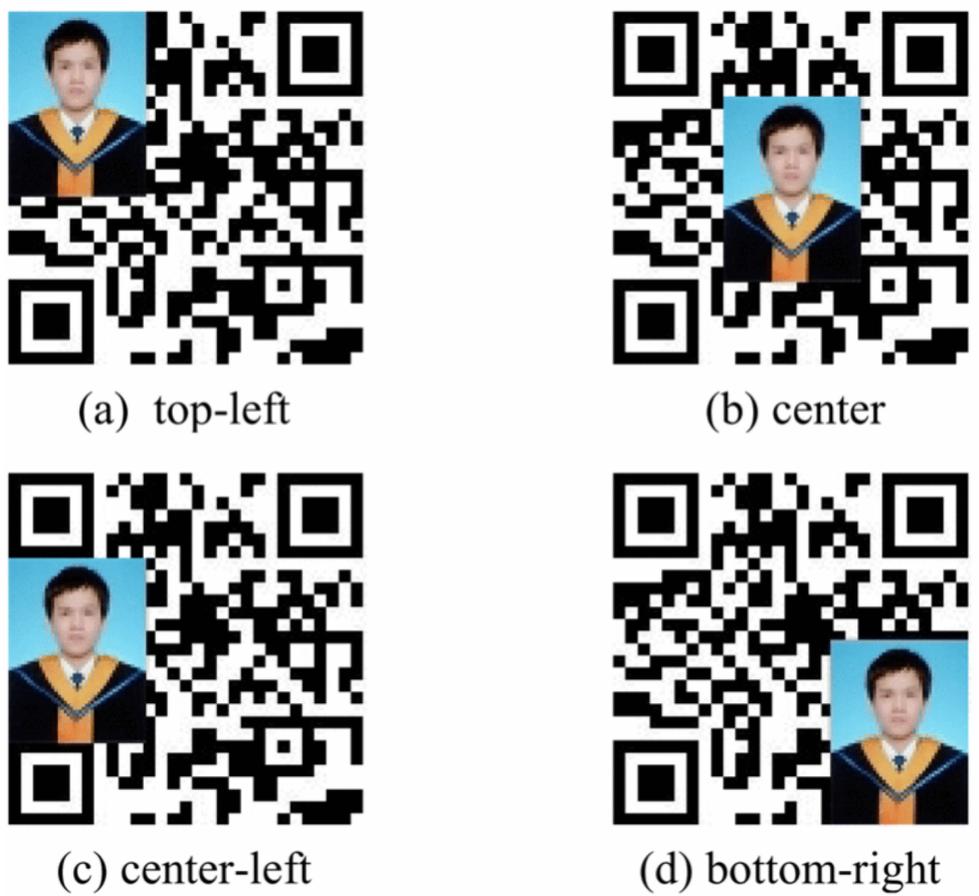


Figure 3. The placement of image at various positions to study the reading performance [16] © 2011 IEEE

Ono et al. formulated an optimization method to generate embellished QR codes. The challenge is to find appropriate position for superimposing illustrations on the QR code. Ono et al. used real-coded genetic algorithm (GA), which is a well-known meta-heuristics for optimization problem. For each illustration i , the position (x_i and y_i axis positions), angle (θ_i), scale (s_i) are arranged as genotypic representation. The phenotypic representation are generated by placing the images in QR code according to the sequence of genotypic representation. The optimization problem to determine the best suitable position is then solved by genetic algorithm. The method requires two or more decoders to correctly decode the decorated QR code, and the result is then again fed to the GA, for further optimization. The algorithm stops when fitness of the best solution reaches 1.0, or the number of generations reaches the predefined limit. This method

is time consuming and is incapable of producing QR codes with high aesthetic value. [15]

4.2 Replacing the pixels of modules

This approach involves replacing the pixels of outer portions of each module with the embedding image, and keeping the central portion unharmed. It overcomes the image visual distortion caused by the direct replacement method. The embedding of image increases the overall aesthetic quality of the QR code.

Digital Halftoning is a dominant technique used for creating embedded QR codes. It is a process of converting "continuous-tone" image into an image formed by dots. The image maintains a gradient-like effect by manipulating the size and shapes of the dots. The technique derives itself from traditional halftone approach which is used commonly for printing images in newspapers and similar print media. *Patterning, dithering, error diffusion* are three common methods of Digital Halftoning [4].

Halftone QR codes

The system converts the object image into halftone image using error diffusion halftoning. It then applies an image filter to generate an importance map which highlights salient image features. The "selected" QR code data modules (denoted by M) which will be covered by object image, are divided into submodules of 3×3 dimension. The centre submodule is bound by the color of the module, while leaving remaining eight submodules independent to change their appearance. These submodules can also be called binary patterns. These patterns are substituted by patterns from a data set denoted by P , such that the *pattern reliability and regularization* are balanced. The patterns are assigned using an optimization technique, maintaining readability throughout the pattern substitution [9]. The *pattern reliability* is evaluated by creating a synthetic QR code database which is generated using same encoding libraries previously used for generating QR codes and random strings. Along with that, spatial perturbations are applied. These spatial perturbations account for various scenarios when scanning a code in reality. Then, barcode reader decodes each synthetic QR code and statistically model the pattern reliability. Pattern reliability is a normalized ratio of number of successful decodes among all samples in the database. This process resulted in a database of 0.6 million synthetic QR code with estimated reliability. Pat-

tern regularization aims at maintaining and controlling the appearance of module in a QR code via the target halftone image using a similarity distance metric. The pattern reliability (to maximize the readability of module) and regularization (to control the appearance of the module) are the two factors which govern the pattern assignment in halftone QR codes [9].

The halftone QR codes is limited by the traditional halftoning technique which may result in poor quality images due to the low image resolution and contrast. Increasing image resolution will exponentially increase the number of binary patterns and will render pattern reliability evaluation and pattern assignment optimization intractable [9].

QR Images

QR Images [11] overcome some of the limitations of other embedding methods. QR Images reduces probability of error detection in order to increase decodability by any of the standard decoder applications. They cover almost the entire area of the QR code with embedding image, which increases the aesthetic beauty. QR Image encoding is a two part process where at first the modified pixels are selected using the digital halftoning techniques (blue or green noise masking) and then an optimization approach is used for modification of the luminance levels to minimize distortion. The pixel selection process creates two groups of pixels: binary pattern generated by halftoning mask denoted by I_{pc} , and a mask that is set on 1 for central pixels and 0 otherwise denoted by M . In the second step, the luminance of the selected pixels are modified to one of the four possible values $\alpha, \beta, \alpha_c, \beta_c$. The levels α_c, β_c are assigned to central pixels (M) and α, β are assigned to adjacent pixels I_{pc} while the other pixels of the QR code remain unchanged. Only the pixels at the center of the QR module are relevant for a correct decoding, therefore, different luminance levels are assigned to central and adjacent pixels, which helps in detection of binary values. After adjusting the luminance, the color vector for each modified luminance is selected by measuring the color differences into a perceptually uniform color space, *HSL*. The RGB values of a modified pixel (where target luminance = l_t) are obtained by transforming the original color into the HSL color space and then the L component is optimized while keeping S and H fixed until the desired luminance (l_t) is achieved [11].

The combination of pixels from embedding image and QR code mod-

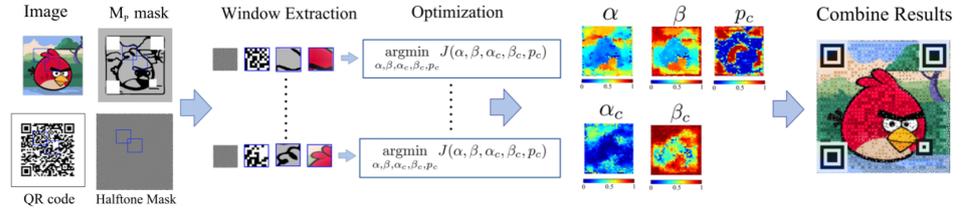


Figure 4. The step wise procedure to convert the original input image, the QR code, and the halftone mask into final QR Image [11] © 2014 IEEE

ules results in distortion of binarization threshold which results in higher probability of binarization error. In order to overcome this error and increase the decodability robustness, two probability error models (Probability of Binarization Error and Probability of Decoding Error) are constructed and combined into one Global Probability of Error. Along with determining error probability for better decodability, it is necessary to define metrics to quantify the visual quality of the QR Images. The metric used for QR Images to measure image similarity is based on SSIM index. Structural similarity (SSIM) index is a metric which qualifies and quantifies the similarity of two images. It explicitly considers factors such as luminance distortion, contrast distortion and loss of correlation in structure measures [12]. This metric considers both the halftone visibility and image structure. Therefore, the QR blocks will be less visible from a distance, but when viewed from closer distances, the important details will be visible [11].

5 Aesthetic Image codes and Data hiding Techniques

In order to increase the aesthetic beauty and data communication accuracy of coded images, this report explores another technique called data hiding.

ARTcode

ARTcode (Adaptive Robust doT matrix barcode) [20] is a beautified code, which allows the encoding of data bits directly onto the embedding image, instead of first converting it into QR code. Thus, removing the aesthetically unpleasant black and white blocks of QR code from resulting image. The encoding involves creation of colored dot matrix from the original image, applying shuffling algorithm and finally embedding data as well as color bits. The image is converted into colored dot matrix structure,

while maintaining its original general appearance. This is a two phase process, in phase I, a color palette of n colors (clusters) is generated by using K-means clustering. Using the original image as input, at first the K-means clustering sets n random RGB space vectors (3-dimensional) as first centres. From these centres, an algorithm finds a nearest center for each color and assigns that color a corresponding cluster. The K-means clustering converges after several iterations. Although the resulting color palette holds fewer colors than the original image, but they are capable of displaying the original image in a best possible way.

In phase II, the colored dot matrix is generated using the error diffusion dithering technique and the color palette obtained from phase I. *Dithering*, another digital halftoning technique, is the process of creating an output image with the same number of dots as the number of pixels in the source image. In dithering, the source image is thresholded with a dither matrix [4]. The dithering (error diffusion) technique quantifies the RGB value of each module with the color from palette, and diffuses the quantification error to the neighbouring modules. This technique is different from the traditional dithering technique which involves colors of equidifferent RGB values. Due to the use of colorful art images for encoding, the colors of equidifferent RGB values will be of no use to ARTcode [20].

This is followed by process of encoding n colors selection. The algorithm manipulates the colors in color palette such that the Euclidean distance in RGB space for any two encoding colors is larger than the threshold value, this helps in easy recognition by scanning devices. The algorithm also calculates and compares the sum of colors' proportion and select colors with largest proportion. These two steps ensure the decoding accuracy and visual effect preservation. [20].

The next step is the application of shuffling algorithm which spreads the encoding modules over the colored dot matrix according to a shuffling table. In order to preserve the image from distortion, the data is not encoded over the entire dot matrix but over limited encoding modules given by shuffling table. The embedding process will only consider the encoding modules for embedment. The shuffling algorithm is an advantage over other contemporary techniques. The shuffling table helps in encoding several individual codes in one single ARTcode. The table can be sent to the receiver and based on the received table, the receiver can decode the ARTcode, leaving the other portions undisclosed [20].

The embedding process divides the encoding modules obtained after shuffling into unit blocks and uses position information of modules in a block to encode data. Each module position is assigned a weight. In order to embed data of more than 1 bit, it is required to change just 1 module. This approach is derived from the process described by Chang et al. in [7].

The ARTcodes offers better accuracy and visual quality, however they are limited by the fact that they can be used only for small-volume data encoding such as URL.

PiCode

PiCode [8] is a picture embedding 2-dimensional barcode, which is an important implementation of data hiding technique. It offers three advantages over previously described methods. First, PiCode uses odd number of modules in vertical and horizontal direction, which helps in improving accurate detection of corners. The detection takes place with the help of fine-corner detection algorithm proposed by Chen et al. in the paper. Second, high capacity QR codes usually have higher number of fixed patterns in the interior regions which help in module alignment. PiCode does not depend on these extra fixed patterns for module alignment during high capacity. Third, the use of adaptive modulation over binary modulation helps in distortion reduction.

The encoding process is divided into two parts. In the first part, the message is converted into stream of bits. It is processed into bits by using source and channel encoding to increase the efficiency and robustness. The Reed-Solomon code is used as a type of channel encoding in PiCode. The code rate is directly proportional to message length, which maximizes the error correction capability for all message lengths.

In the second part, the message bits (0 or 1) are modulated to maintain the decoding robustness as well as perceptual quality. In order to minimize the perceptual difference between the colors of original and embedded image, the modulation is performed in YUV color space. The modulation scheme divides the modules into pixel blocks (image blocks), each containing $k \times k$ pixels, where k depends upon the embedded image resolution. Each image block is assigned either '0' or '1' depending on the intensity of outer and inner pixels. In the end, finder pattern is added to the image block which generates PiCode.

6 Conclusion

The study helped in understanding various prevalent computing techniques for beautiful barcode generation. The halftone dithering techniques along with the optimization techniques are used predominantly in various approaches. Digital Halftoning is one of the most common method to prepare an image for data embedding. The results from various methods, discussed in section 4, show that it is difficult to achieve all the desired features from a single method. Some methods are time intensive, while others are limited by the data capacity. The SSIM index is commonly used as quality metrics in QR codes. The quality metrics are used to quantify the visual quality of the resulting image. These are some of the important techniques for embedding data in images. The scope of this study was limited to image embedding. However, much research is being conducted on use of visual light communication to transfer imperceptible data over a video [5]. These forms a basis for future review reports, in this field of visual light communication.

References

- [1] Ericsson mobility report, June 2018. <https://www.ericsson.com/assets/local/mobility-report/documents/2018/ericsson-mobility-report-june-2018.pdf> [Accessed: 2019-02-28].
- [2] Satoshi Abe, Takefumi Hiraki, Shogo Fukushima, and Takeshi Naemura. Screen-camera communication via matrix barcode utilizing imperceptible color vibration. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings, UIST '18 Adjunct*, pages 166–168, New York, NY, USA, 2018. ACM.
- [3] Simon Ternoir Gautier Chapuis Alexis Laporte, Benoît Reulier and Romain Kassel. Unitag. <https://www.unitag.io> [Accessed: 2019-02-23].
- [4] D. Anastassiou and K. S. Pennington. Digital halftoning of images. *IBM Journal of Research and Development*, 26(6):687–697, Nov 1982.
- [5] Roberto Carvalho, Ching hsiang Chu, and Ling jyh Chen. Ivc: Imperceptible video communication, 2014.
- [6] Chi-Kwong Chan and L.M. Cheng. Hiding data in images by simple lsb substitution. *Pattern Recognition*, 37(3):469 – 474, 2004.
- [7] Chin-Chen Chang, Chun-Sen Tseng, and Chia-Chen Lin. Hiding data in binary images. In Robert H. Deng, Feng Bao, HweeHwa Pang, and Jianying Zhou, editors, *Information Security Practice and Experience*, pages 338–349, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

- [8] C. Chen, W. Huang, B. Zhou, C. Liu, and W. H. Mow. Picode: A new picture-embedding 2d barcode. *IEEE Transactions on Image Processing*, 25(8):3444–3458, Aug 2016.
- [9] Hung-Kuo Chu, Chia-Sheng Chang, Ruen-Rone Lee, and Niloy J. Mitra. Halftone qr codes. *ACM Trans. Graph.*, 32(6):217:1–217:8, November 2013.
- [10] Russ Cox. Qart. <https://research.swtch.com/qart> [Accessed: 2019-02-23].
- [11] G. J. Garateguy, G. R. Arce, D. L. Lau, and O. P. Villarreal. Qr images: Optimized image embedding in qr codes. *IEEE Transactions on Image Processing*, 23(7):2842–2853, July 2014.
- [12] A. Hore and D. Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, Aug 2010.
- [13] Li Li, Jinxia Qiu, Jianfeng Lu, and Chin-Chen Chang. An aesthetic qr code solution based on error correction mechanism. *Journal of Systems and Software*, 116:85 – 94, 2016.
- [14] Y. Lin, Y. Chang, and J. Wu. Appearance-based qr code beautifier. *IEEE Transactions on Multimedia*, 15(8):2198–2207, Dec 2013.
- [15] Satoshi Ono, Kensuke Morinaga, and Shigeru Nakayama. Two-dimensional barcode decoration based on real-coded genetic algorithm. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 1068–1073, June 2008.
- [16] D. Samretwit and T. Wakahara. Measurement of reading characteristics of multiplexed image in qr code. In *2011 Third International Conference on Intelligent Networking and Collaborative Systems*, pages 552–557, Nov 2011.
- [17] Yu-Chee Tseng, Yu-Yuan Chen, and Hsiang-Kuang Pan. A secure data hiding scheme for binary images. *IEEE Transactions on Communications*, 50(8):1227–1231, Aug 2002.
- [18] Min Wu and Bede Liu. Data hiding in image and video .i. fundamental issues and solutions. *Trans. Img. Proc.*, 12(6):685–695, June 2003.
- [19] Mingliang Xu, Qingfeng Li, Jianwei Niu, Xiting Liu, Weiwei Xu, Pei Lv, and Bing Zhou. Art-up: A novel method for generating scanning-robust aesthetic qr codes. *CoRR*, abs/1803.02280, 2018.
- [20] Zhe Yang, Yuting Bao, Chuhao Luo, Xingya Zhao, Siyu Zhu, Chunyi Peng, Yunxin Liu, and Xinbing Wang. Artcode: Preserve art and code in any image. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '16*, pages 904–915, New York, NY, USA, 2016. ACM.

Machine learning based rate adaptation HTTP video streaming

Noah Nettey

noah.nettey@aalto.fi

Tutor: Gazi Illahi

Abstract

This paper will gather some of the proposals how to use machine learning together with HTTP adaptive video streaming. The methods aim to increase the quality of experience for the end-user. The methods and algorithms use different approaches how to select the bit-rate to be received from the server. Some methods utilise already existing technology such as Microsoft's IIS Smooth Streaming and some don't, such as Tiyuntsong. Some methods aim to improve the selection of features and some present new approaches using neural networks. The methods considered consist of the following machine learning approaches: linear regression, reinforcement learning using Q-learning and reinforcement learning together with neural networks.

KEYWORDS: machine learning, HTTP adaptive streaming, Microsoft IIS Smooth Steaming, reinforcement learning, QoE, linear regression, neural networks, GAN, adaptive bit-rate,

1 Introduction

In this paper we will examine some solutions to tackle the problem of rate adaptation in HTTP video streaming using machine learning approaches. Especially we will concentrate on the quality of experience (QoE) aspect, when comparing different alternatives.

The goal of this paper is to gather information about the different techniques, compare them between each other, but also consider the advantages and disadvantages compared to the conventional approach. Furthermore, in this paper we will not discuss the server side solutions, but only the aspect of the clients ability to request the most suitable video quality to maximise the QoE.

The rest of the paper is organised as follows, first we examine a bit the HTTP Adaptive Streaming (HAS) and how it works. Then we will go through a few proposals how we can take advantage of machine learning in HAS. Thirdly we will make some comparisons between the conventional approaches and the different machine learning methods. Lastly we will conclude everything together.

2 Background

At the moment majority of video streaming is done through rather simple system on the client side. The currently prevailing standard for video streaming is HTTP Adaptive Streaming (HAS), used by many large companies such as Netflix and YouTube. The basic idea behind HAS is that the server has the video encoded in multiple qualities and bit-rates. Then, through a manifest file sent from the server to the client, the client is made aware of the qualities available and the manifest file is kept up to date during the streaming. Client then requests the appropriate quality of the stream [2]. Furthermore the video is split into chunks, so the client can request different quality chunk if the network conditions change, and thus adapt to the situation. This way the client can also control the buffer itself [1]. It is important to note that in HAS, the client decides which quality or bit-rate is going to be received. The network conditions such as bandwidth may change drastically during the streaming, especially on mobile devices.

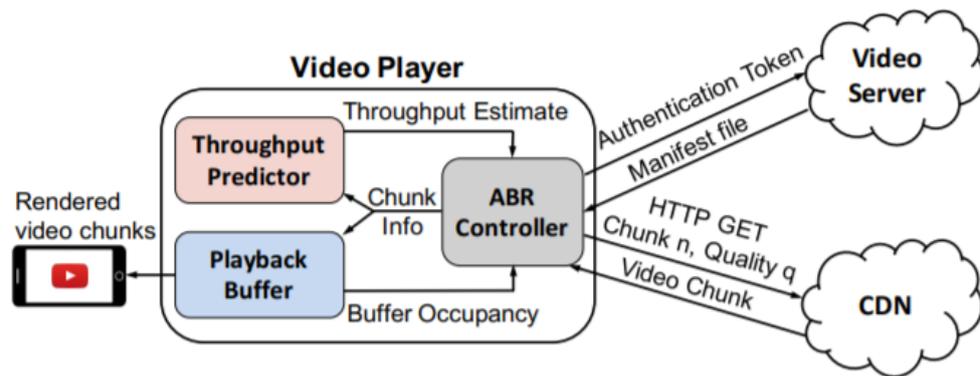


Figure 1. Overview of HAS architecture [4]

The Quality of Experience can be measured through many different measures such as freezes during the stream, buffering time before stream starts, quality of the stream and changes in the quality during the stream. The QoE follows from the Quality of Service measures such as available bandwidth, the current buffer size, the bit-rate of the latest chunk or the average bit-rate of the stream.

Currently the algorithms on which the client relies on, when deciding the quality to be received, are relatively simple compared to the potential for example through machine learning. Such approaches and solutions have been introduced increasingly in recent years. In this paper we will explore some of them, especially from the perspective of QoE.

3 Machine learning and HAS

During recent years several different solutions and approaches have been proposed how to use machine learning in HTTP adaptive video streaming. Some popular approaches have been to use some machine learning algorithm to train a classifier, and then use it to determine the requested video chunk quality. Other popular approach is to use some reinforcement learning method to continuously improve the performance of the methods. The metrics or features that are chosen for these algorithms have a big effect on their performance under different situations. In this section we will discuss the different suggested implementations varying from linear regression to more complex reinforcement learning algorithms.

3.1 Q-learning with traditional heuristic functions

The simplified idea behind reinforcement learning (RL) is to maximise some reward function with an explicit goal based on the current conditions. This is done by both repeating the already taken actions which produced high rewards, but also exploring new actions to produce higher rewards. The key is to find the balance between these two, which often is not that easily done [10]. Typically the RL model is done through Markov Decision Process considering: a finite set of states, a finite set of actions, state transition probability matrix and a reward function [11].

Q-learning is a RL algorithm in which a Q-value is calculated for each state-action pair. This Q-value represents the profitability of taking a certain action in a certain state. Furthermore these values are updated every time an action is taken. The Q-values are defined as

$$Q(s, a) = (1 - \alpha)Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a')] \quad (1)$$

The α is how well we learn from new information and γ represents how important the future rewards are [11].

In this reinforcement learning approach introduced by Hooft, Petrangeli, Claves, Famaey and Turck [11] the idea is to use Q-learning approach with traditional heuristic functions MSS and QoE-RAHAS[7], the former is the Microsoft's IIS Smooth Streaming rate adaptation function. The latter is by Petragneli et al. which aims to maximise the QoE. The states in the RL algorithm are defined through the average available bandwidth and the mean absolute difference in bandwidth. The latter is meant to be able to separate between varying and stable network conditions. The reward function is defined as

$$r = \psi \text{MOS}_w - (1 - \psi)(\text{buffer}_w) - c \quad (2)$$

In this proposed approach the MOS_w is the Mean Opinion Score and buffer_w is the average buffer filling state. Both metrics are considered in the same time window w . On the other hand ψ represent the trade-off between the aforementioned [11]. Furthermore "A value close to 1 means that the only goal of the client is to pursue a high QoE, while a value close to 0 means that the client is driven to select the parameter configuration leading to the lowest buffer filling [11]."

The actions are defined differently for each heuristic function. For MSS the actions are setting a new buffer size and the threshold values. For the

QoE-RAHAS on the other hand the actions correspond to choosing a new buffer size, the threshold value and the target buffer size [11].

Thus for the RL algorithm we need to decide the values for interval D , representing how often we change the parameter configuration and reward interval W , representing how many segments are taken into account when calculating the reward. Then the trade-off parameter ψ , lastly we need to set the parameters for the Q-learning algorithm. It is important to take notice that the W should be lower than D , since reward should point only to the current configuration [11].

So this approach uses reinforcement learning with Q-learning algorithm to enhance the performance of already existing heuristic functions MSS and QoE-RAHAS. The RL algorithms reward function is strongly tied to QoE, thus trying to maximise it.

3.2 Q-learning with factor selection

This method, by T.Wu and W.Van Leekwijck, also relies on the Q-learning algorithm but introduces some different ideas how to approach the problem. In this RL-HAS client the actions correspond to the client requesting any of the possible bit-rates in a certain state [12]. Then the reward function is defined as

$$r = -Af_i - Bq_i - Co_i \quad (3)$$

The f_i represents the freeze duration, and is defined through the length of the current playback buffer b_i and the time it takes to retrieve the next segment e_{i+1} [12].

$$f_i = \begin{cases} 0, & b_i - e_{i+1} \geq 0 \\ e_{i+1} - b_i, & b_i - e_{i+1} < 0 \end{cases} \quad (4)$$

The q_i is representing the quality of the stream requested. This is simply measured with the requested encoded bit-rate and the maximum available bit-rate. So if we request the highest possible bit-rate for a segment no penalty is given [12].

$$q_i = \frac{a_{max} - a_i}{a_{max}} \quad (5)$$

The o_i represents the oscillation in quality. If the current bit-rate is different from the previous one, there might be oscillation in the quality. If such change in quality appears, we look for similar quality change in

the past M segments before this one. If one is found and the change is in the opposite direction compared to our current change, an oscillation is detected and o_i is the average of these changes. If no such change is detected in the previous M segments or if it is in the same direction as the current one, o_i is 0. Lastly A, B and C are positive weights determining how each metric is valued when considering QoE. In the evaluation of this method values 100, 10 and 10 were used [12].

In this approach the state variables are selected through a forward selection method from potential variables. Each potential variable is added one at a time and then compared its effect on the performance of the algorithm. The significance of a potential variable is calculated as the average reward gained with the addition of a potential reward. The procedure is started by calculating which one of the potential variables are most significant and continued by adding the second most significant and so forth. Also if the null hypothesis test is rejected with p-value 0.05 we will not take into account the potential variable [12].

$$H_0 : Pr(\bar{R}_{S+v} > \bar{R}_S) \quad (6)$$

Where the \bar{R}_S is the average reward with variables in set S and \bar{R}_{S+v} with the set S and the potential variable v . In the experiment, in the paper where this approach was proposed, the potential variables were the current and the previous playback buffer, the current and the average bandwidth and the number of segments with the same trend. After the selection the set of the state variables were the current and the previous buffers and the average bandwidth [12].

In this approach Q-learning is also used but the reward function is a bit different although still tied to QoE metrics. The importance of the state variable selection is emphasised by using a factor selection process to choose only the significant features.

3.3 Tiyuntsong: A Self-Play RL Approach for HAS

The Tiyuntsong proposed by T.Huang, X.Yao, C.Wu, R-X. Zhang and L.Sun takes a different approach using RL and neural networks. The motivation behind this method was that "However, we observe that RL-based method often obtains high QoE score via some tricks due to the lack of guidance for optimisation in QoE [3]." Thus the Tiyuntsong utilises two agents which compete against each other, and the reward of each agent is either 0 or 1

for loss and win respectively. The method also uses a GAN based method to find out hidden features from past data [3].

Generative adversarial network (GAN) is an unsupervised machine learning method based on the assumption that two neural networks compete against each other. First networks acts as a generator and the other one as a discriminator. Simply put, the generator tries to produce samples as close to the real ones as possible. Then the discriminator tries to distinguish between those two. The GAN used in this approach, is inspired by the Least Squares GAN. LSGAN utilises the least square loss function for the discriminator [5]. The Tiyuntsong on the other hand takes advantage of the Leaky ReLU function [3].

The states are pushed into a neural network and the state variables are the throughput, download time and previous bit-rate and all of those are for the past k sequence. Also the remaining playback time, buffer length and a vector representing sizes of the next segments are state variables. Last state variable is a vector which contains the extra features generated by the Generative Adversarial Network (GAN) Enhancement Module. The action is defined as a vector representing the possible bit-rate of the next chunk. The output policy network is a probability distribution, as in the probability of selecting an action in a certain state [3].

The reward is represented as a win or loss and is determined by a rule. This rule can be a simple logistic rule or more complex neural network model generalised by AI can be used. In this case, a simple logistic method was used based on bit-rate, buffering freezes and the oscillation of the bit-rate. The following tables represent the rule

(a)

Rule	$b_0 > b_1$	$b_0 = b_1$	$b_0 < b_1$
$r_0 > r_1$	Table 1(b)	1	1
$r_0 = r_1$	0	Table 1(c)	1
$r_0 < r_1$	0	0	Table 1(b)

(b)

$\frac{r_0}{b_0+\epsilon} > \frac{r_1}{b_1+\epsilon}$	1
$\frac{r_0}{b_0+\epsilon} = \frac{r_1}{b_1+\epsilon}$	0
$\frac{r_0}{b_0+\epsilon} < \frac{r_1}{b_1+\epsilon}$	Table 1(c)

(c)

$s_0 > s_1$	1
$s_0 = s_1$	0/1
$s_0 < s_1$	0

Figure 2. The logistic rule table used by T.Huang et al

Then this rule is used to calculate the winning percentages for each

agent [3].

Often the decision of which bit-rate is requested relies on the current state but not on any of the previous states, this happens for example when we use only the Markov Decision Process model. In this method a GAN based approach is used to leverage unsupervised learning to automatically find the features from the past states. Those could be for example, maximum and minimum through puts in the past [3].

In conclusion this method uses RL based approach, but utilises neural networks and two agents competing for the reward. The reward is based on a rule which can be a logistic function or even a neural network model generated by AI. Furthermore, a GAN Enhancement Module is used to gather relevant features from the prior states.

3.4 Machine learning HAS using linear regression

In this approach a linear regression model is created and trained, and then used to classify the requested bandwidth. The linear regression model is trained on a separate MLHAS server and then communicated to an extended HAS client using the trained MLHAS. We are not going to examine the MLHAS system architecture in-depth. The basic structure is still presented [8].

Simplified explanation is that linear regression is used to model a relationship between some outcome and some input variables. The outcome can be some scalar value or some other label for the result. Then the model is trained with some data set and then used to predict future outcomes[6].

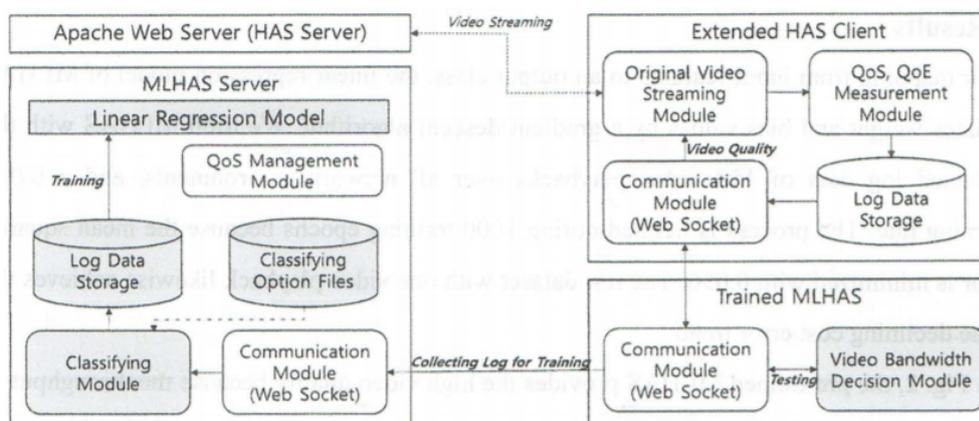


Figure 3. "MLHAS system architecture"[8]

The features used for the linear regression model are combinations of QoS, QoE and video content variables. The QoS features are the real

throughput, the average throughput of the last 3 segments, the delay times for the HTTP response and request, the network variation and the buffer length. Secondly the QoE features are the number of dropped frames, the re-buffering frequency and the duration which is caused when buffer length drops below 0.5 seconds. Lastly the video content features are the file size of a segment and the current video bandwidth. These 11 features are used to create and train the linear regression model. Thus we will get the weight and bias values for the linear regression model, mapping the HAS metrics and bit-rate to determine the bit-rate of the next segment [8].

This approach uses a linear regression model which considers 11 different features based on QoS, QoE and video content. Then the model is trained on a separate server, and that model is used to classify the next bit-rate request.

4 Comparison and discussion of different solutions

In this section we will compare the different machine learning approaches between themselves and consider their similarities, but also differences. We will also consider how the machine learning approaches compare to the conventional ones.

4.1 Differences and similarities

All the proposed solutions we dove into, took different approaches of the problem. Some of them were a bit closer to one another, like the two Q-learning approaches. The research has been evolving regarding the use of machine learning in HAS. The newer solutions and approaches are getting much more complex, which we can also see in this paper. If we compare the method using linear regression and Tiyuntsong, the latter is more complex using neural networks and enhancing the process using GAN.

Fair comparison of the methods presented in this paper is difficult, since all of them weigh the evaluation metrics a bit differently in their own experiments. Although all of them incorporate at least some form of QoE metrics. For example the linear regression MLHAS concentrates more on the performance on poor network conditions. The firstly introduced RL approach uses traditional heuristic functions together with the RL algo-

rithm, but on the other hand the second Q-learning method emphasises the importance of choosing the right features.

Also we have to keep in mind that these approaches are still simulations and the experiments are done in more or less controlled environments, and not implemented in larger scale in real-life situations. But still it seems, that using machine learning in HAS is beneficial and can improve the experience for the end-user.

4.2 Machine Learning approaches vs. traditional HAS

Even though all of the methods used different evaluation metrics the result compared to traditional HAS is clear, using machine learning improves the performance. For the first approach using traditional heuristics, for MSS the average reduction in buffer filling was 2.5% and for QoE-RAHAS it was 8.3% [11]. The second Q-learning method concluded in their experiment across 500 episodes that early on in the episodes the requested quality was very random but as the reinforcement learning algorithm got through more episodes the results were much better[12]. For Tiyuntsong T.Huang et al. stated that: "Compared to DynamicDash, Tiyuntsong improves the average bit-rate by 3.19%, decreases the average re-buffer time by 4.92%, and reduces the 95th percentile average bit-rate change by 16.47% respectively [3]." DynamicDash is an HAS algorithm not using machine learning [9]. Lastly the MLHAS using linear regression performs better especially in strong lossy networks yielding much smaller re-buffering frequency.

5 Conclusion

In conclusion, we provided a little background regarding HAS and then we presented four different approaches, how to incorporate machine learning together with HAS. We also presented a brief introduction to the machine learning method in-hand to provide some context for the proposed solution.

We tried to look all the methods from the perspective of QoE. Even though all of the methods used different approaches of machine learning, we can state that machine learning can definitely be used to enhance the HTTP adaptive video streaming process.

In general the machine learning methods have not been widely adopted

by big companies such as Netflix or YouTube at least in a larger scale. But surely these companies are interested in the capabilities of machine learning to improve the streaming process and to improve the end-user experience. These methods are especially beneficial in varying networks conditions, since there is a huge difference in watching a movie at home from a smart TV, compared to watching it in a train on your phone.

References

- [1] Saamer Akhshabi, Sethumadhavan Narayanaswamy, Ali C. Begen, and Constantine Dovrolis. An experimental evaluation of rate-adaptive video players over http. *Signal Processing: Image Communication*, 27(4):271 – 287, 2012. Modern Media Transport – Dynamic Adaptive Streaming over HTTP (DASH).
- [2] M. Claeys, S. Latre, J. Famaey, and F. De Turck. Design and evaluation of a self-learning http adaptive video streaming client. *IEEE Communications Letters*, 18(4):716–719, April 2014.
- [3] Tianchi Huang, Xin Yao, Chenglei Wu, Rui-Xiao Zhang, and Lifeng Sun. Tiyuntsong: A self-play reinforcement learning approach for abr video streaming. *arXiv preprint arXiv:1811.06166*, 2018.
- [4] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. Neural adaptive video streaming with pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 197–210. ACM, 2017.
- [5] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, and Zhen Wang. Multi-class generative adversarial networks with the L2 loss function. *CoRR*, abs/1611.04076, 2016.
- [6] D.C. Montgomery, E.A. Peck, and G.G. Vining. *Introduction to Linear Regression Analysis*. Wiley Series in Probability and Statistics. Wiley, 2015.
- [7] Stefano Petrangeli, Jeroen Famaey, Maxim Claeys, and Filip De Turck. A qoe-driven rate adaptation heuristic for enhanced adaptive video streaming. *Ghent University-iMinds, Department of Information Technology, Tech. Rep*, pages 2332–7731, 2014.
- [8] Chaemin Seong, Seongjun Hong, and Kyungshik Lim. A machine learning-based adaptive video streaming algorithm in dynamic network environments. *International Information Institute (Tokyo).Information*, 20(9):6369–6376, 09 2017. Copyright - Copyright International Information Institute Sep 2017; Last updated - 2018-04-02.
- [9] Kevin Spiteri, Ramesh Sitaraman, and Daniel Sparacio. From theory to practice: Improving bitrate adaptation in the dash reference player. In *Proceedings of the 9th ACM Multimedia Systems Conference, MMSys '18*, pages 123–137, New York, NY, USA, 2018. ACM.
- [10] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 1998.
- [11] J. van der Hooft, S. Petrangeli, M. Claeys, J. Famaey, and F. De Turck. A learning-based algorithm for improved bandwidth-awareness of adaptive streaming clients. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 131–138, May 2015.
- [12] Tingyao Wu and Werner Van Leekwijck. Factor selection for reinforcement learning in http adaptive streaming. In Cathal Gurrin, Frank Hopfgartner, Wolfgang Hurst, Håvard Johansen, Hyowon Lee, and Noel O'Connor, editors, *MultiMedia Modeling*, pages 553–567, Cham, 2014. Springer International Publishing.

Function composition in serverless computing

Miika Kanerva

miika.kanerva@aalto.fi

Tutor: Mario Di Francesco

Abstract

Modern cloud services provide the Function-as-a-Service (FaaS) model, which enables users to create stateless, function-based applications in the cloud. The service provider manages the function orchestration and the allocated resources. This leaves the users with less operational concerns and allows them to focus on the business logic of their applications. However, composing multiple functions to create complex workflows is still in its early stages. Existing function orchestration systems, such as Amazon Step Functions, offer a way to form workflows including cloud functions and other cloud resources. This seminar paper studies the benefits and drawbacks of the FaaS architecture, function composition in the context of the cloud computing and experimenting with the Amazon Step Functions service. The aim of the experimentation is to show an example of a realistic workflow that is possible to achieve with that service.

KEYWORDS: *Cloud computing, Serverless, Function composition, FaaS, AWS Step Functions*

1 Introduction

Cloud computing has become increasingly relevant in the IT industry through the evolution of containers, virtualization and event-driven workflows. With fine-grained services, application components hosted in the cloud can be deployed and scaled independently. Thanks to these developments, the concept of *serverless computing* is possible today. This term refers to a cloud computing paradigm where building and running cloud applications does not require server management from the cloud user [1]. Instead, most of the operational concerns are managed by the service provider [2]. Some notable examples of serverless platforms are AWS Lambda¹, Google Cloud Functions² and Apache Openwhisk³.

Most of serverless platforms provide the Function-as-a-Service (FaaS) model, or the Backend-as-a-Service (BaaS) model, or both [1]. In FaaS, the service provider manages the resources, lifecycle and event-driven execution of the functions provided by the user [3]. An example of FaaS platform is AWS Lambda. On the other hand, BaaS is a form of serverless computing where the service provides an API that auto-scales and operates transparently [1]. BaaS can be utilized, for instance, as a mobile application backend, as with Google Firebase⁴. Furthermore, Google Cloud Functions can be incorporated to work together with the Firebase backend service.

Although the serverless computing is still a relatively new concept, several studies have been conducted on the topic [2][3][4]. However, few studies exist on the cloud function composition. In the context of serverless computing, function composition refers to the coordination mechanisms between functions [5]. One of the existing studies is from Baldini et al.~[6], which introduces the current state of the function composition in serverless computing and presents the serverless trilemma. The trilemma consist of three competing constraints in modern cloud function composition. These constraints will be further introduced in Section 4.

The objective of this paper is twofold. First, identifying the main characteristics of the Function-as-a-Service architecture. In addition to the main characteristics, the benefits and current challenges and problems of the architecture are introduced. Second, examination of how multiple

¹AWS Lambda: <https://aws.amazon.com/lambda/>

²Google Cloud Functions: <https://cloud.google.com/functions/>

³Openwhisk: <https://openwhisk.apache.org/>

⁴Google Firebase: <https://firebase.google.com/>

functions can be composed to create complex workflows. The service used for experimentation of the function composition in the cloud is AWS Step Functions.

2 Background

Before proceeding to examine the FaaS and the function composition in-depth, it is necessary to introduce two important concepts. First, the underlying architecture of the serverless platforms will be introduced. Second, a brief introduction to container orchestration will be given.

2.1 Serverless Architecture

Van Eyk et al. [3] define serverless computing as a form of cloud computing that allows users to run stateless, event-driven and function-based applications without having to address the operational logic. Contrary to the name *serverless*, servers are still required to run a serverless platform [1]. The main point here is that these platforms are operated by a third-party service provider, for example, Amazon.

Figure 1 illustrates the main components of a serverless architecture. Events are sent in by event sources, such as a user clicking a button or a file being added to a cloud storage. After the event is received, it is added to an event queue. The events are passed down from such a queue to the dispatcher, which determines which function(s) to trigger and tries to find existing instances of the functions [2]. If no existing instances are found, a new instance of a function is created. The event is finally sent to the target function. The function is stopped after the function has sent back a response and execution logs are gathered [2].

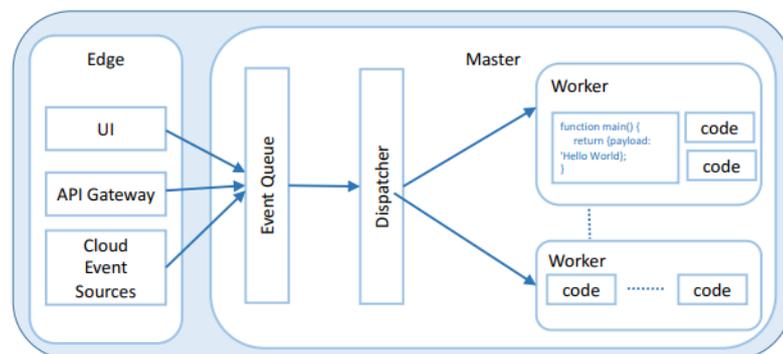


Figure 1. Serverless Architecture [2]

The serverless architecture offers benefits to cloud users by making the

cloud easier to program as they can easily write cloud-based components. Moreover, it is also beneficial for the service provider [7], as the stateless nature and short lifetime of serverless tasks increase resource utilization. Furthermore, the control over the instances allow the providers to utilize the less popular instances that are not used by the users that manage their own servers.

2.2 Container Orchestration

Containers are a virtualization technology for lightweight system-level process virtualization [8]. Virtual containers can encompass an entire application and its dependencies, and can run on every Linux distribution. Container orchestration refers to the management of these virtual containers, which includes the deployment, provisioning, scaling and resource allocation.

Systems such as Kubernetes⁵ and Docker Swarm⁶ offer the user the maximum control over the orchestration of their containers [1]. Consequently, as the amount of control grows, so does the amount of developer responsibility. However, in the context of serverless computing the customer does not have the control over their containers. Instead, the responsibility of the container orchestration belongs to the service provider.

3 Function-as-a-Service

While serverless computing is a broad concept, Function-as-a-Service (FaaS) is one of the methods for implementing a serverless architecture. FaaS enables running stateless, event-triggered functions that are hosted and managed by a third-party service provider [9]. In essence, a FaaS platform replaces the server in a traditional monolithic software architecture. Furthermore, a cloud function can be thought as a microservice with a single endpoint and responsibility [10]. This section will introduce the general characteristics of FaaS, its benefits and the current problems and challenges of the FaaS architecture.

⁵Kubernetes: <https://github.com/kubernetes/kubernetes>

⁶Docker Swarm: <https://docs.docker.com/engine/swarm/>

3.1 Characteristics

One of the defining characteristics of serverless platforms, which makes them attractive, is how the service provider charges the users for the function usage. The functions are charged only by the used resources and execution time [2]. This indicates that the customer is not charged for the time their functions are idle.

The second characteristic of FaaS architecture is that the functions are stateless. Though the functions have access to a local state of the instance they are being hosted in, there is no guarantee that the state will persist across multiple invocations [9]. Therefore, a persistent state of a function should be stored in an external storage (e.g., Amazon S3⁷) outside of the function instance.

The deployment of cloud functions is different from other cloud resources, such as virtual machines. Most of serverless platforms only require the user to upload the code for the function [9]. After that, the code is compiled and built into an artifact, such as a binary or a container image [1]. The deployment pipeline of a function is illustrated below in Figure 2.

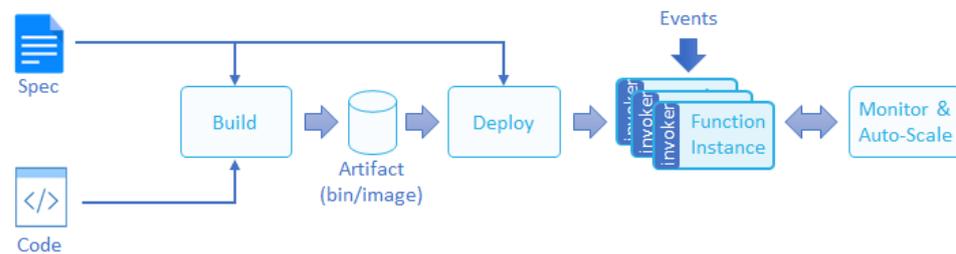


Figure 2. Function deployment pipeline [1]

3.2 Benefits

From the user's perspective, the first and most notable benefit of FaaS architecture is the increased programming productivity [7]. While platforms such as Kubernetes offer more control over the application infrastructure, users have to spend considerable amount of time managing and configuring their application instances. On the contrary, the serverless architecture abstracts the underlying cloud infrastructure from the users, which leaves the users with more time to spend on the business logic of their application [2].

Cloud functions run on-demand and are billed based on the execution

⁷Amazon S3: <https://aws.amazon.com/s3/>

time [11]. Functions are not billed when idle, which makes FaaS optimal for small tasks. From the perspective of the service provider, this also enables a better way to optimize resource consumption in the cloud.

Whereas traditional server-side applications could be tied to a specific framework, FaaS enables the functions to be written in multiple programming languages. In case of multiple functions, each of them can be written in a different programming language.

3.3 Challenges and problems

There exists a possibility that no instances of a user's function are up and running when an event triggers a function. As a result, the FaaS platform has to create and run a new instance for the function to run inside. This is often referred as cold start latency, which can be tens of seconds [7]. This negatively affects the predictability of the function performance.

In order to manage a state between functions, an intermediary storage service, such as Amazon S3, has to be utilized [4]. The problem of the intermediary storage services is that they might add hundreds of seconds of latency when accessed, and incur high access costs [7].

The third challenge of modern FaaS platforms is that there is no standardization between the platforms [1]. As a consequence, portability of an application from one FaaS platform to another might not be a simple task, which might result in vendor lock-in [7].

4 Function composition

With the FaaS architecture, users can create stateless functions that are billed only when executed. However, tools are required to compose and coordinate the flow between multiple functions to create complex workflows with the serverless architecture. This section will examine the current services and their constraints when it comes to function composition. A short introduction to AWS Step Functions and also experimentation with the service will be included.

4.1 The Serverless Trilemma

The study of Baldini et al. [6] identifies three competing constraints in cloud function composition, which together form the so-called *serverless trilemma*. The first is the double billing constraint, which is related to

the billing of the cloud functions. Function orchestration solutions that implement the scheduling of function sequences with a separate function violate this constraint. A problem arises, because the scheduler function will be running until the whole sequence has been executed. This will result in double charges, because the user will have to pay for the execution time of the scheduler and the components of the sequence. The problem is illustrated below in Figure 3.

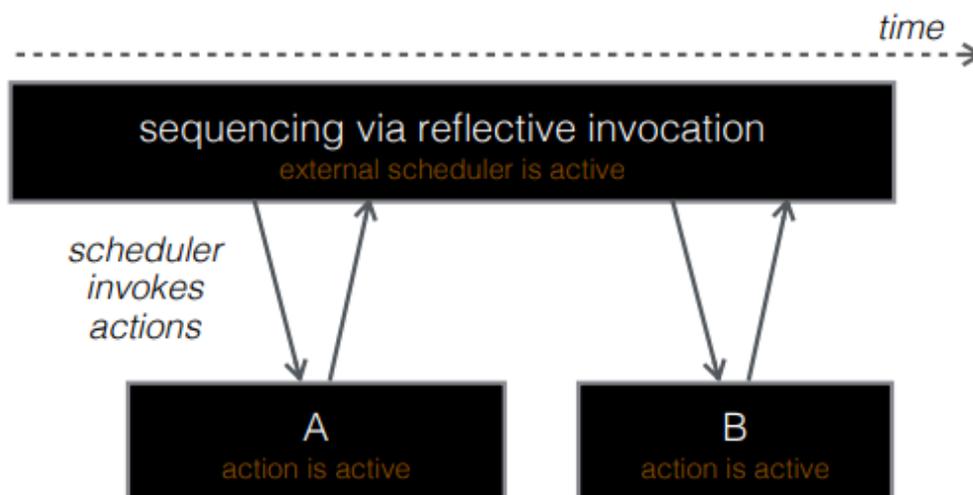


Figure 3. The double billing problem: the user pays for the execution time of both the scheduler and the constituents of the sequence [6]

The second constraint states that the functions should be treated as black boxes. This constraint is broken by the function schedulers that combine the source code of the functions in a sequence. The problem with this type of function schedulers is that the source code of all of the functions is required to be available and to be written with the same language. The idea of the constraint is to avoid these requirements.

Lastly, the third constraint states that the function composition should obey a substitution principle with respect to synchronous invocation. In essence, this constraint states that a composition of functions should also be treated as a function. For example, this could mean replacing a part of a large composition with another function composition. External schedulers, such as AWS Step Functions (ASF), break this constraint because they are not part of the system itself [5].

A system that does not break any of the three constraints is referred as *ST-safe* [6]. López et al. [5] compared existing FaaS orchestration systems, AWS Step Functions (ASF), IBM Composer and Azure Durable Functions. Out of these three, only Azure Durable Functions were *ST-*

safe. Nevertheless, as ASF will be used for experimentation in this paper, as it is the most mature of the existing function orchestrators.

4.2 AWS Step Functions

AWS Step Functions⁸ (ASF) is a service provided by Amazon, that lets users coordinate multiple AWS services into serverless workflows. Step functions are defined as state machines that are described declaratively using the Amazon States Language (ASL), which is a JSON-based language [12]. This subsection will give an introduction to ASL and also introduces some experiments with the ASF.

Concepts

In Amazon States Language, a *state* is a top-level object in the ASL, which describes one in a composition. State description is defined below (see Listing 1). The **Type**-field is a required field and it defines the type of the state. The possible types of a state are Task, Choice, Wait, Succeed, Fail and Parallel.

```
"HelloWorld": {
  "Type": "Task",
  "Resource": "arn:aws:lambda:us-east-1:123456789012:function:HelloWorld",
  "Next": "NextState",
  "Comment": "Executes the HelloWorld Lambda function"
}
```

Listing 1. Example State [12]

The transitions between states link them together and are defined in the **Next**-field, which is defined with the name of the next state. The data that is passed between states is required to be JSON. A user can provide an initial JSON as input, which is passed to the start state. If a state utilizes a resource, for example a Lambda function (see **Resource**-field in Listing 1), the output of a state will be the value returned by the Lambda function.

Experimentation

To create a proof of concept about function composition in AWS Step Functions, some experimentation had to be conducted. The point of this experiment was not to create anything too complex, but instead display the capabilities of ASF. The state machine made for this experiment is named *CreateJPGThumbnails*, which generates three thumbnails of a jpg-image.

⁸AWS Step Functions: <https://aws.amazon.com/step-functions/>

The state machine is invoked when a file is uploaded to a specified S3 bucket, which will be referred as *s3-bucket*. The specification was done through a AWS CloudWatch⁹ rule, which was defined to track the "PutObject" operations of the *s3-bucket* and target the *CreateJPGThumbnails* state machine. In short, when a file is uploaded to *s3-bucket*, the CloudWatch rule would start a new execution of the state machine.

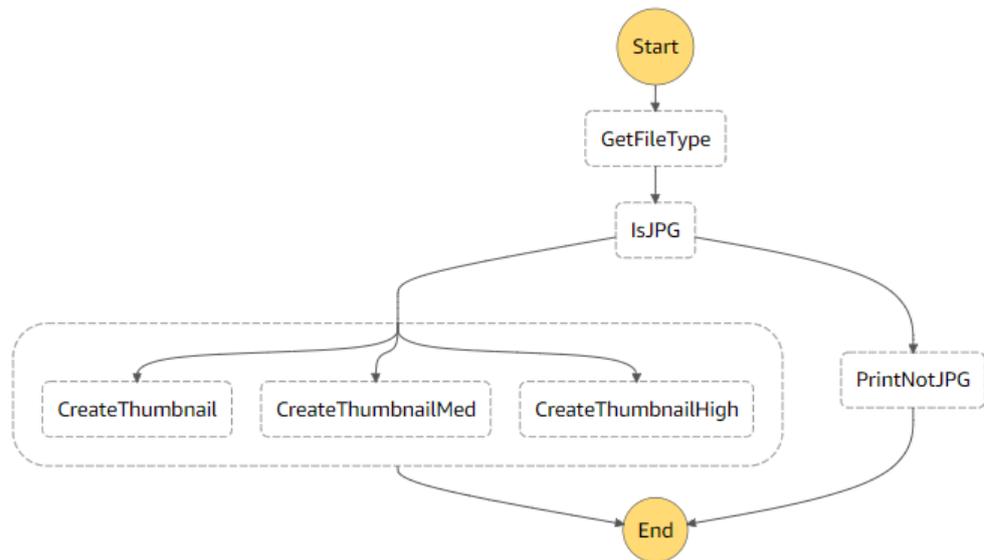


Figure 4. *CreateJPGThumbnails* state machine

The visual representation of the state machine is displayed above in Figure 4. The definition of the state machine is included in the appendix 1. The state machine is executed when a file is uploaded to S3 and gets the information about the "PutObject" operation as initial input. This input is passed down to the state *GetFileType*, which is a Task state that uses a Lambda function named *GetS3ResourcesAndFileType*. The Lambda function (appendix 2) only fetches the relevant data from the initial input and returns that. The output of the *GetFileType* (see Listing 2) is then passed to the next state *IsJPG* as input.

```
{
  "bucket": "arn:aws:s3:::bucket-name",
  "fileKey": "some-image.jpg",
  "fileType": "jpg"
}
```

Listing 2. *GetFileType* example output

The *IsJPG* state is a Choice state, which determines how the execution

⁹Amazon CloudWatch: <https://aws.amazon.com/cloudwatch/>

will continue based on two *Choices*. The first choice checks if a variable "\$.fileType" equals to the string "jpg", and if does the next state in the execution would be *CreateDifferentSizes*. The \$-sign in the variable value refers to the way of accessing nodes of the input in ASL, where just "\$" would return give the whole input and "\$.fileType" just the value of the *fileType* node [12]. On the contrary, the second choice of the *IsJPG* checks whether the file type does **not** match to "jpg", and sends the execution next to the state *PrintNotJPG* if the condition applies.

Depending on the *fileType*, the execution should now enter either *CreateDifferentSizes* or *PrintNotJPG*. Both of these states are last in the execution, since the "End" property is set to true. The state *PrintNotJPG* is a simple Pass state, which outputs a string stating "The file is not a JPG-file, ending.". On the other hand, *CreateDifferentSizes* (dotted outline around the three functions in 4) is a state with type "Parallel". The parallel state consists of three branches, which each consist of one Task state. Since the choice state did not change the input, the same input is passed down to *CreateDifferentSizes* and each of its branches. Each task state uses a Lambda function as a resource, whose source code is found in the appendix 3. The functions are executed in parallel, and each one of them downloads the original image to a temporary storage, resizes the image to a specified size and uploads the new image to another bucket in S3. Finally, each function returns the width, height and name of the new image, which is aggregated to one output when each branch has finished computation (see Listing 3).

```
[
  {
    "width": 200,
    "height": 133,
    "dstKey": "med-resized-image.jpg"
  },
  {
    "width": 100,
    "height": 66,
    "dstKey": "thumbnail-resized-image.jpg"
  },
  {
    "width": 400,
    "height": 266,
    "dstKey": "high-resized-image.jpg"
  }
]
```

Listing 3. Example aggregated output

5 Conclusions

The first objective of this paper was to identify the main characteristics, benefits and challenges of FaaS model. This paper defined three characteristics of the FaaS model. First, the unique billing model where the user pays only for the execution time of the functions. Second, the statelessness of the functions, which indicated that a state is not guaranteed to persist across multiple invocations. Third, the way cloud functions are deployed.

In addition to the main characteristics, some of the benefits of the architecture were introduced. Increased productivity, little operational concern and the way FaaS does not tie applications any frameworks make it attractive to developers. However, the architecture has its challenges. The lack of standardization enforces vendor lock-in and the statelessness of the cloud functions makes the architecture not suitable for all applications. Nevertheless, FaaS is suitable for multiple applications and can be incorporated into other architectures as fine-grained services.

The second objective was to experiment function composition to create complex workflows in the cloud. To understand the limitations of modern function composition services, the *serverless trilemma* was introduced. The experimentation with AWS Step Functions demonstrated that the service can be used to compose both simple and complex workflows. The service supports both sequential and parallel execution of tasks, which enables more options when designing serverless workflows. Though ASF does not comply all of the constraints of the *serverless trilemma*, it can still be utilized to compose existing cloud functions. A suggestion to improve the service would be the possibility of including an existing state machine inside another state machine. This improvement would also make the service *ST-safe*.

References

- [1] S. Allen, C. Aniszczyk, C. Arimura, B. Browning, L. Calcote, A. Chaudhry, D. Davis, L. Fourie, A. Gulli, Y. Haviv, D. Krook, O. Nissan-Messing, C. Munns, K. Owens, M. Peek, and C. Zhang, "CNCF Serverless Whitepaper v1.0,"

- GitHub*, September 2018. URL: <https://github.com/cncf/wg-serverless/tree/master/whitepapers/serverless-overview>.
- [2] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, N. Mitchell, V. Muthusamy, R. Rabbah, A. Slominski, *et al.*, “Serverless computing: Current trends and open problems,” in *Research Advances in Cloud Computing*, pp. 1–20, Springer, 2017.
 - [3] E. Van Eyk, L. Toader, S. Talluri, L. Versluis, A. Uță, and A. Iosup, “Serverless is more: From paas to present cloud computing,” *IEEE Internet Computing*, vol. 22, no. 5, pp. 8–17, 2018.
 - [4] J. M. Hellerstein, J. M. Faleiro, J. E. Gonzalez, J. Schleier-Smith, V. Sreekanti, A. Tumanov, and C. Wu, “Serverless computing: One step forward, two steps back,” *CoRR*, vol. abs/1812.03651, 2018.
 - [5] P. G. López, M. Sánchez-Artigas, G. París, D. B. Pons, Á. R. Ollobarren, and D. A. Pinto, “Comparison of faas orchestration systems,” in *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pp. 148–153, IEEE, 2018.
 - [6] I. Baldini, P. Cheng, S. J. Fink, N. Mitchell, V. Muthusamy, R. Rabbah, P. Suter, and O. Tardieu, “The serverless trilemma: function composition for serverless computing,” in *Proceedings of the 2017 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, pp. 89–103, ACM, 2017.
 - [7] E. Jonas, J. Schleier-Smith, V. Sreekanti, C.-C. Tsai, A. Khandelwal, Q. Pu, V. Shankar, J. Menezes Carreira, K. Krauth, N. Yadwadkar, J. Gonzalez, R. A. Popa, I. Stoica, and D. A. Patterson, “Cloud programming simplified: A berkeley view on serverless computing,” Tech. Rep. UCB/EECS-2019-3, EECS Department, University of California, Berkeley, Feb 2019.
 - [8] A. Tosatto, P. Ruiu, and A. Attanasio, “Container-based orchestration in cloud: state of the art and challenges,” in *2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems*, pp. 70–75, IEEE, 2015.
 - [9] M. Roberts, “Serverless Architectures,” *MartinFowler.com*, May 2018. Available at: <https://martinfowler.com/articles/serverless.html>.
 - [10] E. Van Eyk, A. Iosup, C. L. Abad, J. Grohmann, and S. Eismann, “A spec rg cloud group’s vision on the performance challenges of faas cloud architectures,” in *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering*, pp. 21–24, ACM, 2018.
 - [11] L. M. Vaquero, F. Cuadrado, Y. Elkhatib, J. Bernal-Bernabe, S. N. Srirama, and M. F. Zhani, “Research challenges in nextgen service orchestration,” *Future Generation Computer Systems*, vol. 90, pp. 20–38, 2019.
 - [12] Amazon, “Amazon states language.” URL: <https://states-language.net/spec.html>.
 - [13] Amazon, “Serverless image resizing.” URL: <https://github.com/amazon-archives/serverless-image-resizing>.

1 Appendix A: Step Functions state machine definition

```
{
  "Comment": "A State machine that is invoked when a file is uploaded to S3
    bucket, if the file is a JPG file, three different size images are
    created in another bucket",
  "StartAt": "GetFileType",
  "States": {
    "GetFileType": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:region:id:function:GetS3ResourcesAndFileType",
      "Next": "IsJPG"
    },
    "IsJPG": {
      "Type": "Choice",
      "Choices": [
        {
          "Variable": "\$.fileType",
          "StringEquals": "jpg",
          "Next": "CreateDifferentSizes"
        },
        {
          "Not": {
            "Variable": "\$.fileType",
            "StringEquals": "jpg"
          },
          "Next": "PrintNotJPG"
        }
      ]
    },
    "CreateDifferentSizes": {
      "Type": "Parallel",
      "Branches": [
        {
          "StartAt": "CreateThumbnail",
          "States": {
            "CreateThumbnail": {
              "Type": "Task",
              "Resource": "arn:aws:lambda:region:id:function:CreateThumbnail",
              "End": true
            }
          }
        }
      ]
    },
    {
      "StartAt": "CreateThumbnailMed",
      "States": {
```

```

    "CreateThumbnailMed": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:region:id:function:CreateMedRes",
      "End": true
    }
  },
  {
    "StartAt": "CreateThumbnailHigh",
    "States": {
      "CreateThumbnailHigh": {
        "Type": "Task",
        "Resource": "arn:aws:lambda:region:id:function:CreateHighRes",
        "End": true
      }
    }
  }
],
"End": true
},
"PrintNotJPG": {
  "Type": "Pass",
  "Result": "The file is not a JPG-file, ending.",
  "End": true
}
}
}

```

2 Appendix: GetFileType function code

```

exports.handler = (event, context, callback) => {
  const resources = event['detail']['resources']
  const bucket = resources[1].ARN
  const fileKey = resources[0].ARN.split("/")[1]
  const fileType = resources[0].ARN.split(".")[1]

  const output = {bucket, fileKey, fileType}
  callback(null, output);
};

```

3 Appendix: CreateThumbnail function code

The code for the image resizing utilized the base code from Amazon Archives [13], which was modified to be suitable for the experiment. Each of the

three functions *CreateThumbnail*, *CreateMedRes* and *CreateHighRes* (see Figure 4) used the same function, but the values of max width and height were 100, 200 and 300 respectively.

```
var async = require('async');
var AWS = require('aws-sdk');
var gm = require('gm')
    .subClass({ imageMagick: true }); // Enable ImageMagick integration.
var util = require('util');

var MAX_WIDTH = 100;
var MAX_HEIGHT = 100;

var s3 = new AWS.S3();

exports.handler = function(event, context, callback) {
    var srcBucket = event.bucket.split(":::")[1]
    var srcKey = event.fileKey;
    if(srcKey.includes("resized-high")) {
        callback("File already resized");
        return;
    }
    var dstBucket = srcBucket + "-storage";
    var dstKey = "med-resized-" + srcKey;
    var width;
    var height;

    // Sanity check: validate that source and destination are different
    buckets.
    if (srcBucket == dstBucket) {
        callback("Source and destination buckets are the same.");
        return;
    }

    // Download the image from S3, transform, and upload to a different S3
    bucket.
    async.waterfall([
        function download(next) {
            // Download the image from S3 into a buffer.
            s3.getObject({
                Bucket: srcBucket,
                Key: srcKey
            },
            next);
        },
        function transform(response, next) {
            gm(response.Body).size(function(err, size) {
```

```

        // Infer the scaling factor to avoid stretching the image
        unnaturally.
        var scalingFactor = Math.min(
            MAX_WIDTH / size.width,
            MAX_HEIGHT / size.height
        );
        width = scalingFactor * size.width;
        height = scalingFactor * size.height;

        // Transform the image buffer in memory.
        this.resize(width, height)
            .toBuffer(imageType, function(err, buffer) {
                if (err) {
                    next(err);
                } else {
                    next(null, response.ContentType, buffer);
                }
            });
    });
},
function upload(contentType, data, next) {
    // Stream the transformed image to a different S3 bucket.
    s3.putObject({
        Bucket: dstBucket,
        Key: dstKey,
        Body: data,
        ContentType: contentType
    },
    next);
}
], function (err) {
    if (err) {
        console.error(
            'Unable to resize ' + srcBucket + '/' + srcKey +
            ' and upload to ' + dstBucket + '/' + dstKey +
            ' due to an error: ' + err
        );
    } else {
        console.log(
            'Successfully resized ' + srcBucket + '/' + srcKey +
            ' and uploaded to ' + dstBucket + '/' + dstKey
        );
    }

    callback(null, {width, height, dstKey});
}
);

```


A Survey on Intrusion Detection Datasets

Auli Mustonen

auli.mustonen@aalto.fi

Tutor: Buse Gul Atli

Abstract

The protection against network attacks is of vital importance on the Internet. These attacks can be detected by Intrusion Detection Systems (IDSs), which observe the behaviour of the network traffic. Protocols carrying the traffic are evolving continuously. Therefore, their presence in the IDS datasets used by machine learning-based IDS need to be revised periodically. In this survey, we examine the recently published IDS datasets and make an evaluation on properties they include. Especially, the focus is to evaluate, if the network protocols and attacks in the dataset relate the current status of network traffic. The main focus is in literature research, but we utilize current data mining and packet analyzer tools to study the contents of these datasets in practice. Through a detailed evaluation of each dataset, we deduce the validity of each dataset for current intrusion detection systems, and finally, reach conclusions and future trends.

KEYWORDS: *Intrusion Detection, Machine Learning, IDS dataset*

1 Introduction

Without high-quality intrusion detection and management systems, it is not possible to provide reliable Internet services. The detection system may reside in an individual computer as a Host-based IDS (HIDS), which tracks system configurations and applications for abnormal behaviour. Instead of a host, the Network-based IDS (NIDS) is installed in network elements [12]. NIDS monitors network traffic trying to detect abnormal activities. One of the main challenges in doing so, is the availability of up-to-date, inclusive and efficient IDS datasets. The older datasets may fail to distinguish intrusions from a diversified, benign traffic. The traffic information may be anonymized, which does not correspond the current behaviour of the traffic. Older datasets may also lack sufficient number of different attacks and properties such as metadata and the feature set. There are many important approaches when constructing proper IDS datasets. The dataset must contain traffic samples of the attacks to be detected. The selection of sample traffic should be up to date. The IDS will fail to detect the novel attacks, if they do not exist in the dataset.

Our survey focuses on datasets used by network-based IDSs. First, our contribution in this report is to evaluate the most up-to-date IDS datasets based on the most recent literature reports. Second, we study the download files of the datasets using the newest data mining and packet analyzer tools to complement the evaluation by practical aspects. The structure of this survey is as follows. The next chapter covers some earlier researches concerning IDS datasets. Chapter 3 includes detailed evaluations of the recently published dataset. Chapter 4 sums up the results of dataset evaluations. Finally, the future trends and conclusions are presented in Chapter 5.

2 Related Work

The IDS datasets have been studied intensively in process of time, because the quality of the dataset is crucial for the efficiency of the IDS system. The studies concentrate on the up-to-dateness and volume of traffic captures. The properties of IDS datasets vary from a small special purpose dataset to huge multi-protocol datasets with extensive attack profiles. Hindy et al. have studied an extensive set of current IDS datasets [7]. Their taxonomy includes the evaluation of the capabilities and assets

of the datasets. In addition, the taxonomy consists of different traffic protocols in the dataset and the evaluation of the validity of those protocols. Frequently, the traffic data in previous datasets contains such protocols which are not used any more. On the other hand, Bhuyan et al. criticize datasets which have been constructed by creating the traffic by tools and infiltrating the attacks in it [13]. We will explore new aspects, which concern the benefits of artificially generated traffic. Karatas et al. have examined the KDDCUP99, NSL-KDD, CICIDS2017 and CSE-CIC-IDS2018 datasets, but they are missing datasets for mobile data. Our study extends this dataset assembly by evaluating mobile and WiFi datasets as well. The evaluation in [6] mentions the count of features in the datasets, but does not evaluate them in detail. In that respect, our study includes a more detailed evaluation of traffic features.

3 Dataset Evaluation

3.1 Basis of the evaluation

This section examines the most cited IDS datasets, which are presented in Table 1 and Table 2. The traffic data in these datasets has been collected from public internet and special networks such as WiFi, LAN and mobile systems. The proportion of mobile network traffic has increased dramatically in recent years. Therefore, the inclusion of the mobile dataset in this study is essential.

The evaluation of the datasets is based on the following criteria:

- the dataset features should have varying values
- diversity of attacks
- relevancy of the protocols
- sufficiency of the traffic volume
- quality of traffic labelling

3.2 CICIDS2017

The CICIDS2017 dataset has been constructed by Canadian Institute for Cybersecurity. The downloadable files include Comma-Separated Values (CSV) files and Packet Capture (PCAP) files for five days with varying set of attacks. The CSV files have been constructed by means of CICFlowMeter tool, which extracted over 80 features such as time stamp, source and

Table 1. Intrusion detection datasets.

Dataset	Year	ML algorithms used for ML-based IDSs	Attacks	Reference
KDDCUP99	2008	Bayesian Clustering, Tree Classifier	DoS, U2R, R2L, Prob	[5]
KDD'99	2018	ANN, FLN, PSO, PSO-FLN, ELM	DoS, U2R, R2L, Prob	[9]
DEFCON	2012	k-means, KMC+NBC	Zero-day, SSH Brute Force, DoS, DDoS	[3]
ISCX2012	2013	K-Means clustering, KMC+NBC	DoS, u2r, r2l, probe	[22]
ISCX2015	2016	PCA	r2l, l2l	[20]
AWID	2016	J48, Random Forest, OneR	Total Frames Amok, Deauthentication, Authentication Req, Beacon, Probe Resp, ARP, Fragmentation, Evil Twin, Caffè Latte	[4]
CICACGM	2017	Random Forest, K-Nearest Neighbor, Decision Tree, Random Tree, Regression	fast-flux, botnet, others	[1]

Where:

ANN = Artificial Neural Network

DoS = Denial of Service[9]

ELM = Extreme Learning Machine[9]

FLN = Fast Learning Network

KMC = K-Means Clustering

NBC = Naive Bayes Classifier

PCA = Principal Component Analysis

Prob = Probing Attack

PSO = Particle Swarm Optimization

R2L = Remote to Local Attack [19]

U2R = User to Root Attack [5], [11]

DDoS = Distributed Denial of Service [3]

Table 2. Intrusion detection datasets.

Dataset	Year	ML algorithms used for ML-based IDSs	Attacks	Reference
NGIDS-DS	2017	Fuzzy inference	Exploits, DoS, Worms, Generic, Reconnaissance, Shellcode, Backdoors	[21]
CICAndMal2017	2018	Random Forest, K-Nearest Neighbor, Decision Tree	Adware(10 malware families), Ransomware (10 malware families), Scareware(11 malware families), SMS Malware(11 malware families)	[2]
CSE-CIC-IDS2018	2018	RNN	Bruteforce, DoS, Infiltration, Botnet, DDoS, Heartbleed, Web attacks	[6], [14]
CICIDS2017	2018	KNN, RF, ID3, Adaboost, MLP, Native-Bayes, QDA	DoS, DDoS, Brute Force, XSS, SQL Injection, Infiltration, Port scan, Heartbleed and Botnet.	[10]
NSL-KDD	2018	Decision Tree, Fuzzy Rule, Random Forest, Gradient BoostedTree Learner	DoS, U2R, R2L, Probing Attack	[12]

Where:

DDoS = Distributed Denial of Service [6], [3]

DoS = Denial of Service [6]

ID3 = Iterative Dichotomiser 3 [10]

KNN = K-Nearest Neighbors [10]

MLP = Multilayer Perceptron [10]

QDA = Quadratic Discriminant Analysis [10]

R2L = Remote to Local Attack [19]

RF = Random Forest [10]

RNN = Recursive Neural Network [14]

U2R = User to Root Attack [5], [11]

destination IPs, source and destination ports [10], [17], [16]. The traffic volume and its coverage provide a valid base for the contemporary IDSs.

The feature values should contain sufficient information to be useful in machine learning. The contents of the feature files can be studied by means of tools such as Weka, which provides min, max, mean and standard deviation values for each feature. By means of the Weka tool, we could verify that the values of separate features are clearly divergent, because they have a clearly non-zero standard deviation. In addition, the distribution of feature values can be affected by the malicious traffic. Figure 1 verifies that large number of DDoS attack packets considerably change the distribution of average packet size values.

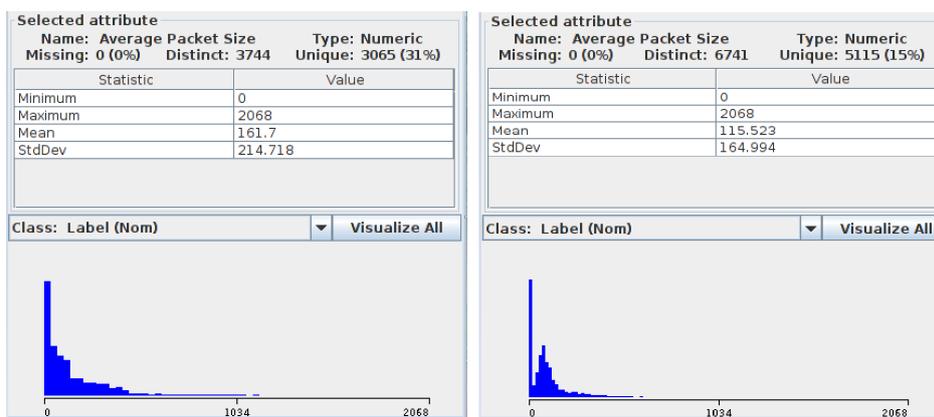


Figure 1. Packet size distribution of benign and DDoS infected traffic on the right.

The different attack sets reflect existing traffic profiles in the network. The attack packet flow labels include exact names of the attack types [17]. All the contemporary network protocols such as https and ssh have been applied in the dataset traffic [17]. The protocol distribution in a traffic capture can be examined by Wireshark tool.

We have examined the CICIDS2017 dataset according to the criterions listed in 3.1. First, the feature values vary sufficiently. Second, a diversity of contemporary attacks has been applied in constructing different attack profiles. Third, the protocol sets have been constructed to present the current status of traffic in Internet. Fourth, the volume of the captured traffic responds the traffic of a whole work week. Fifth, the labelling of the packages is very accurate. These qualities support the selection of proper dataset for different machine learning needs.

3.3 CICAndMal2017

The CICAndMal2017 Android malware dataset has been constructed using Android smart phones to provide a real operating environment [2]. The dataset is labelled and it covers 42 different malware families grouped into adware, ransomware, scareware and SMS malware. The malware families include 20 contemporary attacks such as SMSsniffer, FakeApp, RansomBO and Shuanet. More than 80 features were extracted from network traffic by means of CICFlowMeter-V3, Figure 2.

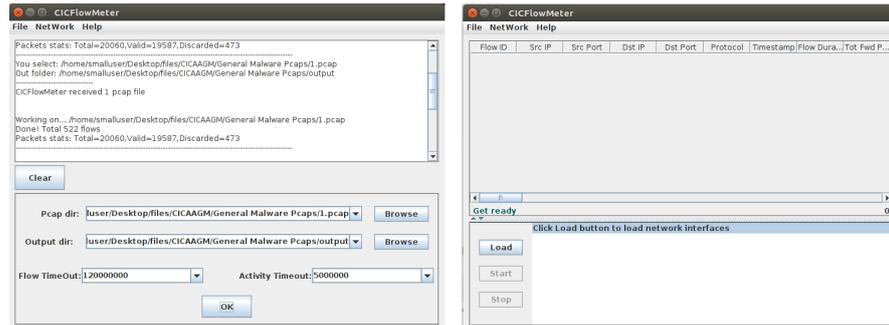


Figure 2. CICFlowMeter-V3 user interface in offline and online mode (on the right).

The dataset traffic include http, https, ftp and mail protocols used by the newest Android applications. We used the CICFlowMeter-V3 to extract feature values for a more detailed examination.

Similar to the CICIDS2017 dataset, we also evaluated the CICAndMal2017 dataset against the principles of 3.1. First, the standard deviation of feature values clearly deviated from zero. Second, the malware attacks from thousands of Android applications have been applied to the dataset during 2015-2017. Third, the protocols used by the newest applications have been applied. Fifth, the whole dataset has been labelled according to the malware types. Thus, this dataset fulfills the requirements of a modern IDS dataset.

3.4 CICAAGM

The CICAAGM dataset was created by Lashkari et al. for detecting adware and mobile malware [1] in 2008-2016. The aim was to create a dataset using real mobile devices, because some of the malware is able to hide itself from detection in emulator environment. This labeled dataset includes traffic of 1900 applications as well as malware and adware of 12 different categories. Applications such as AVpass, FakeFlash/FakePlayer,

GGtracker and Penetho were selected from Google Play according to popularity. Two different datasets were established, one for training/testing and the other for evaluation. The dataset includes 80 different features, and is publicly available as pcap files for network traffic and as csv files for the extracted traffic features.

The features of the dataset meet the first evaluation criteria in 3.1, because they have notable variety and clearly non-zero standard deviation. Second, the dataset involves 2090 new malware attacks [1]. Third, the applied protocols are enlisted only as TCP and UDP on network level. Fourth, the traffic captures of the adware, benign traffic and general malware groups are extensive. Fifth, the dataset downloads include a labeled feature set. These findings support the validity of the CICAAGM dataset for malware and adware detection systems.

3.5 AWID

The AWID dataset is publicly available via an application. It includes captured data generated by desktop computers, laptops, smartphones, tablets and smart TVs in a typical SOHO environment. The main focus covers attacks exploiting Wired Equivalent Privacy (WEP) protocol vulnerabilities [4]. The LAN traffic was infiltrated by the most frequent attacks on 802.11 such as key retrieving, ARP injection, dictionary, keystream retrieving and availability attacks [4]. The penetration against the SOHO network was implemented by means of Kali Linux, MDK3 and Metasploit. As a result, two labeled subsets AWID-CLS and AWID-ATK were recorded with full and reduced subsets. This dataset includes extensive sets of present attacks against the WEP protocol and can be used in WiFi network IDSs.

3.6 NSL-KDD

The NSL-KDD dataset is frequently used by IDSs in order to find anomalies in the network traffic [12]. The download comprises 2 Attribute-Relation File Format (ARFF) and text files both for train and test datasets. In addition to the full versions, these both sets are also available as 20% versions. The traffic flows are labelled either as normal or anomaly, Figure 3.

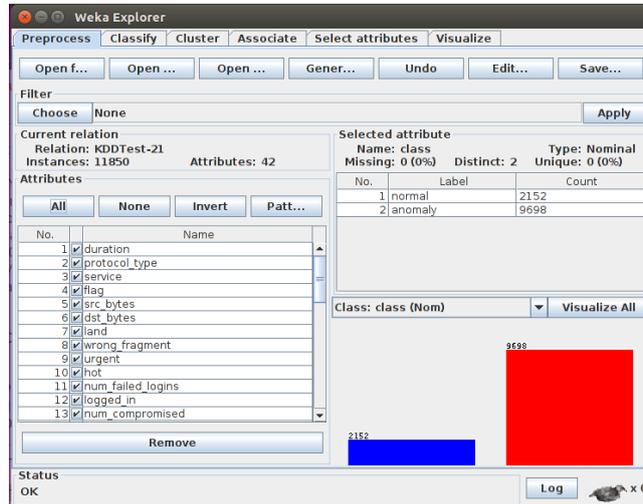


Figure 3. Proportions of benign and malicious traffic based on the test set of NSL-KDD.

A sample of a NSL_KDD text file is presented in Figure 4. It displays an excerpt of the features such as protocol type, duration, src bytes, dst bytes, flag and service.

```

1 13,tcp,telnet,SF,118,2425,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,00,0,00,0,00,0,00,1,00,0,00,0,00
2 0,udp,private,SF,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,3,0,00,0,00,0,00,0,00,0,75,0,50,0,00,255
3 0,tcp,telnet,S3,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,00,1,00,0,00,0,00,1,00,0,00,0,00,255
4 0,udp,private,SF,53,55,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,00,0,00,0,00,0,00,1,00,0,00,0,00
5 0,tcp,private,SH,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,00,1,00,0,00,0,00,1,00,0,00,0,00,16,1
6 0,tcp,http,SF,54540,8314,0,0,0,2,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,2,9,0,00,0,00,0,50,0,11,1,00,0,00,0,22,1
7 0,tcp,inmap4,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,228,9,0,00,0,00,1,00,1,00,0,04,0,06,0,00,255
8 7570,tcp,telnet,SF,0,44,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,00,0,00,0,00,0,00,1,00,0,00,0,00,2

```

Figure 4. Sample of a NLS-KDD text file.

The NSL-KDD dataset was constructed from the KDDCUP99 dataset by removing duplicates in the training set and repetitive data from the test set. Both datasets have totally 41 features extracted and calculated from the traffic data [6].

3.7 KDD'99

The KDD'99 dataset is the most frequently used and cited dataset. It applies buffer overflow attack and DoS attacks such as pod, Neptune and Smurf [10]. It has been generated for multiple-level hybrid classifier IDSs [5], Figure 5. At the first level, traffic is divided to DoS, Probe and Others. On the second level, the Bayesian clustering is able to sort the attacks to Normal, U2R and R2L [11]. The objective of R2L attack is to get illegal local access to the system, after which a U2R attack can be initiated to finally get access to the system as root user [19].

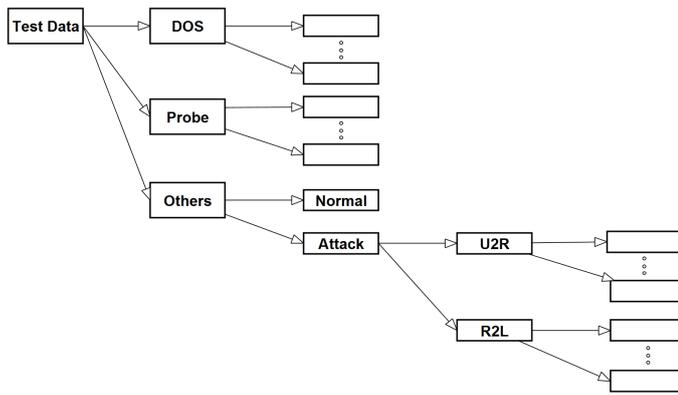


Figure 5. Multiple-level hybrid classifier [5].

3.8 KDDCUP99

The KDDCUP99 dataset was constructed for the KDD CUP Contest. It is a subset of the DARPA collection and it has been collected during nine weeks by simulation of normal military traffic in LAN network. The relative proportions of applied protocols are different in training and test data. There are 24 attacks in training dataset and, correspondingly, test data includes additional 14 attacks. The download include both full and %10 training and test datasets. The KDDCUP99 dataset includes over 40 features. All recent U2R and R2L attacks have been included in the training data as well as DoS and probe attacks [5]. It also includes probing attacks such as port scanning and surveillance.

3.9 DEFCON

The DEFCON dataset has a profile based approach to network breaches [3]. These profiles have been created considering intrusion details, applications and protocols. However, the dataset contains a very low proportion of benign traffic. The dataset consists of protocol related compositions, which include protocols such as tcp, udp, arp, ipv6, ipx, stp, icmp, ftp, http, dns and snmp. It has been collected during seven days. The malicious traffic was generated as multi-stage attack scenarios, which were constructed using insider, Http denial of service, DDoS and brute force attacks. The DEFCON dataset is available in pcap format.

3.10 NGIDS-DS

The NGIDS-DS dataset uses a qualitative modeling approach with the fuzzy logic system (FLS) [21]. It represents a new generation of datasets

[21]. The realism of this dataset is compared to other available datasets by means of completeness of traffic capture, coverage of current attacks and successful labelling of the traffic. The relational realism of NGIDS-DS is 0,95 while the other second and third most realistic databases are ISCX with 0,47 and DARPA with 0,33. The downloaded NGIDS-DS includes host log files, the pcap files and feature sets [21].

3.11 ISCX-IDS-2012

The ISCXIDS2012 dataset comprises testbed pcap and xml files. These files contain packets captured during seven days and they have been labeled either as normal or attack [22]. The traffic flow includes protocol such as http, ssh, email and ftp [18]. The infiltrated attacks are DoS, DDoS and Brute Force SSH [3], [18]. The dataset has been created for machine learning testing, evaluation and comparison purposes.

3.12 ISCX-2015

The ISCX2015 dataset was constructed in Brunswick University in Canada because of the need of an application level DoS attack dataset [8]. The ISCX2015 dataset includes 30 features, and it is available in pcap format. The DoS attacks infiltrated in the ISCX2015 dataset are DoS slow send and DoS slow read attacks. Other DoS low-volume attacks either use short-time pulses to overload the target host or by overloading target's resources such as Apache Range Header attack [8]. Totally ten different kinds of slow send and slow read attacks has been used. The dataset includes 84 features, but the label value is empty.

3.13 CSE-CIC-IDS2018

The CSE-CIC-IDS2018 dataset was created by planning detailed profiles and attack descriptions [6], [15]. The files are available in pcap and csv formats. Totally, the dataset included over 80 features such as flow duration, number of packets, minimum idle time and average size of packet. The dataset has been infiltrated with the attack types brute force, DoS, Web attack, infiltration attack, botnet attack, DDoS and portscan [15]. The traffic flows have been labeled according to the attack type or benign.

In the dynamic analysis, the the behaviour of the malware is monitored while the malware is running in sandbox or virtual environment. During malware recording, the dynamic analysis is stopped, when there are no

malware traffic detected. This way the efficiency of dynamic analysis can be increased. Malware attacks are often detected in bursts. The dynamic analysis tracks changes in the behaviour of source system and compares them against selected set of features.

4 Results

This section sums up the results of the evaluation in section 3. When the attacks in the datasets are concerned, the DoS, U2R, R2L and Prob attacks have been applied to most datasets. The DDoS attack was introduced in the DEFCON dataset in 2012, and recently it has been one of the main focuses in dataset creation. The application level attacks were involved for the first time by Shiravi et al. in DEFCON dataset in 2012 [3]. Lashkari et al. created traffic patterns for applications [1], when they constructed the CICAAGM dataset in 2017. Sharafaldin et al. monitored the communication behaviour of application level protocols while creating the CICIDS2017 dataset [10]. Also the current intrusions may include a series of consecutive attacks to hide the most serious attempt. Therefore, the significance of profiles for different applications and attack types has increased. Until 2017, the labelling was done only with labels normal and attack. NGIDS-DS dataset in 2017 was the first to use labels indicating the real names of the attacks.

5 Future Trends and Conclusions

Several future aspects have arisen during the evaluation. First, the proportion of mobile traffic increases continuously because of the high amount of new applications. Second, most of the new dataset attacks have been carefully planned by means of current publicly available IDS datasets. Therefore, the task of IDSs has become even more challenging. Third, these challenges require also IDS datasets to be constructed in controlled test environment to achieve the best results. Fourth, the nature of the attacks will change from brute force type attacks to specific purpose attacks, which need only reasonable processing resources.

References

- [1] A. Lashkari, A. Kadir, H. Gonzalez, K. Mbah, A. Ghorbani. Towards a Network-Based Framework for Android Malware Detection and Characterization. In *15th Annual Conference on Privacy, Security and Trust*, pages 233–242, 2017. doi: 10.1109/PST.2017.00035.
- [2] A. Lashkari, A. Kadir, L. Taheri, A. Ghorbani. Toward developing a systematic approach to generate benchmark android malware datasets and classification. In *International Carnahan Conference on Security Technology (ICCST)*, pages 1–7, Oct 2018. doi: 10.1109/CCST.2018.8585560.
- [3] A. Shiravi, H. Shiravi, M. Tavallaee, A. Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. *Computers & Security*, 31:357–374, 2012. doi:10.1016/j.cose.2011.12.012.
- [4] C. Koliass, G. Kambourakis, A. Stavrou, S. Gritzalis. Intrusion Detection in 802.11 Networks: Empirical Evaluation of Threats and a Public Dataset. *IEEE Communications Surveys Tutorials*, 18(1):184–208, 2016.
- [5] C. Xiang, C. Yong, L. Meng. Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees. *Pattern Recognition Letters*, 29(7):918–924, 2008.
- [6] G. Karatas, O. Demir, O. Sahigoz. Deep Learning in Intrusion Detection Systems. In *International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism*, pages 113–116, 2018.
- [7] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. Atkinson, X. Bellekens. A taxonomy and survey of intrusion detection system design techniques, network threats and datasets. *Association for Computing Machinery*, 1(1), June 2018. Available: <https://arxiv.org/pdf/1806.03517.pdf>.
- [8] H. Jazi, H. Gonzalez, N. Stakhanova, A. Ghorbani. Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling. *Computer Networks*, 121:25–36, 2017. Doi: 10.1016/j.comnet.2017.03.018.
- [9] H. Mohammed, M. Bahaa, I. Alyani, Z. Mohammed. A New Intrusion Detection System Based on Fast Learning Network and Particle Swarm Optimization. *IEEE Access*, 6:20255–20261, 2018. Doi: 10.1109/ACCESS.2018.2820092.
- [10] I. Sharafaldin, A. Lashkari, A. Ghorbani. Toward generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, pages 108–116, 2018.
- [11] M. Ali, B. Mohammed, A. Ismail, M. Zolkipli. A New Intrusion Detection System Based on Fast Learning Network and Particle Swarm Optimization. *IEEE Access*, 6(27 March):20255–20261, 2018.
- [12] M. Arafat, A. Jain, Y. Wu. Analysis of Intrusion Detection Dataset NSL-KDD Using KNIME Analytics, International Conference on Cyber Warfare and Security. In *Academic Conferences International Limited*, pages 573–583, 2018.

- [13] M. Bhuyan, D. Bhattacharya, J. Kalita. Network Anomaly Detection: Methods, Systems and Tools. In *IEEE Communications Surveys & Tutorials*, volume 16, 2014.
- [14] T. Shibahara, T. Yagi, M. Akiyama, D. Shiba, T. Yada. Efficient Dynamic Malware Analysis Based on Network Behavior Using Deep Learning. In *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016. doi:10.1109/GLOCOM.2016.7841778.
- [15] University of New Brunswick. CSE-CIC-IDS2018 on AWS. 2019. [Online]. Available: <https://www.unb.ca/cic/datasets/ids-2018.html>.
- [16] University of New Brunswick. Datasets. 2019.
- [17] University of New Brunswick. Intrusion Detection Evaluation Dataset (CICIDS2017). 2019. [Online]. Available: <https://www.unb.ca/cic/datasets/ids-2017.html>.
- [18] University of New Brunswick. Intrusion Detection Evaluation Dataset (ISCX2012). 2019.
- [19] V. Bulavas. Investigation of network intrusion detection using data visualization methods. In *2018 59th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*. IEEE, 2018. doi:10.1109/ITMS.2018.8552977.
- [20] Vasan K., Surendiran B. Dimensionality reduction using Principal Component Analysis for network intrusion detection. *Perspectives in Science*, 8:510–512, July 2016. Doi: 10.1109/j.pisc.2016.05.10.
- [21] W. Haider, J. Hu, J. Slay, B. Turnbull, Y. Xie. Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling. *Journal of Network and Computer Applications* 87 (2017), 87:185–192, March 2017. Doi: <http://dx.doi.org/10.1016/j.jnca.2017.03.018>.
- [22] W. Yassin, N. Udzir, Z. Muda, N. Sulaiman. Anomaly-based Intrusion Detection through k-means Clustering and Naives Bayes Classification. In *Proceedings of the 4th International Conference on Computing and Informatics, ICOC*, number 49, pages 298–303, August 2013.

Comparison of deep convolutional neural network for mobile applications

John Cheng

john.cheng@aalto.fi

Tutor: Jiayue Xu

Abstract

The usage of mobile phone is increasingly rising. At the same time, we enter a new era of Artificial Intelligence. As a direct consequence, the former makes use of the later more often than before. One application is the image classification where the user's pictures can be recognised without him or her indicating it.

In this paper, we are discussing three models that are used in this situation and comparing their performances in order to extract advices on which model to use depending on the main focus, namely, latency or accuracy while minimizing the computational cost.

The research shows that ResNet is the more relevant when accuracy is the main focus while MobileNets is recommended for latency.

KEYWORDS: Inception Network, ResNet, MobileNets

1 Introduction

With the Fourth industrial revolution, we have entered a new era where, facilitated by hardware improvements, Artificial Intelligence (AI) is omnipresent and steadily advances. Thanks to that, we have more computational power to manipulate data and come up with algorithms capable of finding patterns that would seem impossible in our eyes [7].

Those improvements can be seen as frightening to humans as we start to lose against robots at games like Go [1]. However, when we put those thoughts aside, AI is a helpful tool which can help save lives and enhance our everyday moments.

For the past few years, another change took place and mobile phone manufacturers have steadily improved their products, making them more powerful in every aspect; all arguing about having the best camera, display or processing power. A direct consequence of this increase in computing power lays in the way we capture moments, experiences of our respective lives. In fact, professional cameras are not needed anymore to take impressive landscape or portrait pictures [6].

When those two changes are combined, developers become able to adapt existing algorithms and create AI-enhanced smartphones. The word *adapt* is used for a specific reason. On the one hand, mobile phones are capable to achieve more than ever. On the other hand, they are still far from having the same processing prowess of a computer. Thus, algorithms can not be the same for both hosts due to the hardware differences [10].

Currently, modern deep convolutional neural network include many different implementation models such as MobileNets, InceptionNet, ResNet, and InceptionResNet. However, as those solutions arise, another problem appears. How can we decide which one is going to be the most efficient for our case? Such a delicate question will not come with an easy answer. Therefore, we will focus on three models and compare them in order to understand their characteristics and how to maximize their efficiency.

In a first part, we will explain the InceptionNet, followed by ResNet model and the MobileNets model. After that, we will analyze their performances. Finally, we will thoroughly compare them and advise on how and when to use each one of those.

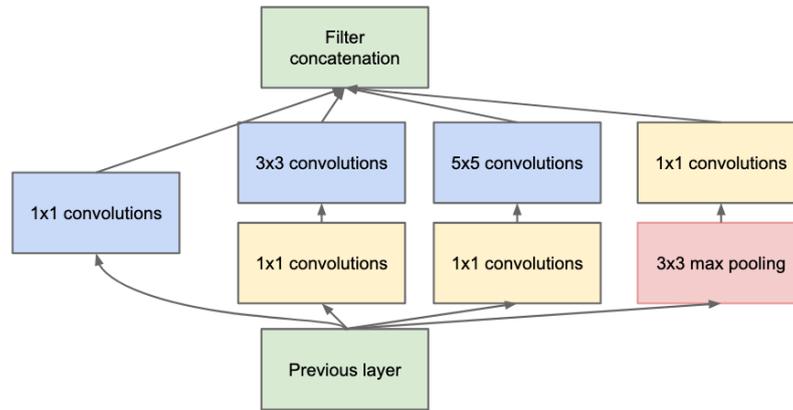


Figure 1. Inception Net module

2 Compared models

In this paper, three main models are going to be discussed in a chronological way in order to have a better understanding of each one of them as well as a view on how they have affected the way machine learning is used in low-resources situations. On the one hand, they share a common problem in that they are trying to solve vision problems in low-resources environment thus trying to find a trade-off between computational cost, latency and precision as efficient as possible. On the other hand, those models came up with different ways to cope with the situation. To start with, InceptionNet will be thoroughly explained. Then, ResNet will be discussed and in a final part, MobileNets model.

2.1 Inception Network model

In 2014, the Inception model won the ImageNet Visual Recognition Challenge and was an important milestone in the development of modern Convolutional Neural Network. One general assumption is that to improve your model, one can just increase its size, thus, increasing the depth and width. As a matter of fact, it seems like an easy way to have better result. Unfortunately, it is, at the same time, prone to overfitting as it implies having more parameters and the computational cost is increased as much as the size of the model.

Thus, a first step to resolve those issues is to move from fully connected models to a sparsely connected one and exploit this new structure to find a way to approximate an optimal local sparse structure and cover it with readily available dense components.

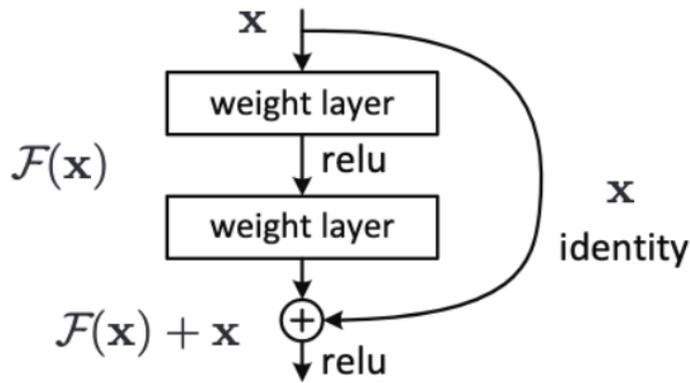


Figure 2. ResNet building block

The Inception Network model makes use of stacked modules composed of many expensive convolutions in the aspect of computational costs. As the idea is to use such architecture in a low computational power situation, a solution is needed to counter the cost. A way has been found and a previous step has been added to those expensive convolutions in order to reduce the dimension and then convolve with a large patch size (Fig 1). Thus, it is possible to increase the number of units for each layer while keeping the computation cost under control[9].

Every one of those modules are stacked on top of each other and can be compared to the way humans intuitively process visual information with different aggregated scales from which abstract features can be extracted from [8].

However, even though it seems natural to build a module composed only of Inception module, there is a benefice in using them only for the higher layers and stick with traditional convolutions otherwise as it improves the memory efficiency while training[9].

In a nutshell, InceptionNet has changed the architecture of traditional models by switching the focus from fully connected ones to more sparse models. By doing so, it helps to reduce the computational cost as it is able to drastically reduce the dimensions of the input while increasing the depth of the model.

2.2 ResNet model

ResNet or Deep Residual Network has been the winning model for the 2015 ILSVRC classification competition.

At the time, the assumption was that the deeper the model, the better

it will be. However, it could be observed that with the deepening of the architectures, there is the risk of saturating the accuracy and that this consequence wasn't caused by overfitting. Thus, it appeared that deepening a suitable model led to higher training errors.

That is how ResNet was created and resolved in its own way the degradation problem. It introduced the concept of residual learning framework. While normal model would try to fit a mapping of underlying stacked layers, in ResNet, they try to fit a residual mapping which corresponds to the difference between the normal mapping and the input, considering that they are both of the same dimensions. The reason behind this new fit comes from the observation that with deeper models, there is degradation and that this phenomena is caused by the solvers not being able to handle properly approximations of identity mappings by multiple nonlinear layers.

In a model with multiple layers, it is possible to realize this new fitting by feedforwarding with shortcut connections (Fig 2). Those shortcuts are skipping one or more layers and add the identity mapping to the output of the skipped layers. The main advantage of the identity mapping is that it doesn't add any parameter or modify the computational cost, thus, can easily be trained with Stochastic Gradient Descent with backpropagation without having to change to the solvers.

To summarize, ResNet has allowed the improvement of deep neural networks through a new fitting method that can be easily adapted to suitable models without much changes, thus, minimizing issues from the saturation of accuracy [4].

2.3 MobileNets model

MobileNets are open-source models proposed by Google in 2017 for efficient on-device vision. They are the most suitable for mobile and embedded based vision applications. In this section, we are covering the main characteristics the model is built on. In a first part, the depthwise separable filter will be explained followed by its two shrinking hyperparameters: the *width multiplier* for thinner models and the *resolution multiplier* for a reduced representation.

Depthwise Separable Filter

A Depthwise Separable Filter (DSF) or Depthwise Separable Convolution (DSC) differ from a standard convolution (Fig 3) as they are composed

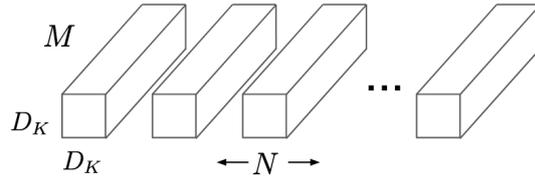


Figure 3. Standard Convolution

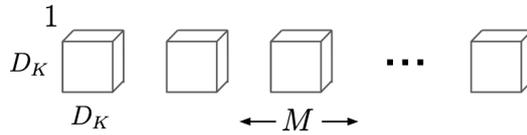


Figure 4. Depthwise Convolution

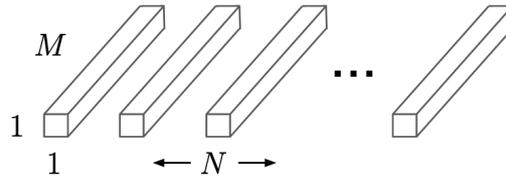


Figure 5. Point-To-Point Convolution

of two layers: a depthwise convolution (Fig 4) and a pointwise convolution (Fig 5). The main advantage is that DSF is differentiating the filtering step and the combining one. Thus, reducing the computation and model size [5].

The Depthwise Convolution is a kind of convolution composed of three main steps. In the first step, the input image and the filter are split into different channels and kept separate from each other. The number of channels has to be the same for both the input and the filter. In a second step, the input and corresponding filter are convoluted to produce a 2D tensor for each channel. Lastly, in a third step, those outputted tensors are stacked back together [2].

The Pointwise Convolution is then applied in order to combine and unify the new output.

After each of those parts of the DSF, a batchnorm and ReLU take place.

Shrinking Hyper-Parameters

MobileNets have been created to be used in small and low latency situation and even though they have achieved a spectacular job by introducing the two layers used to create a DSF, the *Width Multiplier* and *Resolution Multiplier* are two useful hyper-parameters that improves it even further

Model	Accuracy (top-5 accuracy)	Million parameters
Inception Network	89.3	6.8
ResNet	93	25.5
MobileNets	89.9	4.2

Table 1. Inception Network, ResNet and MobileNets accuracy

in a way that they allow to reduce the cost and parameters with a reasonable trade-off between latency, accuracy and size.

On the one hand, the Width Multiplier (α) is uniformly thinning the model through its different layers by multiplying the number of input channels as well as the number of output channels by the Width Multiplier. This parameter will always be lesser or equal to 1. Otherwise, it will not fulfill its responsibility and increase the size of our model.

On the other hand, the Resolution Multiplier's (ρ) purpose is to reduce the model internal representation. This parameter is used on the input image, thus, the following layers are made smaller by the same factor. In the same way and for the same reason as α , $\rho \in (0, 1]$.

Because of those hyper-parameters α and ρ , the computational cost and the number of parameters are quadratically reduced by roughly α^2 and ρ^2 and the computational cost of the model can be expressed as in Eq. (1).

$$D_K \cdot D_K \cdot \alpha M \cdot \rho D_F \cdot \rho D_F + \alpha M \cdot \alpha N \cdot \rho D_F \cdot \rho D_F \quad (1)$$

To conclude, MobileNet model is a big step forward for low-resources vision applications. In fact, while it increases the complexity of the architecture, it is able to drastically reduce the computational cost while keeping the latency, accuracy and depth under control.

3 Performances

The different models and their respective architectures explained, we can discuss their performances. Indeed, it is important to have a benchmark in order, later on, to compare them and make deductions as to why and when using a specific model is relevant. Thus, each model's performance are going to be summarized in term of accuracy and latency.

To add to that, in order to be comparable, the results here-after are all based on the ImageNet dataset. ImageNet is an image dataset organized

according to the WordNet hierarchy. ImageNet aims to provide on average 1000 images to illustrate each synset which are a meaningful concept in WordNet. Images of each concept are quality-controlled and human-annotated [3].

On the one hand, accuracy has been chosen as a point of comparison because it is easy to build a model. But it needs to actually be able to fulfill its purpose. And that is where this criteria is interesting as it reveals exactly if the trained model can perform correctly without care for efficiency.

On the other hand, we decided to focus on the number of parameters used as it directly influences the latency. Indeed, the more parameters, the slower the model.

4 Discussion

The goal of this paper is to give advice on which model to use based on a context, on assumptions and facts. The fact, and scope in this situation, concerns where the model is used which is on mobile devices with limited resources. Based on that, we will discuss two categories of users:

1. users want an experience that has as little latency as possible
2. users want to have a result as accurate as possible

As stated at the beginning, there is not one model for all. Each one of the presented model has its own characteristics, advantages and disadvantages. Nevertheless, they are all well performing models based on the ImageNet dataset and have all been recognised as state-of-the-art architectures. Thus, there are three situations that will be discussed here-after.

To start with, the first scenario focuses on latency or speed of execution. The user, in this situation, needs a result with little delay. Namely, parameters are the main cause of latency because the more parameters, the more accesses to the memory and thus, the more latency. We can now deduct that, for this scenario, the MobileNets model would be the most appropriate.

Secondly, the user is looking for a result as accurate as possible. It is then easily deductible that the ResNet model is the best for this scenario. As a downside, the model is also the one with the most parameters which

means that there is a risk of high latency.

Lastly, if we think about flexibility, MobileNets is the most convenient as it is possible to adjust its two hyper-parameters in order to fit the size of the model depending on the context.

5 Conclusion

Through this paper, we have explored three different models: Inception Net, ResNet and MobileNets. Their architectures have been discussed as well as their respective performances with the goal being to determine which model to use depending on the situation. The motivation behind this research has been to help developers to understand the trade-off between accuracy and latency under limited resources.

There has been a discussion and the following deductions have been done:

1. In a situation where the user is focusing on accuracy, the ResNet model is the most reliable.
2. In a situation where the user is focusing on latency, the MobileNets model is the right one.

References

- [1] AlphaZero: Shedding new light on the grand games of chess, shogi and Go.
- [2] Depthwise separable convolutions for machine learning - eli bendersky's website.
- [3] ImageNet.
- [4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition.
- [5] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv:1704.04861 [cs]*, April 2017. arXiv: 1704.04861.
- [6] <http://www.gadgetreview.com>. Decline of the Point and Shoot, Rise of the Smartphone Camera, June 2013.
- [7] Nicole M. Radziwill. Quality 4.0: Let's Get Digital - The many ways the fourth industrial revolution is reshaping the way we think about quality.

arXiv:1810.07829 [cs], October 2018. arXiv: 1810.07829.

- [8] Bharath Raj. A simple guide to the versions of the inception network.
- [9] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions.
- [10] Desmond Yuen. The Future of Artificial Intelligence Is Mobile Phones Market., November 2017.

A survey of verification tools for weak memory models

Natalia Gavrilenko

natalia.gavrilenko@aalto.fi

Tutor: Keijo Heljanko

Abstract

A concurrent program can exhibit more behaviours than a set of thread interleavings. The additional behaviours are caused by optimising hardware and compilers reordering memory accesses. Reordering rules vary between different types of processors, meaning that a behaviour forbidden by one processor can be observed on another. A formal definition of reordering rules and synchronisation guarantees is referred to as a memory model of a processor. Weak memory models are models which allow behaviours beyond the set of interleavings.

For application-level programmers, weak behaviours are typically hidden behind synchronisation primitives, such as locks. However, implementations of these primitives rely on a memory model of the underlying hardware. Additionally, hardware-level synchronisation is widely used in performance-critical routines, for example, in operating system kernels. Errors in implementations of locks and OS routines may cause malfunctioning of nearly all system incorporating them. Manual verification of weak concurrency is limited by the complexity of memory models and the number of behaviours growing exponentially. Verification tools can address these limitations for a subset of concurrent programs. This work surveys the state-of-art tools for verification of concurrent programs under weak memory models.

KEYWORDS: shared-memory concurrency, memory models, model checking, software verification

1 Introduction

A significant part of modern software, ranging from user-level applications to operating system kernels, relies on shared-memory concurrency. In such programs, concurrent threads can simultaneously access shared memory for reading and writing data. In order to guarantee consistency of written and observed values, accesses from different threads to the same memory address must be synchronised. Without synchronisation, concurrent shared-memory accesses can cause a *data race*: a situation when a thread reads or overwrites data that has been updated only partially. For example, a data race may occur if a thread gets interrupted after it started writing a non-atomic value and before the write has been finished. If another thread is scheduled at this point, then, without proper synchronisation, it can read the partially updated data.

The role of synchronisation is not limited to non-atomic memory accesses. In multicore systems, lack of synchronisation can lead to reading stale values from local caches even if the data has been updated atomically. Delayed propagation of updates allows to save processor cycles but complicates programming: a programmer must know when a value is guaranteed to be visible to all threads. Moreover, processors may reorder memory accesses in order to optimise performance. Whereas such reorderings increase processor utilisation, they can also break some concurrent algorithms. In order to prevent undesired reorderings, these algorithms require additional synchronisation, for example, with memory barriers.

Application programs usually enforce synchronisation via locks and monitors, which are available in nearly all high-level programming languages supporting concurrency. If used correctly, they guarantee absence of data races and assure visibility of memory updates to all threads. However, such synchronisation introduces significant overhead which is often unacceptable for performance-critical routines, for example, in operating system kernels. In this case, it might be preferable to use hardware-level synchronisation based on memory barriers and dependencies between instructions, which is much cheaper in terms of performance. Furthermore, implementation of locks and monitors themselves requires hardware-level synchronisation.

A specification of synchronisation guarantees for a particular processor architecture is referred to as a *memory model* of this processor. Until recently, memory models have been expressed in expository prose. Such definitions were often incomplete and leave open questions. The modern approach relies on mathematically rigorous definitions of memory models which can be used for formal verification. The notable memory models of the mainstream processors include [33] for

x86-TSO, [8, 21] for Power PC, [8] for ARMv7, and [17, 32] for ARMv8. The need for formal memory models has been also argued for alternative types of processing units, e.g., for Nvidia and AMD GPUs [4]. In addition to hardware memory models, high-level programming languages and operating system kernels may use memory models to define behaviors allowed by their runtimes. For programming languages, the widely used models are [26] for Java and [9] for C/C++. The Linux kernel model of [7] is currently a part of the kernel specification formally describing the behaviour of Linux concurrency primitives.

Memory models of modern processors are complex and sometimes even contain counter-intuitive rules. Due to the differences in memory models, the same algorithm might be correct on one processor, but demonstrate illegal behaviour on another one. Thus, a misinterpretation of a memory model can lead to dangerous errors in operating systems and other system software. Moreover, flaws in implementation of locks and monitors will propagate to the application programs relying on their correctness. The purpose of memory-model-aware verification is to prove correctness of a concurrent program in the context of a given memory model. Since manual construction of proofs is error-prone and limited to a small subset of the existing lock-free codebase, exhaustive verification requires automated tools.

The aim of this work is to provide a survey of tools which allow automated verification of concurrent programs in the context of memory models. The paper is structured as follows. Section 2 provides an overview of memory models and introduces the main concepts. Section 3 discussed the challenges and approaches to verification of concurrent software in general and in the context of weak memory models in particular. Finally, Section 4 provides an overview of the tools which can be used for reachability analysis under weak memory models.

2 Memory models

The sequentially consistent (SC) model defined by Lamport [25] is one of the first models of concurrent systems. It comprises concurrent threads and the global memory that can be accessed by all of the threads. The model has no intermediate cache buffers meaning that all threads interact directly with the memory. Since the memory can serve only one thread per time, each update of a shared variable is immediately visible to all threads. The model preserves the order of instructions within a thread, i.e., out-of-order execution is forbidden. As a result, in the sequentially consistent model, the set of legal executions is equivalent to interleavings of atomic instructions from different threads.

<i>thread 0</i>	<i>thread 1</i>	<i>interleavings:</i>	
$a : \text{write}(x, 1)$	$c : \text{write}(y, 1)$	a, b, c, d	c, d, a, b
$b : r_0 = \text{read}(y)$	$d : r_1 = \text{read}(x)$	a, c, b, d	c, a, d, b
$\text{exists } r_0 = 0 \wedge r_1 = 0$		a, c, d, b	c, a, b, d

Figure 1. Program SB

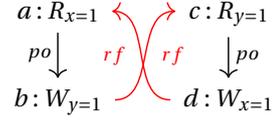


Figure 2. Axiomatic execution

The sequentially consistent model is not valid for modern processors. Instead, all mainstream architectures require more relaxed (weak) models, where updates of the shared memory can be delayed and/or reordered. This relaxation allows to reduce the overhead of cache synchronisation, thus yielding better performance. However, it also makes reasoning about the correctness of concurrent programs more difficult. Moreover, in weak models the set of possible executions is in general larger than in the sequentially consistent model, because each memory read may observe different values in the same interleaving. This increases the complexity of computer-aided verification since a larger set of executions must be checked.

The variety of weak memory models is best illustrated by the classic SPARC hierarchy [23]. It includes TSO (Total Store Order), PSO (Partial Store Order) and RMO (Relaxed Memory Order) models, ordered from the strongest to the weakest. Memory models of the other processing units and programming languages can be derived from these ones, but are not identical to them (with the exception of the x86-TSO [29]). Other models may include additional relaxations and, in opposite, strengthen a classic model by forbidding a behaviour in special cases.

The TSO model can be described as the sequentially consistent model with write buffers between threads and the shared memory [29]. All memory writes issued by a thread are placed into these buffers preserving their original order. A thread can read data only from its own buffer and from the shared memory. A read request issued by a thread is first checked against its buffer. If the buffer contains no writes to this memory address, then the request will be satisfied from the memory. At any moment, thread buffer can be flushed to the shared memory, making the updates visible to all other threads. A buffer can be also flushed explicitly by executing a memory barrier.

Figure 1 illustrates the difference between the sequentially consistent and the TSO models. For this program, the sequentially consistent model allows six executions, which are equivalent to the set of all possible interleavings. The behaviour with final values $r_0 = 0 \wedge r_1 = 0$ is forbidden since no interleaving leads to this result. However, the program can reach the final state $r_0 = 0 \wedge r_1 = 0$ if it is executed on TSO and the reads are satisfied before the buffers have been flushed.

The TSO, PSO and RMO memory models can be expressed in terms of reordering

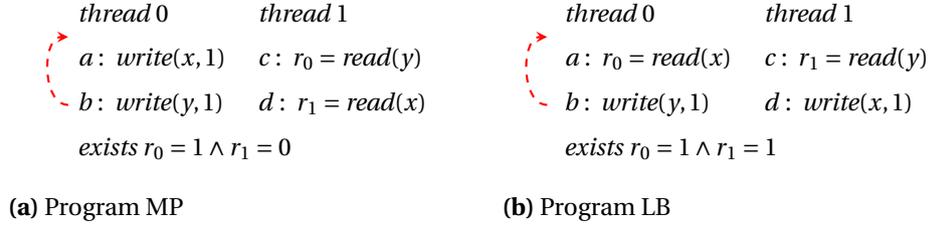


Figure 3. Memory model in terms of reordering

of events within a thread [23]. In this formalism, TSO can be defined as a model allowing execution of memory reads before program-order-earlier writes if they address different memory locations. The order of other pairs (read-to-read, read-to-write and write-to-write) is preserved. In terms of reordering, the behaviour in Figure 1 can be explained as follows. If event b is executed before the earlier event a , then the behaviour with final values $r_0 = 0 \wedge r_1 = 0$ becomes sequentially consistent. Since such reordering is allowed by TSO, the behaviour is legal.

The PSO model preserves the order of read-to-read and read-to-write pairs, but allows writes addressing different memory locations to overtake each other. Figure 3a demonstrates a behaviour with write-to-write reordering, which is legal on PSO but forbidden on TSO. The weakest RMO memory model allows reordering all *independent* memory events if they address different memory locations. Independence means that neither an address nor a value of a latter event depends on the value observed by an earlier read. An example of such reordering is shown in Figure 3b. This behaviour is forbidden by TSO and PSO, but arises on RMO. If a reordering leads to an undesired behaviour, it can be forbidden by inserting a memory barrier. However, memory barriers require additional processor cycles, thus, insertion of redundant barriers should be avoided.

All the above definitions of memory models are given in the operation semantics. Whereas this semantics is intuitive, it has several limitations. Firstly, operational models are rigid in the sense that forbidding or allowing a corner case behaviour may lead to significant changes to the model. Secondly, verification algorithms designed for one operational model are often inapplicable to other models or require substantial modifications. Axiomatic definitions of memory models have been developed to address these limitations. CAT [6] is a de-facto standard language for axiomatic memory models.

In axiomatic semantics, atomic instructions of a program are mapped to a set of *events*. Events may have different types, such as writes (to the shared memory), reads (from the shared memory), accesses to thread-local registers, branches, barriers and so on. Events are interlinked by control-flow *relations*. The exact set of these relations depends on the particular semantics. Examples of common

control-flow relations are *program order* (po) of events within a thread, and *control dependency* (ctrl) from a branch guard, such as an *if* clause, to events inside this branch. Control-flow relations are static in the sense that they persist in all possible executions of a concurrent program ¹.

Provided that a program contains only deterministic computations and has no side effects, its execution can be fully described ² by two communication relations [8]. The first one, *reads from* (rf) is a relation between a write (to the shared memory) and a read (from the shared memory) receiving a value from this write. The second, *coherence order* (co), refers to the total order of writes to the same memory location.

The control-flow and communication relations are present in all axiomatic memory models. They can be used to derive new relations via unary (complement, inverse, reflexive- and transitive closure) and binary (union, intersection, composition, setminus) operators. Illegal executions can be prohibited via constraints (acyclicity, irreflexivity, emptiness) on relations. For example, the behaviour of the program LB shown in Figure 2 can be axiomatically forbidden by constraint $\text{acyclic po} \cup \text{rf}$.

3 Verification methods

In general, software verification falls into two categories: dynamic testing and static analysis. With testing, behaviour of a program is observed during execution and potential errors are captured at runtime. Whereas testing is a powerful approach for verification of sequential (single thread) programs, it has limited applicability to concurrent software. If several threads are executed concurrently, the behaviour of a program depends on the scheduling. As a result, a concurrent program might reveal no bugs during testing, but fail at the next execution. For this reason, verification of concurrent software typically relies on static analysis of the source code.

Static verification includes deductive verification [15] and model checking [12]. Broadly speaking, deductive verification means transforming a program into a set of mathematical statements and constructing a proof showing that these statements conform to the specification. However, constructing a proof is usually tedious and requires complex domain-specific definitions of correctness, which prevents using deductive methods for routine verification of arbitrary programs.

Model checking refers to exploring reachable states and validating the states and

¹Subject to some limitations which are beyond the scope of this work.

²If a program does not contain events which can spuriously fail, e.g, ARM exclusive stores.

the transitions against a specification. In the base case, a state includes a set of values for each shared and thread-local variable. Execution of an atomic instruction in one of the threads, triggers a transition from one state to another. This approach allows straightforward verification of the *safety* property, that is, non-reachability of a bad state, such as deadlock. It can be also extended to verification of more complex constraints, such as absence of accesses to uninitialised memory. This work focuses on the verification of state(s) reachability.

In general, the reachability problem is undecidable [10]. The proof can be easily obtained by reducing the halting problem to the reachability of a final state. However, reachability is guaranteed to have a solution for a subset of programs. Firstly, it is decidable for acyclic programs (programs without loops) where all computations are deterministic and no side effects allowed³. Secondly, cyclic programs with deterministic computations are decidable if termination of their loops is decidable. For sequential cyclic programs and for cyclic programs under the sequentially consistent memory model, verification techniques are relatively well developed. However, generalisation of these techniques to arbitrary memory models is still an open problem. For this reason, the majority of automated verification tools for weak memory models either offer no support for cyclic programs or unroll cycles up to a given bound.

One of the challenges associated with model checking of concurrent software is the number of states growing exponentially with the number of concurrent threads and instructions per thread. Without additional optimisation, it limits verification to very small programs. Verification of larger programs requires optimisations targeting two constraints: memory and execution time. The memory constraint can be addressed by stateless algorithms, i.e., algorithms that do not explicitly store states. However, additional optimisations are still required in order to achieve feasible execution time.

Partial Order Reduction (POR) [20] allows to identify and merge executions which are equivalent in terms of their effect on the memory, thus reducing the verification space. For example, reordering independent instructions, such as writes to different memory locations, will not change their combined effect, meaning that these two executions can be merged. In its classic version, POR requires explicit representation of states and is limited to the sequentially consistent memory model. Dynamic Partial Order Reduction (DPOR) [16] can be used with stateless model checking algorithms. Several modifications of DPOR have been successfully applied to TSO, Power and a subset of the C11 memory model [1, 2, 3].

Satisfiability solving [11] is another approach to avoid state explosion during

³Provided that a memory model forbids out-of-thin-air reads.

Tool	Publications	Source code
HERD	Alglave et al. [8]	https://github.com/herd/herdtools7
DARTAGNAN	Ponce-de-León et al. [30, 31]	https://github.com/hernanponcedeleon/Dat3M
CBMC	Clarke et al. [13], Kroening and Tautschnig [24]	https://github.com/diffblue/cbmc
NIDHUGG	Abdulla et al. [1], Abdulla et al. [2]	https://github.com/nidhugg/nidhugg
TRACER	Abdulla et al. [3]	https://github.com/PhongNgo/tracer
RMEM	Flur et al. [17, 18], Pulte et al. [32]	https://github.com/rem-s-project/rmem

Figure 4. Selected tools

model checking. Like stateless algorithms, it does not require storing all states explicitly. Furthermore, it allows skipping exploration of some states if they are proven not to lead to the state(s) of interest. In this technique, a program is encoded into a Satisfiability Modulo Theories (SMT) formula together with a set of assertions that should be tested. The encoding must capture the control flow (instructions are executed iff the corresponding branching conditions hold) and the data flow (reads from and writes to shared and local variables are consistent) of the program. It should also include constraints of the memory model. The final SMT formula is supplied to a solver which must find a satisfying assignment or return a proof that no such assignment exist.

4 Verification tools

The aim of this section is to provide an overview of recent and/or widely used tools which support at least one memory model weaker than TSO. Figure 4 shows a list of selected tools with pointers to the corresponding publications and the source code. The main characteristics of the tools are summarised in Figure 5.

HERD [8] was the first tool allowing verification with arbitrary memory models. It accepts two inputs: an axiomatic memory model defined in CAT [6] and a *litmus* file. A litmus file comprises a program itself and a set of assertions on final states of the program. The source code of the program must be written in one of the languages recognised by HERD. Currently, HERD supports critical subsets of the following programming languages: x86, PPC, ARM, Aarch64 and RISC-V assembly, C (with the C11 atomics and a batch of Linux kernel primitives) and LISA [5].

Before testing, HERD converts a program into a set of internal events. For these events, it computes a set of *candidate executions*, that is, a set of all possible communications via the shared memory. Candidate executions are expressed in terms of the communication relations *rf* and *co* between the events accessing the same memory location. This representation allows to separate the data flow from the

scheduling. Since the scheduling is ignored, the amount of candidate executions is in general much lower than the number of interleavings. For each candidate execution, HERD calculates relations defined in the memory model and evaluates whether they violate the constraints of the model. If an execution is legal, then the states of registers and memory locations will be computed. This technique is compatible with nearly all axiomatic memory models. However, despite of the efficient representation of executions, it does not scale beyond small programs with a limited number of threads.

In addition to the basic weak behaviours, HERD provides limited support for mixed-size concurrency [18]. Mixed-size concurrency is an emerging research area. It aims to formalise weak behaviours of hardware, and potentially high-level languages and runtimes, in situations when unsynchronised memory accesses are not aligned to a machine word. In this case, a read event can have multiple rf edges starting at different writes. The coherence order of writes per location also gets a different meaning in this context. This can lead to behaviours which cannot be described by the classic aligned model. To the best of our knowledge, only two tools, HERD and RMEM (discussed later in this section), consider the mixed-size phenomena to some extent.

The functionality of **DARTAGNAN** [30, 31] is similar to that of HERD, except that the former does not support mixed-size accesses. However, in contrast to HERD, DARTAGNAN recognises pointer arithmetics on arrays. The tool accepts two inputs: a memory model defined in CAT and a program in the litmus format (except LISA). Programs can be also given in the *pts* format, which is a pseudo-language with C-like syntax and C11 atomics. Instead of enumerating executions explicitly, DARTAGNAN relies on the SMT-based bounded model checking [11]. It encodes both, a program and a memory model, into an SMT formula. The formula is evaluated by Z3 [14], the backend solver used by DARTAGNAN.

Before encoding, DARTAGNAN performs a three-step relation analysis [19]. At the first step, it computes an over-approximation of edges (event pairs) for each relation defined in a memory model. At the next stage, the over-approximated sets are compared against the constraints of a memory model. If an edge in the set cannot contribute to any constraint, then encoding of such edge is unnecessary. For example, it is safe to skip edges that cannot form a cycle if a relation is only used in the acyclicity constraint. At the last step, redundant event pairs are removed from the over-approximated sets. This procedure starts from the top relation of a constraint and recursively propagates to the base relations. For each relation, DARTAGNAN will encode only the edges contained in the over-approximated sets, i.e., only the edges which can yield an illegal execution. The resulting encoding is

Tool	Verification principle	Memory models	Program format	Mixed-size
HERD	enumeration	CAT	litmus	yes
DARTAGNAN	SMT	CAT	litmus, pts	no
CBMC	SMT	SC, TSO, PSO	c	no
NIDHUGG	guided simulation	SC, TSO, PSO, Power, ARM	c, ll	no
TRACER	guided simulation	C11 (limited)	binary (c)	no
RMEM	enumeration	Power, ARMv8, RISC-V, TSO	litmus, elf	yes

Figure 5. Main characteristics of the tools

smaller and can be evaluated by a solver more efficiently than a naive encoding. According to [19], the optimised version of DARTAGNAN outperforms HERD up to several orders of magnitude on large Linux kernel benchmarks.

CBMC [13] is another tool relying on SMT encoding. It was introduced over 15 years ago as a general-purpose model checker for C and C++ programs. In addition to the assertions on intermediate and final states, the tool can verify pointer and array safety. The support for the weak concurrency has been added to CBMC only recently [24]. Currently, the tool allows verification of C and C++ programs under SC, TSO and PSO memory models defined operationally. Weak behaviours are analysed at the level of uncompiled and unoptimised C or C++ code, to which the guarantees of the hardware memory models are applied.

By default, CBMC utilises a custom SMT solver built on top of MiniSAT⁴. However, the tool allows replacing of the custom solver with an alternative one. In contrast to DARTAGNAN which encodes arithmetics at the level of integers, CBMC uses bit-precise encoding of computations. In terms of recognised syntax, CBMC is a mature tool supporting almost all ANSI-C features, including arrays, pointers and data types of different length. However, mixed-size memory accesses are used only in verification of arrays and pointers safety, but mixed-size concurrency is not captured by the tool.

Like HERD, **NIDHUGG** [1, 2] tests reachability by exploring executions. It supports the operational memory models SC, TSO, PSO [1], Power [2] and partially ARM⁵. The tool accepts input programs written in C (with pthreads) and compiles them into LLVM assembly, whereas LLVM programs can also be passed directly. In order to explore reachable states, NIDHUGG simulates execution of a program. Essentially, it acts as a runtime which governs scheduling and memory behaviour according to the verification task. With naive implementation, this approach would suffer from state explosion, since all possible combinations of interleavings and memory interactions must be explored. In order to reduce verification space, NID-

⁴<http://minisat.se/>

⁵<https://github.com/nidhugg/nidhugg/blob/master/doc/manual.pdf>

HUGG adapts the DPOR algorithm [16] to the weak memory models.

The intuition behind applying DPOR to the weak concurrency is the following. The classic DPOR algorithm used with the sequentially consistent model relies on the transitive *happens-before* relation. A happens-before edge interconnects two events such that the event at the tip of the arrow depends on the event at the beginning of this edge. Dependency means that the latter event reads or overwrites the value written or observed by the former. Since weak memory models allow updates to propagate out-of-order, a legal weak execution can contain a cyclic happens-before relation. An example of such cycle can be found in Figure 2 (assuming that *po* and *rf* edges contribute to happens-before). This is incompatible with the DPOR algorithm which requires happens-before to be acyclic. To address this limitation, NIDHUGG introduces auxiliary *update* events. A happens-before edge between a read and a write is forced through an update event associated with the write. The update event preserves both parts of the happens-before edge if it is ordered from a write to a read, but cuts it in the opposite direction, thus breaking a potential cycle. Implementation details of the algorithm can be found in [1] for TSO and PSO and in [2] for Power.

TRACER [3] applies the DPOR-based optimisation of NIDHUGG to the release-acquire subset of the C11 memory model [9]. The semantics of the model are defined operationally in the source code of the tool. The tool is built on top of CDSHECKER [28], meaning that it reuses and partially modifies the source code of the latter. The main advantage of TRACER over CDSHECKER is the superior performance achieved by the adapted DPOR. According to [3], the number of explored executions and running times are up to two orders of magnitude better on TRACER than on CDSHECKER.

As input programs, TRACER accepts binary files compiled (with the tool's library included) from the supported subset of C or C++. Using binary formats for reachability analysis under the C11 memory model implies several limitations. Firstly, the semantics of the original program can be strengthened by the compiler if it does not support some of the syntactic elements. For example, this approach might be inapplicable to the verification of the *consume* semantics, which most compilers do not distinguish from *acquire*⁶. Secondly, the tool will not generate behaviours with C11 *satisfaction cycles* since they are not observed on the real hardware [28]. Thirdly, the tool might generate false positives if dependencies present in the original code have been optimised by the compiler [28].

RMEM [17, 32] (previous name PPCMEM) allows to test programs against operational models. It supports the Flat [32], the Flowing [17] and the POP [17] models

⁶<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2016/p0371r1.html>

of ARMv8 and the Power model of [21]. The tool also has experimental support for x86-TSO [29] and RISC-V RVWMO [34] models. The models are expressed in Lem [27].

As an input, RMEM accepts litmus tests (PPC, Aarch64, RISC-V and x86 dialects) and small *elf* binaries. Before execution, input programs are translated into the intermediate ISA SAIL [22], a custom instruction set of RMEM. One of the distinctive features of the tool is the support for mixed-size and misaligned memory accesses presented in [18]. Reachability testing is conducted via exhaustive search through all valid executions, which limit scalability of the tool. In addition to the exhaustive search, the web version of RMEM ⁷ provides an interactive graphical interface allowing a user to execute a program manually. Information on executions returned by RMEM is the most detailed among the tools discussed in this work.

5 Conclusion

Whereas weak behaviours of concurrent programs have been studied for several decades, considerable progress has been made during the recent years. An important achievement is the CAT language allowing to express almost any memory model succinctly. Other advancements are concerned with the emergence of the new memory models. Since the introduction of the C/C++ memory model in the C11 standard, the low-level concurrency can be expressed directly in a high-level language without inlining assembly. Formalisation of the Linux kernel memory model has led to the increase of the lock-free codebase in the kernel. Formal definitions of hardware memory models provided means for automated verification of multiple concurrent algorithms.

Despite the expansion of the software relying on the weak concurrency, the tool support is lagging behind. One of the limitations is the scalability of the verification algorithms. In spite of the novel optimisation techniques, even the most scalable state-of-art tools are far from verification of concurrent programs with thousands lines of code. The second constraint is related to the semantics of the programs recognised by the tools. Only CBMC and (to some extent) NIDHUGG and TRACER can be applied to the real-world programs, while other tools are limited to the toy semantics. Thirdly, all existing tools except HERD and DARTAGNAN support only a fixed set of operational models. Moreover, some of the tools rely on the simplified versions of the actual models. Addressing these limitations in the future development is essential for the routine verification of the real-world software.

⁷<https://www.cl.cam.ac.uk/~sf502/regressions/rmem/>

References

- [1] P. A. Abdulla, S. Aronis, M. F. Atig, B. J., C. Leonardsson, and K. Sagonas. Stateless model checking for TSO and PSO. *Acta Inf.*, 54(8):789–818, 2017.
- [2] P. A. Abdulla, M. F. Atig, B. Jonsson, and C. Leonardsson. Stateless model checking for POWER. In S. Chaudhuri and A. Farzan, editors, *CAV*, volume 9780 of *LNCS*, pages 134–156. Springer, 2016.
- [3] P. A. Abdulla, M. F. Atig, B. Jonsson, and T. P. Ngo. Optimal stateless model checking under the release-acquire semantics. *PACMPL*, 2(OOPSLA):135:1–135:29, 2018.
- [4] J. Alglave, M. Batty, A. F. Donaldson, G. Gopalakrishnan, J. Ketema, and D. Poetzl et al. GPU concurrency: Weak behaviours and programming assumptions. In Ö. Özturk, K. Ebcioğlu, and S. Dwarkadas, editors, *ASPLOS*, pages 577–591. ACM, 2015.
- [5] J. Alglave and P. Cousot. Syntax and analytic semantics of LISA. *CoRR*, abs/1608.06583, 2016.
- [6] J. Alglave, P. Cousot, and L. Maranget. Syntax and semantics of the weak consistency model specification language Cat. *CoRR*, abs/1608.07531, 2016.
- [7] J. Alglave, L. Maranget, P. E. McKenney, A. Parri, and A. S. Stern. Frightening small children and disconcerting grown-ups: Concurrency in the Linux Kernel. In X. Shen, J. Tuck, R. Bianchini, and V. Sarkar, editors, *ASPLOS*, pages 405–418. ACM, 2018.
- [8] J. Alglave, L. Maranget, and M. Tautschnig. Herding cats: modelling, simulation, testing, and data-mining for weak memory. In M. F. O’Boyle and K. Pingali, editors, *PLDI*, page 40. ACM, 2014.
- [9] M. Batty, S. Owens, S. Sarkar, P. Sewell, and T. Weber. Mathematizing C++ concurrency. In T. Ball and M. Sagiv, editors, *POPL*, pages 55–66. ACM, 2011.
- [10] R. S. Boyer and J. t. Moore. A mechanical proof of the unsolvability of the halting problem. *ACM*, 31(3):441–458, 1984.
- [11] E. M. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. *Formal Methods in System Design*, 19(1):7–34, 2001.
- [12] E. M. Clarke, T. A. Henzinger, and H. Veith. Introduction to model checking. In E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem, editors, *Handbook of Model Checking*, pages 1–26. Springer, 2018.
- [13] E. M. Clarke, D. Kroening, and F. Lerda. A tool for checking ANSI-C programs. In K. Jensen and A. Podelski, editors, *TACAS*, volume 2988 of *LNCS*, pages 168–176. Springer, 2004.
- [14] L. M. de Moura and N. Bjørner. Z3: an efficient SMT solver. In C. R. Ramakrishnan and J. Rehof, editors, *TACAS*, volume 4963 of *LNCS*, pages 337–340. Springer, 2008.
- [15] J.-C. Filliâtre. Deductive software verification. *STTT*, 13(5):397–403, 2011.
- [16] C. Flanagan and P. Godefroid. Dynamic partial-order reduction for model checking software. In J. Palsberg and M. Abadi, editors, *POPL*, pages 110–121. ACM, 2005.
- [17] S. Flur, K. E. Gray, C. Pulte, S. Sarkar, A. Sezgin, L. Maranget, W. Deacon, and P. Sewell. Modelling the ARMv8 architecture, operationally: concurrency and ISA. In R. Bodík and R. Majumdar, editors, *POPL*, pages 608–621. ACM, 2016.

- [18] S. Flur, S. Sarkar, C. Pulte, K. Nienhuis, L. Maranget, K. E. Gray, A. Sezgin, M. Batty, and P. Sewell. Mixed-size concurrency: ARM, POWER, C/C++11, and SC. In G. Castagna and A. D. Gordon, editors, *POPL*, pages 429–442. ACM, 2017.
- [19] N. Gavrilenko, H. Ponce-de-León, F. Furbach, K. Heljanko, and R. Meyer. BMC for weak memory models: Relation analysis for compact SMT encodings. Submitted for publication, 2019.
- [20] P. Godefroid. *Partial-Order Methods for the Verification of Concurrent Systems - An Approach to the State-Explosion Problem*, volume 1032 of *LNCS*. Springer, 1996.
- [21] K. E. Gray, G. Kerneis, D. P. Mulligan, C. Pulte, S. Sarkar, and P. Sewell. An integrated concurrency and core-ISA architectural envelope definition, and test oracle, for IBM POWER multiprocessors. In M. Prvulovic, editor, *MICRO*, pages 635–646. ACM, 2015.
- [22] K. E. Gray, P. Sewell, C. Pulte, S. Flur, and R. Norton-Wright. The Sail instruction-set semantics specification language, 2017.
- [23] SPARC International Inc, D. L. Weaver, and T. Germond. *The SPARC architecture manual*. Prentice-Hall, 1994.
- [24] D. Kroening and M. Tautschnig. CBMC - C bounded model checker - (competition contribution). In E. Ábrahám and K. Havelund, editors, *TACAS*, volume 8413 of *LNCS*, pages 389–391. Springer, 2014.
- [25] L. Lamport. How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs. *IEEE Trans. Computers*, 28(9):690–691, 1979.
- [26] J. Manson, W. Pugh, and S. V. Adve. The Java memory model. In J. Palsberg and M. Abadi, editors, *POPL*, pages 378–391. ACM, 2005.
- [27] D. P. Mulligan, S. Owens, K. E. Gray, T. Ridge, and P. Sewell. Lem: reusable engineering of real-world semantics. In J. Jeuring and M. M. T. Chakravarty, editors, *ICFP*, pages 175–188. ACM, 2014.
- [28] B. Norris and B. Demsky. A practical approach for model checking C/C++11 code. *ACM Trans. Program. Lang. Syst.*, 38(3):10:1–10:51, 2016.
- [29] S. Owens, S. Sarkar, and P. Sewell. A better x86 memory model: x86-TSO. In S. Berghofer, T. Nipkow, C. Urban, and M. Wenzel, editors, *TPHOLS*, volume 5674 of *LNCS*, pages 391–407. Springer, 2009.
- [30] H. Ponce-de-León, F. Furbach, K. Heljanko, and R. Meyer. Portability analysis for weak memory models. PORTHOS: one tool for all models. In F. Ranzato, editor, *SAS*, volume 10422 of *LNCS*, pages 299–320. Springer, 2017.
- [31] H. Ponce-de-León, F. Furbach, K. Heljanko, and R. Meyer. BMC with memory models as modules. In N. Bjørner and A. Gurfinkel, editors, *FMCAD*, pages 1–9. IEEE, 2018.
- [32] Christopher Pulte, S. Flur, W. Deacon, J. French, S. Sarkar, and P. Sewell. Simplifying ARM concurrency: multicopy-atomic axiomatic and operational models for ARMv8. *PACMPL*, 2(POPL):19:1–19:29, 2018.
- [33] P. Sewell, S. Sarkar, S. Owens, F. Zappa Nardelli, and M. O. Myreen. x86-TSO: a rigorous and usable programmer’s model for x86 multiprocessors. *Commun. ACM*, 53(7):89–97, 2010.
- [34] A. Waterman and K. Asanovi. The RISC-V instruction set manual. Volume 1: User-level ISA, Version 2.2. *EECS Department, UC Berkeley*, 2017.

Virtual and Augmented Reality in the Industrial Sector

Robert Lee Seligmann

robert.seligmann@aalto.fi

Tutor: Antti Ylä-Jääski

Abstract

Currently, Virtual and Augmented Reality have entered the consumer market, leading to the development of increasingly advanced devices. The promotion of such devices seems to mainly focus on the entertainment industry, although these technologies have found applications that lie beyond this industry. This paper will research recent scientific literature on the topic and survey the different research and applications of these technologies in other industries, mainly focusing on the industrial sector. The aim of this paper is to understand how the architecture, engineering, construction, and manufacturing industries are applying them, and to determine if these technologies have sufficiently matured to be used in professional operations. Despite the technologies' current limitations, the review finds that virtual and augmented reality are effective tools to support communication, decision-making, design evaluation, and training, thus leading to the reduction of errors, costs, and accidents.

KEYWORDS: *Virtual Reality, Augmented Reality, Construction, Manufacture*

1 Introduction

Although it may appear that Virtual and Augmented Reality are recent technologies, there is existing research and prototypes dating as far back as the late 60s [3]. However, those were limited by the hardware at that time. Since then, hardware performances, screen resolution and size have considerably improved, particularly in the field of mobile phone technologies. Thus, Augmented Reality (AR) has become accessible through the majority of smartphones, while Virtual Reality (VR) head-mounted displays (HMD) are being fully commercialized for consumers.

These technologies have now been endorsed by the consumers; yet, they are often associated with gaming or entertainment. However, VR and AR have a wider spectrum of applications that extend beyond the scope of entertainment, enabling new business and market opportunities across different industries. To remain competitive, companies are researching innovative ways of integrating new and emerging technologies to improve on their current system. Construction and manufacture have complex processes where any mistake can potentially cause delays, thus increasing the costs [16]. Moreover, mastering skills in these disciplines requires training, but real-life practice may be costly and unsafe. To solve these issues, several businesses have started experimenting with these technologies and, in some cases, successfully integrated them in their workflow.

The aim of this paper is to deepen our knowledge in the topics by first, providing a state-of-the-art survey on both technologies and, secondly, by reviewing how industrial companies are innovatively applying these technologies. Through this survey, we will see that virtual and augmented technologies can be applied effectively even outside the entertainment industry. In some cases, they can surpass other technologies, by providing a virtual environment that can be used from product planning to evaluation, as well as a training tool.

The paper is organized as follows. The next section provides an overview of both VR, AR, and an outline of the different industries that use them. Section 3.1 focuses on the applications of XR¹ technology in the construction industry, followed by the manufacturing industry in Sec. 3.2. Afterwards, broader insights are described in Sec. 4. Finally, the conclusions are drawn.

¹Cross Reality (XR) will be used in this report to refer both VR and AR

2 Technology survey

2.1 Virtual Reality

Virtual Reality immerses the user in a computer-generated virtual environment in which they can interact using input devices. The first notion of virtual reality has existed since the 20th century, and the first immersive headset, named the "Sword of Damocles", was created by Ivan Sutherland in the 60s [3]. Since then, it has often been used for medical or military training, but the technology was still too experimental to be democratized. Research on this technology stalled over the years until a Kickstarter campaign for an affordable and effective VR headset named Oculus Rift ² started in 2013. This campaign reignited people's interests, leading businesses to start producing their own consumer-grade headsets.

Since 2013, various headsets of varying qualities have been released, and more are scheduled to be released in the upcoming year. Price and features also vary, ranging from low-end (made of cardboard for smartphone use) to high-end headsets that offer an increased level of immersion through a full six Degrees Of Freedom (DOF). The latter is more interesting as it offers a higher level of immersion by using additional input devices such as captors that track the position and rotation of the headset and controllers. They amplify the immersion and the interactivity, allowing more complex actions in the virtual environment.

Recently, one of the main issues regarding the screen resolution has been addressed when the Finnish company Varjo released its human-eye resolution headset in late February. The quality of a simulation will now depend on the quality of the models which can potentially, with the graphics capabilities currently available, approaches a realistic representation.

2.2 Augmented Reality

In contrast to Virtual Reality, Augmented Reality superimposes computer generated elements on top of the real world [17]. The technology also derived from the work of Ivan Sutherland, but the term "Augmented Reality" originated from the work of Caudell and Mizell [8] at Boeing who suggested a solution to help workers assemble and install wires in an aircraft using a heads-up display (HUD) headset. Currently, AR devices still

²<https://www.kickstarter.com/projects/1523379957/oculus-rift-step-into-the-game>

includes HUD headset; however, the technology is now often integrated to handheld devices such as smartphones or tablets. The integration is possible due to the significant improvement of mobile phone hardware. The powerful hardware and the high-definition camera allow to replicate a similar experience to an AR headset.

To superimpose virtual objects over the real world, an AR device must first recognize the physical world. There exist a few approaches on how to handle recognition [19]. The easiest ones are marker-based (e.g., images or objects) or location-based. The most interesting approach is Simultaneous Localization and Mapping (SLAM) which estimates the camera's position while creating a 3D point cloud of the environment, thus allowing to create dynamic experiences. Low latency and the quality of the tracking are paramount for augmented reality; as the viewer sees both the real world and the 3D objects, it is primordial that both are rigidly interlocked together to preserve the feeling of immersion. If the latency is too high or if the tracking is too poor, the 3D object may, respectively, move slower than the user's movement or stagger around its intended position.

2.3 Industry Applications

This section will overview some of the dominant industry applications of the technology, excluding construction and manufacturing which will be discussed in more detail in the next section.

Entertainment

Entertainment is the most mainstream usage of VR and AR. However, it not limited to only video games and movies. Museums and exhibitions, such as the National Museum of Finland [12] and the Peterson Automotive Museum³ have offered virtual interactive experiences. In the former case, visitors can travel back inside R. W. Ekman's painting 'The Opening of the Diet 1863 by Alexander II' to explore the Imperial Palace and interact with the crowd. The latter utilizes an AR application for story-telling, entrancing the visitors with virtual visualizations (e.g., the interior components of the car) blended with the cars.

³www.developer.microsoft.com/en-us/windows/stories/petersent-museum-hololens

Healthcare

The healthcare industry has utilized XR from medical training to treatments for patients. Virtual Reality therapy facilitates the treatment of fears and disorders, such as phobias and post-traumatic stress disorder, by replicating a scenario that triggers the fear [21]. This allows the patients to slowly and safely adapt to their fear. Moreover, it can reduce chronic pain as well as the psychological pain perceived in patients undergoing surgery. Other research demonstrated that it can be a tool for patient with cognitive and physical challenges to aid them in their rehabilitation. The Finnish company Osgenic⁴ develops virtual simulations where surgeons can safely practice their skills and procedures without endangering the patient. Additionally, 3D visualization can easily be modelled from the Magnetic resonance imaging (MRI) scan which could be used for simulations or as a reference model viewed with AR.

Military

The military has been utilizing virtual combat simulators for many years to simulate aerial, ground, and aquatic vehicles [18]. Soldiers are expected to put their lives on the line; therefore, extensive training is required for them to safely accomplish their mission. Mission Rehearsal Exercises (MRE) can be executed with VR by simulating a virtual battlefield where the soldier will be faced with a specific mission to complete. They can train to make decisions in a stressful condition and experience the environment without risk exposure. Commanders can also train using AR, the information of the battlefield (e.g., troupes, civilians, terrain map) can be virtualized to give an overview of the entire field which helps make strategic decisions and plan future actions.

Marketing

XR can provide a more engaging and impactful experience, encouraging the viewer to take on the next step, e.g., the *IKEA Place* application allows the user to "place" IKEA furniture in the user's space at true scale, helping them visualize and decide which furniture to purchase. The Finnish company Zoan⁵ is currently creating a high-definition 3D representation of the city of Helsinki, aiming it to be the "Virtual Capital of the World". People from other countries will be able to visit a detailed virtual version of Helsinki at the same time.

⁴<https://www.osgenic.com/>

⁵<https://zoan.fi/work/virtual-helsinki/>

3 Industrial applications

This section will describe the state of XR in the industrial sector and its research and applications.

3.1 Architecture, Engineering, and Construction

Over the years, the industries of Architecture, Engineering and Construction (AEC) have become increasingly complex and require a tremendous amount of communication, collaboration, and assessment, thereby increasing the need for information and visualization technologies [17]. The construction industry is one of the largest industrial employers in the European Union (EU), with approximately 12 million employees [11]. This represents a major sector for the EU, generating 9.8% of the Gross Domestic Product and employing 7.1% of the European workforce. However, they perform poorly compared to other industries. This can partly be attributed to their tendency to fail in exploiting the potential of new technologies and practices, thereby potentially compromising any possible improvement regarding safety, employee quality of life, competitiveness, cost, and productivity. This can also be attributed to the conservative nature of the sector and the mentality of the workforce who are often unprepared or unwilling to embrace such technologies. As humans tend to be creatures of habit, the implementation of such peculiar devices into their workflow could be frowned upon.

Pre-construction

The first aspect to review is the design and planning phase of the project. A construction process relies heavily on efficient communication across the whole chain. Any miscommunication can cause delays which inevitably end in unexpected additional costs. These conversations require special care, particularly with the client. Typically, the design phase starts with a 2D drawing of the building produced by the architect [10]. During construction, a significant number of alterations occur; these are often last minute changes from the client due to their difficulties to fully understand the design from the abstract 2D sketches.

Davidson and al. [10] suggested a VR application that recreates the design into a 3D representation, combined with a tool known as Bill of Quantities (B/Q) which is used to estimate the cost of all the items necessary for the construction. Using the application, the client can navigate through

the 3D representation at world-scale and understand how the building, including the furniture, will look like in the real world. Meanwhile, the client can request changes that are directly reflected in the designs (similarly to Building Information Modeling). In the end, the B/Q report is generated, taking into account the changes made, and will inform the client on the financial status of the project. This process enables a closer relationship with the client who will have more appreciation and confidence towards the project as the product will look closer to their vision. This can also save time and money during the construction by avoiding last minute changes caused by misinterpretations.

Construction

Whereas the previous phase took advantage of VR, the construction phase is more suited for AR technology. Augmented reality does not require a dreadful setup like VR; instead, AR devices are more lightweight and mobile whilst still allowing to beware of the surroundings. This, however, implies that the computing power is lower, reducing the graphics quality. However, the most important part remains the spatial tracking quality, which is not perfectly accurate but should improve over the years [17].

Currently, mapping a 2D drawing to the physical world is challenging; yet, engineers still rely on the drawings to make decisions and inspections, thereby leaving the possibility for human errors to occur [1]. These errors could possibly multiply as the project advances, leading to a waste of time and money. To avoid these issues, Augmented Reality can be utilized as a monitoring or planning tool [1, 4]. Fig. 1 compares the different visualization methods as seen in [15]. For planning, engineers can display reference models (e.g., beams, walls, or columns) at the location of the real-world position [14]. These visualizations provide a better representation to work with than 2D drawings and can serve as a foundation to start building faster and more accurately. Once the process has started, they can continue to be utilized to monitor the progress of the project by comparing the planned versus built representation. This allows someone to review what has been achieved in accordance with the one expected in the schedule [15]. To improve communication, the work in [22] suggested to virtually annotate messages to the physical object. Therefore, other technicians can access the information when inspecting the area.

Similar to the previous phase, this leads to early defect detection and better communication, ultimately saving time and costs. During post-

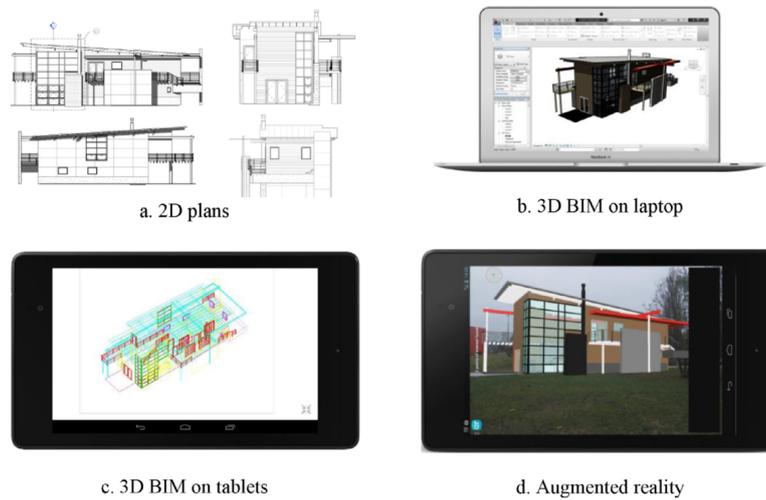


Figure 1. Comparison of the presentation methods by Meza and al. [15]

construction, maintenance procedures can be facilitated through Augmented Reality [20]. Some procedures may be dangerous or complex; therefore, solutions have been developed to guide the user throughout the tasks. The visual guide can be overlaid directly on top of the real-world object, highlighting the current operation to perform and displaying the necessary details to accomplish the tasks. This can be useful for skilled engineers, but students are the ones who would benefit the most from it.

Education

VR and AR have both been sought out as educational tools. Most lessons are taught in a classroom with very few on-site training [20]; thus, the students are not completely involved in the process. On-site exercises is then a critical step to acquire experience; however, it exposes the students to perilous activities in an industry with a high rate of accidents. These incidents are likely due to the lack of safety training provided on site and in class. Class training is generally taught with a PowerPoint presentation and videos. Thus, they lack a realistic depiction of the conditions of a site. The limited interactivity does not sufficiently engage the students for them to completely develop an understanding of the topic. Thus, VR can be utilized as a tool to bridge the gap between classroom and on-site training by providing a risk-free environment simulation. Whereas AR can enhance on-site training by projecting visualizations which can help them get a deeper understanding.

The research in [7] studied the benefits of Augmented Reality on 3 groups of students. The results demonstrate that the group that accessed AR videos in addition to the standard course material performed the best

on the final assignment. Another research in [20] explored how VR can raise the students' motivation and engagement by providing an interactive training simulation that can be used for safety training, operational training, and structure analysis. In the simulation, students can learn through various scenarios that can be customized to implement unexpected issues such as flaws from previous work or safety hazards [11]. By implementing such features, they can practice in an environment that allows incidents to occur without the risk of injury. This means they are able to see the consequences of their actions firsthand and reflect on their decisions. The pedagogical aspect can also be extended to other industries.

3.2 Manufacture

The manufacturing industry is in some aspects very similar to the construction industry. Product design is the activity that will largely influence the whole product development cycle and the final product [16]. Any flaws in the design decision will lead to unexpected costs and redesigns. Similarly to the AEC industry, XR can be adopted in the early stages of the product development cycle for evaluation and decision-making. Product design is a complex process lying beyond creativity and sketches, there are many aspects to consider such as the trends and opinions of the market, current technologies, and the resources at their disposal; therefore, it is necessary to recruit individuals with various background to collaborate on the product. Efficient communication is required to ensure that everyone shares the same level of knowledge to work and make valuable decisions.

The case study in [5] observed if VR can be utilized as a decision-making tool. The study was carried out with 5 design and manufacturing engineers with no prior experience with XR. Their task was the renovation of an existing structure in the company's factory. Using VR, they were able to quickly identify and correct issues with the current design that would be hard to detect otherwise. It granted various benefits, including the reduction of physical prototypes, improved product, and time-saving. Additionally, it has improved the engineers' engagement. The standard conference room reunion remains a passive experience, usually containing laptops or other devices that are likely to distract the people. Instead, the VR environment allowed an active experience where they were able to group together, observe, evaluate, and discuss about the designs.

Although the study in [5] investigated the usefulness of VR within a

company that is not familiar with the technology, other companies have fully integrated it into their design process. The research by [6] interrogated the practices of 25 companies currently applying VR to solve real-life use cases. A major percentage of the companies were related to the automotive sector where few have been utilizing VR before it has been popularized at the consumer level. One of the most discussed scenarios evaluated the visibility of the driver/passenger with a specific posture or setting. It is an important factor to access and would normally require building a clay model of the frame. By viewing the vehicle from the inside in VR, engineers are able to evaluate the visibility during movement or interaction. In general, the structure of a vehicle holds 6 pillars (or supports); therefore, it is crucial to perceive their size and position relative to the driver to understand how they influence their external vision. Larger pillars improve the safety of the vehicle but reduce the driver's visibility, and vice-versa. Another common scenario is the study of the instrument panel, VR can be utilized to evaluate the veiling glare emitted from the instrument panel [13]. The advancement of computer graphics and lighting algorithms enables to accurately render the light reflection. This assessment is critical for night-time conditions as the reflected light can compromise the driver's sight.

Other scenarios pushed the technology further by assessing the ergonomics and reachability of the design. These scenarios assessed the impact of physical tasks on the human operator to ensure they can perform the assignment safely and efficiently. Other cases focused on, e.g., the accessibility of the filters, the door handles on larger vehicles, or the configuration of the instrument panel. Aerospace engineers frequently used VR as well to visualize real-scale virtual prototypes and study the ergonomics of interior structures such as the cockpit, deck, workstation, and passenger seats. Airbus has been using XR as an integral part of their development and design process (e.g., the reachability of the oxygen masks, AR quality checks, or ensuring the security features and available space complies with the authorities' regulation) [2]. Airbus stated that XR reduced the inspection time of 60'000 brackets from three weeks to three days.

Current graphics hardware can render a near realistic visualization of the materials and lighting of a product, enabling designers to evaluate the aesthetics characteristics of the design. The manufacturer Ford owns a dedicated laboratory in Michigan which utilizes VR to design new vehicle concepts. This technology helps them understand the aesthetic qualities

of the designs [5, 9]. In their cases, the model of the car is built to include every nut and bolt; therefore, it is possible to evaluate every component in the car whether it is aesthetic or functional. Any changes to the design are reflected in real-time, allowing them to see the subtle variations, thereby accelerating the prototyping process and reducing the amount of physical clay models built. Their simulation can host more than one person, allowing another expert, even across the globe, to explore the design and give insightful feedback. Ford stated that VR has sped up the production process, cut down costs and improved craftsmanship [9].

In some cases, the scenarios use visualizations that have no physical representation in real life. It is mostly used for abstract data visualization. In [6], the National Renewable Energy Laboratory developed a VR application to observe the wind wakes generated by turbines. This information has been used to improve the design of the turbines and their placements.

4 Discussion

In recent years, Virtual and Augmented Reality technologies have improved significantly and have become widely accessible at an affordable cost. The reviewed literature on both technologies suggests that these technologies can become a valuable tool for professionals in the industrial sector. The survey focused on the architecture, engineering, construction, and manufacturing industry, listing the different practices and research conducted.

In all cases, XR technology can positively impact the workflow of a company. Viewing and interacting with a 3D model at world-scale allows the user to better understand the product, its interactions and relations with the world, leading ultimately to an improved version of the final product and a reduction of costs [10, 1, 5]. With the computing and graphical performances currently available, this functionality can also serve as a tool to evaluate more profound attributes of a design, such as the ergonomics, the aesthetics, and the usability [5, 6, 13]. Additionally, the focus on visualization with these technologies seems to positively affect the sense of engagement of the participants, which can be noticed during group meetings or lessons [5, 20]. The latter is particularly useful, as it assists the students in understanding key concepts. Furthermore, virtual simulations can provide a sandbox for them to develop critical skills that

would otherwise require on-site practice, which entails risk exposure as well as additional costs for material usage and damage to equipment.

From the literature, it can be inferred that XR technology is stable and can be used for professional operations. Moreover, its integration can provide meaningful way to support the workflow. It is worth noting that the literature reviewed was based on the technology available at the time, yet the technology is gradually improving. With the recent release of the eye-resolution headset and the recent announcement of the Microsoft Hololens 2, VR and AR are likely to attract more businesses.

5 Conclusion

This paper has investigated the applications of virtual and augmented reality technology in the industrial sector. The reviewed literature suggests that their ability to visualize and interact with models at world-scale can be a valuable tool for designers, engineers, and students. Additionally, the interactive environment has been shown to provide a more active discussion within the team, increasing their engagement. This leads to the reduction of defects, time-loss, and costs, while still improving the craftsmanship of the final product. Although these technologies seem to have sufficiently matured, they should continue to improve and overcome some of their shortcomings, leading to enhanced immersive experience and contributing to greater use of XR technology.

Future research could expand in more detail regarding the other industries mentioned in Sec. 2.3. The human-eye resolution headset (VR-1) by Varjo is not targeted to the average consumer, it is intended for design, training and simulation, and AEC. This leads to some open research regarding how the resolution of a headset affects the overall experiences by comparing the VR-1 with the other latest HMDs on the market. Effective training simulations are hard to achieve due to the time it requires to model a scene that is realistic enough to depict real working conditions; therefore, another research could explore the effectiveness of a 3D laser scanned model of an actual construction site inside a VR simulation. From the literature and further research, it was noticeable that there is a lack of standardization due to the competitiveness of the market; thereby, each device has its own framework. A more laborious research would be the development of a unique framework that can be used throughout every device and provide the same type of features for interactions.

References

- [1] Siddhant Agarwal. Review on Application of Augmented Reality in Civil Engineering. pages 1–4, 2016.
- [2] Airbus. Virtual reality with real benefits, September 2017.
- [3] C. Anthes, R. J. García-Hernández, M. Wiedemann, and D. Kranzlmüller. State of the art of virtual reality technology. In *2016 IEEE Aerospace Conference*, pages 1–19, March 2016.
- [4] Ajang Behzadi. Using Augmented and Virtual Reality Technology in the Construction Industry. *American Journal of Engineering Research*, pages 1–4, 2016.
- [5] Leif P. Berg and Judy M. Vance. An Industry Case Study: Investigating Early Design Decision Making in Virtual Reality. *Journal of Computing and Information Science in Engineering*, 17(1):011001–1–011001–7, November 2016.
- [6] Leif P. Berg and Judy M. Vance. Industry use of virtual reality in product design and manufacturing: a survey. *Virtual Reality*, 21(1):1–17, March 2017.
- [7] Nathan Blinn, Michael Robey, Hamzah Shanbari, and Raja R A Issa. Using Augmented Reality to Enhance Construction Management Educational Experiences. pages 1–10, 2015.
- [8] Thomas P. Caudell and David W. Mizell. Augmented reality: an application of heads-up display technology to manual manufacturing processes. *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, July 1992.
- [9] Phil Covington. Inside Ford’s Virtual Reality Labs, January 2017.
- [10] Jordan Davidson, John Fowler, Charalampos Pantazis, Massimo Sannino, Jordan Walker, and Farzad Pour Rahimian. Virtual Reality Applications in Architecture: Bill of Quantities & Virtual Reality. pages 1–6, 2018.
- [11] Jack Goulding, Wafaa Nadim, Panagiotis Petridis, and Mustafa Alshawi. A VIRTUAL REALITY INTERACTIVE TRAINING ENVIRONMENT FOR THE CONSTRUCTION INDUSTRY. pages 1–17.
- [12] Rebecca Hills-Duty. National Museum of Finland Offers Virtual Time Travel, February 2018.
- [13] J. Kovar, K. Muralova, F. Ksica, J. Kroupa, O. Andrs, and Z. Hadas. Virtual reality in context of Industry 4.0 proposed projects at Brno University of Technology. In *2016 17th International Conference on Mechatronics - Mechatronika (ME)*, pages 1–7, December 2016.
- [14] G. Senthil Kumaran, K.R. Santhi, and P.M.Rubesh Anand. Impact of Augmented Reality (AR) in Civil Engineering. *Advanced Materials Research*, 18-19:63–68, June 2007.
- [15] Sebastjan Meža, Žiga Turk, and Matevž Dolenc. Measuring the potential of augmented reality in civil engineering. *Advances in Engineering Software*, 90:1–10, December 2015.

- [16] Dimitris Mourtzis, Vasilios Zogopoulos, and Ekaterini Vlachou. Augmented Reality supported Product Design towards Industry 4.0: a Teaching Factory paradigm. *Procedia Manufacturing*, 23:207–212, 2018.
- [17] Sara Rankohi and Lloyd Waugh. Review and analysis of augmented reality literature for construction industry. *Visualization in Engineering*, 1(1):1–18, 2013.
- [18] W. G. R. M. P. S. Rathnayake. Usage of Mixed Reality for Military Simulations. In *2018 International Conference on Current Trends towards Converging Technologies (ICCTCT)*, pages 1–5, March 2018.
- [19] Sales. Know the Augmented Reality Technology: How does AR Work?
- [20] Peng Wang, Peng Wu, Jun Wang, Hung-Lin Chi, and Xiangyu Wang. A Critical Review of the Use of Virtual Reality in Construction Engineering Education and Training. *International Journal of Environmental Research and Public Health*, 15(6):1204, June 2018.
- [21] B. K. Wiederhold, I. T. Miller, and M. D. Wiederhold. Using Virtual Reality to Mobilize Health Care: Mobile Virtual Reality Technology for Attenuation of Anxiety and Pain. *IEEE Consumer Electronics Magazine*, 7(1):106–109, January 2018.
- [22] Stefanie Zollmann, Christof Hoppe, Stefan Kluckner, Christian Poglitsch, Horst Bischof, and Gerhard Reitmayr. Augmented Reality for Construction Site Monitoring and Documentation. *Proceedings of the IEEE*, 102(2):137–154, February 2014.

Fraudulent user identification methods on social media

Akzharkyn Duisembiyeva

akzharkyn.duisembiyeva@aalto.fi

Tutor: Mika Juuti

Abstract

As the social media becomes an ubiquitous part of society, fraudulent user accounts also appear. The definition of fraudulent users varies based on social media, and different criteria were selected for proposed methods and algorithms. One of main aims motivating the appearance of fraudulent users is to boost of the number of followers and artificially boosting the apparent social status of user. The purpose of creating fraudulent user profiles include adding a fake review for the places and spreading the fabricated news. Distinguishing between counterfeit and real users, news or articles in the Internet becomes a problem, and various research have been conducted to resolve this issue. This paper focuses on surveyed of completed and ongoing research in this field such as main solutions and their limitations.

KEYWORDS: *fraudulent users, social media, graph-based fake user identification methods, social status, fake followers*

1 Introduction

Investigating the fraudulent profiles in social media is an interesting topic in Machine learning. According to Kumar et al. [11] these fraudulent accounts are difficult to be distinguished from accounts of real users, and are designed to spread the posts with false information. Therefore, understanding of the threat of having a counterfeit users is necessary.

Finding fake users is challenging topic due to high percentage of fraudulent users in social media and incapability of human to distinguish the real user from counterfeit, as it is demonstrated through various research works from the fields of psychology, sociology and data mining. The methods to mislead people and tools for finding fake users use the understanding of human cognition and reaction to information.

The algorithms for finding fraudulent accounts in social media have not appeared recently, since their period of creation matches the dates when the popular social media platforms were developed and became widely spread. One of the first methods was developed by Noble et al. [13] in 2003, by Sun et al. [22] in 2005, by Eberle et al. [5] in 2007 and by Papadimitriou et al. in 2010 [15]. All methods used graph-based approach. This means that the topic of fraudulent users was a problem already since the beginning of 21st century. The later algorithms reviewed and utilized the central ideas behind these methods.

The paper focuses on the common issues caused by having fraudulent users in social media described in section 2 and reviewing three graph-based methods: *Fraudar*, covered in section 4.2, *CopyCatch* in section 4.1 and *LPAD* in section 4.3. The notions and technical terms are given in section 4 for providing with a basic technical background.

2 Background

The paper focuses on the ways of determining the fraudulence of user profile in social media, investigating the tools and techniques that could be utilized to achieve this, finding the advantages and disadvantages of existing tools, discussing about the main recent achievements and getting familiar with current limitations on this topic.

Speaking about the scope of the problem, the impact and potential danger of fraudulent user identification are significant. These impact and potential danger include usage of fake accounts for political influence, af-

fecting the social status of real user and possibility of fake users uploading a harmful content.

Usage of bot account in social media during political events. Bessi et al. [1] and Shao et al. [21] measured the impact of bot users during the main political events as election.

Harm of counterfeit followers in social media for business. The social status of users compounds of having the large number of followers, "likes", comments and realistic pictures. Generally, the fraudulent account can have large number of followers and seems as an account of successful influencer. Knowing that social media advertisement targets users, having a substantial number of fake followers leads to wrong statistics and marketing, which undermines the advertiser's business model. According to Forbes [23], this harm applies for the campaign to advertise a brand or business that is accomplished by fake user account with a large number of followers, comments and "likes". The business loses both targeted audience, money and advertisement opportunity for product.

Handling dishonest business and commercializing of having a large number of social media users. The problem of fake users boosting has an importance, since business promotion is handled in dishonest ways as buying bots that are imitating the existing users as followers. According to Crys Wiltshire "Fake followers also greatly impact the metrics that are reported on when brand see such low engagement from supposedly huge influencers" [24]. The price for purchasing bot accounts as followers is given by Scurrel et al. [8]. According to Scurrel et al. [8], the number of followers can be increased by purchasing them from various websites such as buzzoid.com, with prices normally varying between 5 and 15 dollar per 1000 followers.

The possibility of fake users uploading a harmful content. The harm of fake users in social media is connected with the fact that users are allowed to upload the content (e.g., posts and pictures) as desired. Therefore, the content of pictures might be a sabotage, e.g., nudity, capture of terrorist events, immorality or providing the pictures of other people without consent, which breaks the personal privacy of people.

In conclusion, the impact of fake information produced by fake users can be devastating, which affects the various areas of society. This paper focuses on fake users on various social media platforms due to influence of these social media platform in mass media and general life style, geographical expansion and having users of the wide age range.

3 Characteristics of fake users

The fraudulent users share the different characteristics in various social media. These characteristics establish the criteria for being considered as fraudulent or not by the researchers.

According to Beutel et al. [3], the clues that can show if the user is fake or not are: being disabled, describing the purpose of account in username instead of composing the username from firstname, lastname or nickname, lack of content in feed while having a large number of followers, and sharing the malicious or commercial information with followers in feed or providing links to prohibited resources.

According to Hooi et al. [4], the fraudulence of users is based on the following characteristics of their profile data: links on profile associated with malware or scams, clear bot-like behavior (e.g., replying to large numbers of tweets with identical messages), deleted or suspended status of account.

According to Kagan et al. [20], the characteristics of fake users many of these profiles are inactive status, generated commercial content of the user's feed, whereas other profiles largely share and follow the content of feed from other users.

4 Techniques

Overall, researchers developed various types of algorithms for detecting fraudulent users. In general, these methods can be combined into several categories. This paper focuses on graph-based algorithms.

The graph-based algorithms can be divided into several categories, and the most common are: edge distributions [20], dense block detection [4], co-clustering [3] and link prediction [6]. The principles of algorithms work are described in the subsections 4.1, 4.2 and 4.3. The subsection 4.1 describes the co-clustering approach using data sets of Twitter and Weibo, subsection 4.2 describes the edge distribution and density-based method using data sets of Twitter, while subsection 4.3 combines edge distribution and link prediction algorithms using data sets of social media as Academia, ArXiv, Class, DBLP, Flixster, Twitter and Yelp.

The paper describes three various algorithms. Table 1 provides the brief overview of methods.

In order to calculate and represent the success rate of this algorithm, the

Table 1. Comparison of graph-based methods

Method	Type	Achievements	Drawbacks
CatchSync	Clustering	Prevents injected attacks, restores normal patterns	Cannot detect camouflage accounts
FRAUDAR	Edge distribution and density-based	Handles sybil attacks, camouflage	Cannot handle sparse graphs
LPAD	Edge distribution and link prediction	Handles unsuspected behaviour in fully labeled dataset	Difficulties with hijacked profiles

researchers used *F measure*, which is defined as the weighted harmonic mean of the precision and recall of the test [19].

$$F \text{ measure} = \frac{2 * \textit{precision} * \textit{recall}}{\textit{precision} + \textit{recall}},$$

where *precision* is the fraction of relevant instances among the retrieved instances, while *recall* is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. [17].

The precision refers to accuracy of the model out of those predicted positive, how many of them are actual positive. In fraudulent users detection, a false positive means that an user that is non-fraudulent (actual negative) has been identified as fake (predicted fraud). Therefore, businesses, as examples of parties interested in this topic, might incorrectly classify the real users, if the precision is not high for the fraudulent users detection model.

Speaking about the recall in fraud detection, if a fraudulent user (Actual Positive) is predicted as non-fraudulent (Predicted Negative), this will lead to negative consequence for the business.

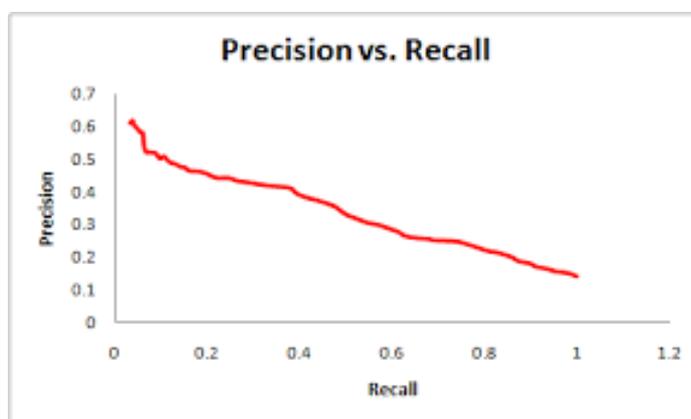
**Figure 1.** The precision- recall curve [9]

Figure 1 shows the tradeoff between the precision and recall. The high percentage of F measure does not imply that the fraud detection algorithm is a guarantee of perfect result for algorithm.

4.1 Clustering graph algorithm

Beutel from CMU conducted PhD research, among which the “*CatchSync*” algorithm was presented. This algorithm was used for social media as Twitter and Weibo, and was applied on two large real datasets: 1-billion-edge Twitter social graph and 3-billion-edge Tencent Weibo social graph. This algorithm outperforms the previous algorithms in the same field *Outrank* [12] and *Spot* [16], since *CatchSync* is capable of completing the labelling tasks and finding new patterns of unusual group behavior. [2].

The graph represented as adjacency matrix in the program. The clustering method on graph is an approach of clustering nodes based on who they connect to, while co-clustering jointly clusters both classes of nodes in a bipartite graph.

The nodes of graph represents “*followee-follower*” relationship, where the accounts are placed in the edges of graph. Beutel uses graph mining approach which uses exclusively the graph structure to find nodes that are suspicious due to their position in the graph.

According to Beutel, the previous content-based approaches provided by other researchers in the field of clustering algorithms are limited, since they focus on the information in the profile and feed, however these algorithms miss the number of purchased followers.

The algorithm exploits two of the tell-tale signs left in graphs by fraudsters. The conditions and measurements which are used to determine if the account is real or not are “synchronicity” and “rarity”. The algorithm works on the resulting synchronicity-rarity plots. Authors define *synchronized behavior* as the suspicious nodes have extremely similar behavior pattern, because they are frequently required to perform some task together. According to Beutel et al. [2], an example of synchronized behavior is following the same user. The concept of *rare behavior* stands for the connectivity patterns of nodes that are very different from the majority.

At the first step, a feature space for target nodes is derived. Then, at the second step, synchronicity and normality of behavioral patterns of the source nodes is computed, according to the relative positions of their target nodes in the feature space. Finally, distance-based outlier detection method is utilized to detect the point that is distant from other observations in the synchronicity-normality plot.

The main limitation of this algorithm is handling the *camouflage profiles*, which will be covered in section 4.2 due to the difficulty recognizing

these profiles. The algorithm presented in this paper outperforms other existing algorithms with detection accuracy by 36 % on Twitter and 20 % on Tencent Weibo, as well as in speed. The *F measure* is 0.8 on Twitter data set.

4.2 Edge distribution and density-based graph algorithm

Researchers from Carnegie Mellon University (CMU) suggested use of edge distributions. The features of fake users that are referred as “*fraudster*” used for this algorithm is *connection of fake users to other fake users*.

Their algorithm is called “*FRAUDAR*” [7]. Other algorithms in this fields were developed previously, such as “*Spoken*” [18] and “*NetProbe*” [14], however the suggested “*FRAUDAR*” algorithm is capable of handling “*camouflage*” accounts unlike the predecessor algorithms, therefore it outperformed the predecessor algorithms by finding undiscovered fraud types. The notion of *camouflage* account was used to denote the fraudsters that pretend to look as real users by adding links to popular items/idols (like famous singers/actors, or well-liked products).

The algorithm focused on potential attacks as well by considering that *fraudsters* are aware of all information given in the graph and fraud-detection algorithms. Therefore, the algorithm benefits and outperforms the predecessor algorithms from considering worst-case scenario. The *camouflage* attacks can be completed using 4 different methods. The first type is injection of fraud with no camouflage. The second is random selection of which edges are camouflaged which are equal to the number of fraudulent users in graph. The third method is placing the camouflaged edges directing towards fake user vertex with probability proportional to the degree of vertex. The last method is using a random subset of existing users to from the fraudulent block.

The algorithm procedure is divided into several steps. First of all, the suspicion factor for each node is computed. The probability of being a fraudulent account increases with this factor. The suspiciousness for each edge between suspicious nodes is computed also. The group of suspicious nodes connected with each other form a suspicious area. At the second step, the dense blocks for each suspicious areas are defined. Then, the concentration of blocks is computed by considering the axiom “A subset with smaller size is more suspicious than one with the same total suspiciousness but larger size” [7]. The last step in this algorithm is finding the final set of suspicious nodes and edges.

As an assumption for being a fraudulent user by authors, the behavior of fraudulent user is predictable in a way that the fake user gives a feedback to another fake user, and, if the algorithm finds one fraudulent user, then the neighboring fake user can be identified.

As the result, the fraudulent users can be identified by this proposed algorithm with F measure of 0.95 on Twitter dataset considering all attack scenarios with density degree of 0.04, which is the main limitation of this approach. Speaking about the issue of sparse graphs, the fraudulent account are more resistant for algorithm in non-dense graph, since the algorithm searches for densest block and supposes that adversarial block is significantly more dense than the densest normal blocks in the graph, so this block can be detected. Therefore, this algorithm is limited to finding the fraudulent users in non-dense graphs with density degree less than value of 0.04.

4.3 Edge distribution and link prediction graph-based algorithm

Kagan et al. evaluated anomaly detection algorithm on 10 complex networks of three types: fully simulated networks, real world networks with simulated anomalous vertices, and real world networks with labeled anomalous vertices (Academia, ArXiv, Class, DBLP, Flixster, Twitter and Yelp).

The algorithm denoted as *LPAD* relies on more computational efficiency of topological features that is taken for supervised machine learning method (Random Forest classifier). The previous work in this field was accomplished by Fire et al [6] in 2012. Fire et al. proposed *Stranger*, a method for detecting fake profiles in social networks based on anomalies in a social structure of fake user. Their algorithm relied on the classifier that was simulated as a fake profile with a behavior assumption that fake user randomly sends friendship requests to other users in the network. *Stranger*, the predecessor work in this field, had limitation due to relying on community features, which are, according to Kagan et al., have a disadvantage of being sensitive to incomplete data. In addition to this, edge-based features do not have such drawback [10].

The link prediction method is denoted as discovery and determining current and future links between nodes in network. According to Kagan et al. [10], the features used for determining if the user is fake or not are based on *behavioral patterns*. The algorithm works in a way, that, first of all, the link prediction classifier is constructed and the additional set of

features is created, and, secondly, the second set of features is utilized to construct a meta-classifier that identifies anomalous vertices in graph.

The algorithm works in a way that *the users are built in a graph as nodes*. The assumption for a node to be fraudulent in algorithm having many edges for vertex with low probabilities of existing (improbable edges). The whole method is divided into two parts and based on the unsupervised two-layered meta-classifier.

The first part is *constructing link classifier*. At this step, the link prediction classifier is built based on the extraction of 19 features. Existing edges denote the positive examples, whereas the negative instances are non-existing ones. The negative cases represent the real user in social media, while the positive cases exemplify the malicious users. In order to construct the link prediction classifier, the Random Forest Algorithm was used for training sets.

The second part is *detecting Anomalous Vertices*. After the classifier was built, the unsupervised anomaly detection algorithm was used. The algorithm works in a several steps.

First of all, the link prediction classifier is trained to calculate the probability that an edge does not exist in the graph. Second step predicts the probability of each edge not existing using the link prediction classifier. Thirdly, the average probability is computed for each vertex. The last step is inspecting the vertices that have the highest average probability.

The features for determining if the node is malicious or not are: the probability of a vertex v to be anomalous, which is equal to the average probability of its edges not existing; the standard deviation of a set of vertex v probability of not existing for edges; the number of vertex v of edges that were labeled as anomalous, where the threshold for considering as anomaly is set to 0.8 for v ; the percent of v edges that are labeled as anomalous; the standard deviation of the classification of v edges; the median of set of vertex v probability of not existing for edges; the number of edges that vertex v has.

One of the strong sides of algorithm is ability to detect "hijacked" profiles, which perform unexpected behavior as random connecting to other vertices in the network [10].

The main limitation of the algorithm is randomness in behavior of social media users in a network as Twitter. As an instance, some fake users can follow anyone without needed consent from another user or can have a specific goal and strategy. As the result, the algorithm was able to classify

10 % of vertices as anomalous in the test set, which consisted of 900 samples. The authors state that algorithm has precision of 35 %, while recall is kept at 100 %. This amounts to $F\ measure = 52\ %$ at Twitter dataset.

5 Conclusion

The identification of fraudulent users in social media was investigated by broad range of researchers. The harm of such fake users in social media is substantial. The successful classification of fraudulent users was measured with the help of calculating F measure. Despite having solid tools and methods from graph-based algorithms, fake users determination faces limitations that prevent the algorithms from finding accurate and reasonable result. However, research projects and papers held in this topic present promising advancements. As for now, the algorithms used Twitter dataset for training and testing. Therefore, the process of determining fraudulent users in social media can be used already in popular platforms.

References

- [1] Alessandro Bessi and Emilio Ferrara. Social bots distort the 2016 us presidential election online discussion. 2016.
- [2] Alex Beutel. *User behavior modeling with large-scale graph analysis*. PhD thesis, Ph. D. Thesis at Carnegie Mellon University, 2016.
- [3] Alex Beutel, Kenton Murray, Christos Faloutsos, and Alexander J Smola. Cobafi: collaborative bayesian filtering. In *Proceedings of the 23rd international conference on World wide web*, pages 97–108. ACM, 2014.
- [4] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. Copycatch: stopping group attacks by spotting lockstep behavior in social networks. In *Proceedings of the 22nd international conference on World Wide Web*, pages 119–130. ACM, 2013.
- [5] William Eberle and Lawrence Holder. Anomaly detection in data represented as graphs. *Intelligent Data Analysis*, 11(6):663–689, 2007.
- [6] Michael Fire, Gilad Katz, and Yuval Elovici. Strangers intrusion detection-detecting spammers and fake profiles in social networks based on topology anomalies. *Human Journal*, 1(1):26–39, 2012.
- [7] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. Fraudar: Bounding graph fraud in the face of camouflage. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 895–904. ACM, 2016.

- [8] Scurrell Jennifer Victoria and Grossenbacher Timo. Identifying a large number of fake followers on instagram. accessed: 05.02.2019. URL: <https://srfdata.github.io/2017-10-instagram-influencers/>.
- [9] Jelena Jovanovic. Classification. URL: <http://ai.fon.bg.ac.rs/wp-content/uploads/2015/04/Classification-basic-concepts-2015.pdf>.
- [10] Dima Kagan, Yuval Elovichi, and Michael Fire. Generic anomalous vertices detection utilizing a link prediction algorithm. *Social Network Analysis and Mining*, 8(1):27, 2018.
- [11] Shah N. Kumar, S. False information on web and social media: A survey. 2018. URL: <http://adsabs.harvard.edu/abs/2018arXiv180408559K>.
- [12] HDK Moonesinghe and Pang-Ning Tan. Outrank: a graph-based outlier detection framework using random walk. *International Journal on Artificial Intelligence Tools*, 17(01):19–36, 2008.
- [13] Caleb C Noble and Diane J Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM, 2003.
- [14] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. Netprobe: a fast and scalable system for fraud detection in online auction networks. In *Proceedings of the 16th international conference on World Wide Web*, pages 201–210. ACM, 2007.
- [15] Panagiotis Papadimitriou, Ali Dasdan, and Hector Garcia-Molina. Web graph similarity for anomaly detection. *Journal of Internet Services and Applications*, 1(1):19–30, 2010.
- [16] Charles Perez, Marc Lemercier, Babiga Birregah, and Alain Corpel. Spot 1.0: Scoring suspicious profiles on twitter. In *2011 International Conference on Advances in Social Networks Analysis and Mining*, pages 377–381. IEEE, 2011.
- [17] David Martin Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.
- [18] B Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Christos Faloutsos. Eigenspokes: Surprising patterns and scalable community chipping in large graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 435–448. Springer, 2010.
- [19] Yutaka Sasaki et al. The truth of the f-measure. *Teach Tutor mater*, 1(5):1–5, 2007.
- [20] Neil Shah, Alex Beutel, Bryan Hooi, Leman Akoglu, Stephan Gunnemann, Disha Makhija, Mohit Kumar, and Christos Faloutsos. Edgecentric: Anomaly detection in edge-attributed networks. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 327–334. IEEE, 2016.
- [21] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Alessandro Flammini, and Filippo Menczer. The spread of fake news by social bots. *arXiv preprint arXiv:1707.07592*, pages 96–104, 2017.

- [22] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE, 2005.
- [23] Tom Ward. Don't get catfished by fake instagram accounts. accessed: 05.02.2019. URL: <https://www.forbes.com/sites/tomward/2017/08/15/dont-get-catfished-by-fake-instagram-accounts/#156fefe3297c>.
- [24] Crys Wiltshire. The fake followers epidemic. accessed: 05.02.2019. URL: <https://www.gshiftlabs.com/social-media-blog/the-fake-followers-epidemic/>.

CoAP Congestion control and DoS attacks

Luong Khiem Tran

luong.tran@aalto.fi

Tutor: Mohit Sethi

1 Introduction

The expression Internet of Things (IoT) was first introduced in 1999 [2]. At that time, Things were Radio frequency identification (RFID) tags which are simply used for automatically identifying and tracking objects attached with the tags. In recent years, there have been increasing numbers of developed IoT solutions. For examples, users can remotely control and monitor fridges, lights and doors with smart home applications or smart energy and building automation system have been deployed in different buildings for effectively consuming power or providing adaptive ventilation. According to [3], by the end of 2020, at least 25 billion objects or devices are expected to be connected to the Internet. The communication for those devices can be developed based on different protocols depending on different requirements. The Constrained Application Protocol (CoAP) has been designed and standardized by the Internet Engineering Task Force (IETF) to provide a simplified Hypertext Transfer Protocol (HTTP) interaction between IoT devices.

CoAP is located at the application layer and developed based on the Representational State Transfer (REST) architecture. Unlike HTTP based on Transmission Control Protocol (TCP), CoAP messages are transferred based on the unreliable datagram-oriented transport layer such as the User Datagram Protocol (UDP). However, UDP has been used in differ-

Application	CoAP Requests/Responses - CoAP Messaging layer
Session	DTLS
Transport	UDP
Network	IPv6 - 6LoWPAN Adaptation layer
Data Link	IEEE 802.15.4 MAC
Physical	IEEE 802.15.4 PHY

Figure 1. CoAP Protocol Stack

ent Denial-of-Service (DoS) attack mechanisms because UDP does not require return routability check. For instance, according to [6], Network Time Protocol (NTP) Servers running on UDP have been used for Distributed Denial-of-Service (DDoS) attacks. The magnitudes of responses from those NTP servers were 19 times larger than the original request. Besides, because CoAP relies on the UDP transport protocol, it also faces another challenge in congestion control to ensure the reliability of packet sending. There are some mechanisms for congestion control, such as basic congestion control of CoAP [16] (Default CoAP), CoAP simple congestion control/advanced (CoCoa) [4] and recently FASOR [11] which are analyzed in this article.

This article analyzes two challenges of using CoAP for IoT solutions including vulnerabilities to DoS attacks and congestion control solutions for CoAP packets.

The next section, Section 2, introduces fundamental characteristics of CoAP, DoS and congestion control problem over networks. Section 3 presents and analyzes different CoAP vulnerabilities to DDoS attacks. Section 4 presents and analyzes various congestion control challenges and solutions over UDP that can be applied to CoAP deployment. Finally, Section 5 concludes the works of this article.

Feature	Detail
Web protocol	REST architecture - Request response interaction model - 4 REST methods GET, POST, PUT and DELETE - URI and content-types Support HTTP mapping for integrating to the existing WEB
UDP binding	Support unicast requests with optional reliability Support multicast requests
Machine-to-machine requirements	Built-in discovery of services and resources Multicast support Very low overhead and simplicity Asynchronous message exchanges
Constrained environment	Resource-constrained devices - 8-bit microcontrollers - Limited RAM and ROM Constrained network - 127 bytes frame - Throughput \leq 250 kbit/s

Figure 2. CoAP features [16], [13]

2 Background

2.1 Constrained Application Protocol CoAP

This section provides overall main features of CoAP which are utilized for the analysis in next sections. Figure 1 illustrates the protocol stack where CoAP is based on. Figure 2 describes details of four features of CoAP including supporting Web protocol, UDP binding, supporting Machine-to-machine requirements and applied on constrained environment.

CoAP is a protocol at the application layer of IoT applications. CoAP is designed to run with resource-constrained devices, on constrained networks, and be able to integrate with the existing Web through HTTP translation [16]. The resource-constrained devices often have 8-bit microcontrollers with small storage capacity of ROM and RAM. The current standard for constrained networks is Internet Protocol version 6 (IPv6) over Low-Power Wireless Personal Area Networks (6LoWPAN). The 6LoWPAN is the design specification for network environments with high packet error rates, using 127 bytes data frame and low throughput [13]. The REST model of HTTP is applied for one-to-one communication between CoAP devices with four supported methods GET, POST, PUT and DELETE.

However, unlike HTTP, CoAP supports IoT specialized requirements focusing on machine-to-machine (M2M) applications. The requirements include built-in discovery of services and resources discovery, multicast, very low overhead and simplicity for constrained environments. M2M applications that operate autonomously with less manual human efforts depend upon multicast requests to explore the services and resources directory. CoAP endpoints that offer multicast service discovery need to join

the group of all-CoAP-node multicast addresses including 224.0.1.187 for Internet Protocol version 4 (IPv4) and FF0x::FD for IPv6. The multicast feature of CoAP can be only enabled by utilizing UDP at the transport layer which supports the use of multicast Internet Protocol (IP) destination addresses.

CoAP transmissions are secured with Datagram Transport Layer Security (DTLS) protocol at the session layer similar to Transport Layer Security (TLS) for HTTPS. There are four security modes of for a CoAP device:

- **NoSec:** DTLS is disabled. This mode can be used for unimportant transmissions or used with Internet Protocol Security (IPsec). The NoSec mode does not require encryption processes, thus less energy consumption for constrained devices. However, this mode should be utilized deliberately or there are many security risks exposed, especially the risk of DoS attacks which is discussed in section 3.
- **PreSharedKey:** DTLS is enabled with a list of pre-shared keys. Each key is attached with a list of nodes that the device can use the key to communicate.
- **RawPublicKey:** DTLS is enabled with one asymmetric key pair without a certificate for the device and a list of public keys of nodes that the device can communicate with.
- **Certificate:** Similar to RawPublicKey but each device also has an X.509 certificate for the key pair and an additional list of root trust anchors for validating certificates of public keys of interacted nodes.

UDP which is not designed for reliable transmissions [1]. Therefore, CoAP utilizes different message types to control reliability levels at the message layer in the figure 1. The confirmable message type is used for reliable transmissions that require retransmitting the messages multiple times until receiving the corresponding Acknowledgement (ACK) packet. The retransmissions introduce again the traffic congestion that has been already challenged the TCP protocol from the very beginning. With the large number of IoT devices transferring vast amounts of packets over networks, the congestion control has been already considered from the CoAP standard. The congestion control mechanisms are discussed more details in the section 4.

2.2 Distributed Denial of Service Attacks

DoS is an attack method that attempts to prevent legitimate users from using a service [18]. Distributed DoS or DDoS attack is the large number of DoS attacks launched from multiple compromised hosts that are remotely controlled by the attacker. There are multiple targets of a system of for a DoS attack, such as physical devices of the system, computational resources of devices or the communication of the local network of the system [10].

Regarding to the selected attacked target, different approaches can be applied for launching a DoS attack. One approach is interrupting the functionalities of internal system nodes by physical device destruction, forcing devices running out of energy or manipulating the original functions of devices. Another approach is flooding a vast amount of packets to overload either processing capacity of the target system or the traffic of the local network utilized by the target system. According to [18], one flooding type is the direct DDoS attack with the packets are all sent from compromised nodes to the victim. One other flooding type is the reflector DDoS attack which adds another layer of machines called reflectors. The compromised nodes send requests with spoofed source address, that is the victim address, to the reflectors. The reflectors respond each request to the victim with some responses or with one response having larger size than the request. The larger size of responses comparing to the request size, the more critical of the attack. The ratio between the size of responses and the request size is the amplification factor of the DDoS attack. These discussed attack approaches are realized on CoAP systems in the section 3.

In this article, the DoS attack over CoAP systems is analyzed in two aspects. The first aspect is the DoS attack direct to the CoAP distributed system itself. The second aspect is the compromised CoAP devices are possibly utilized for DDoS attacks to other systems.

2.3 Network Congestion Control

The communications between hosts are based on packet transmissions from sources, through different interconnected nodes before coming to the destination [17]. The middle nodes process forwarding data to following nodes with the difference between input and output transmission rates. Queuing and dropping packets are mechanisms for controlling data trans-

mission that is over capacity of a middle node. Congestion happens when there are packet loss or timeout due to queuing delay that lead to retransmit packets from sources.

As specified in the UDP usage guidelines [7], congestion control is the most essential part of maintaining the stability of the Internet. Applications or protocols that operate over UDP transport layer must implement mechanisms to restrain congestion collapse and to ensure some extents of fairness with other coexisting traffic.

The UDP usage guidelines provide specific directions for different application traffic types including applications performing bulk transfers, low data-volume applications or applications supporting bidirectional communications.

Applications that transfer more than a small number of UDP datagrams per RTT are considered bulk transfer applications. These applications should employ TCP-Friendly Rate Control (TRFC), window-based, TCP like congestion control, or at least comply with the congestion principles [9].

Low data-volume applications can be specified by at any time transferring only a small number of UDP datagrams with a destination. Those applications should still control the transfer rate on average about one UDP datagram per RTT. For packet loss retransmission, three considered components include RTT estimation for each destination, packet loss detection and exponentially back-off retransmission timeout. The IETF standard [15] based on Karn's algorithm [12] is a comprehensive recommendation for the RTT estimation mechanism. Applications that cannot maintain a reliable RTT estimate can apply different following approaches:

- Utilizing predefined retransmission timeout which is exponentially backed-off on having packet lost. Default RTO of CoAP is 2 second.
- For the exchanges without return traffic, applications should limit sending rate to one UDP datagram per 3 seconds or less.

Applications that bidirectionally exchange packets should implement congestion control from both sides. Client-server model with request-response-style is a type for these applications. To control congestion, both client and server should independently detect and respond to congestion of the transmissions, utilize acknowledged responses across the entire round-trip path for limiting retransmitted or new requests.

The above congestion control guidelines have been applied in designing

various congestion control solutions to support CoAP applications. The section 4 discusses these solutions in more details.

3 CoAP Vulnerabilities to DDoS Attacks

3.1 IP Address spoofing on UDP and Risk of Amplification of CoAP packets

According to the security consideration of [16], CoAP devices utilizing NoSec or PreShareKey security mode probably expose risks to IP address spoofing attacks. A forged endpoint is able to read and write messages transferred in the constrained network. The PreShareKey mode with a nodes/key ratio $> 1:1$ reduces the attacker effort to obtain keys by interrogating the memory of devices and stealing cryptographic keys. Attackers are now able to launch different attacks including:

- Spoofing Reset messages of an endpoint to respond Confirmable messages of Non-confirmable messages, thus isolating the endpoint from other nodes.
- Spoofing ACKs to Confirmable messages to interrupt reliable transmissions because the sender will never retransmit lost messages, thus the actual responses are probably omitted at the sender.
- Spoofing responses with malicious payload to attack more severely and impact more endpoints in the network. For examples, poisoning proxy caches or disrupting the resource directories.
- Spoofing a multicast request for a target endpoint to generate a flood DoS attack with high number of useless responses to the victim.

The research [8] has indicated that attackers may manipulate reflectors running CoAP protocol to flood UDP packets to the victim with less obstruction from firewalls. Attackers in this case can spoof the IP address of the victim before sending requests to reflectors. The reflectors running CoAP applications trust the IP address of the victim then responding all bogus requests to the victim. Moreover, statistics from this report [5] demonstrate that there are about 16 percent of the IPv4 and IPv6 blocks and over 20 percent of the IPv4 and IPv6 autonomous systems are still susceptible to spoofing.

In a different circumstance, the research [8] has identified the amplification risk of request/response protocols running over UDP and estimated

the bandwidth amplification factor (BAF) for CoAP usage. The root cause of this amplification risk is the CoAP responses can be significantly larger than the requests. In the case that an attacker is able to POST or PUT a very large content to a reflector, then generating multiple spoofed requests to that content. Although the block-wise transfer feature of CoAP can be applied to divide a large response into smaller ones, the attacker can request configuration to the maximum block size. The amplification factor increases to 32x times compare to the request size. Recorded amplification attacks utilized Domain Name System (DNS) and Simple Service Discovery Protocol (SSDP) with the BAF ranges between 28x and 98x times of 1Mbps bandwidth of sending requests.

Another presentation about DDoS attacks based IoT [14] has mentioned that there is a significant increment of 26000 CoAP nodes in 2017 to 220000+ CoAP nodes in 2018. Besides, from the presentation, the amplification factor for DDoS attacks using CoAP is from 16 to 97 using the /.well-known/core of resource discovery link in each CoAP host. The /.well-known/core is the entry point of each CoAP host that allows other nodes explore supported resources of the current host without any human intervention. However, utilizing this feature requires more consideration on security aspect to limit the DDoS attack risks.

3.2 CoAP extensions and DoS risks

The CoAP standard prepare a base for various extensions to support different requirements of M2M applications. Those extensions include: itemize

- The observe mechanism to auto-update representation states from servers without sending GET requests;
- The resource directory mechanism to allow one entity can server as a central point for discovery resources of all nodes in a subnet;
- Or the publisher subscriber mechanism which is a more complex observe architecture with one broker that is always up to serve data of different topics to different subscribers that already subscribed to the corresponding publishing sensors. Those sensors can be routinely in sleeping mode to save energy.

However, these extensions also introduce risks to DoS attacks for the CoAP systems. For example, a Resource Directory (RD) which responds to /.well-known/core potentially larger amplification factor than the simi-

lar request to a normal CoAP node. Because, most of resource links of all around nodes have been registered to the Resource Directory entity. Those resource links will be returned for the same entry point request from a malicious endpoint. The other risk is from the publisher subscriber mechanism where a malicious client can publish a very large amount of data to overload the processing capacity of the broker, thus halting the normal behavior of broker. If the broker is down, all subscribers and publishers will be denied of their normal operations.

4 Congestion Control Solutions for CoAP

4.1 Default CoAP congestion control

The Default CoAP has been developed in a very basic way but still following the congestion principles mentioned in the section 2.3. There are some essential points of deploying the Default CoAP: itemize

- The round-trip time (RTT) is static configured and is used for calculating when a sender should totally stop expecting any ACKs of a sending message and its retransmitting ones.
- The initial retransmission timeout is randomly selected from a static configure range which is by default from 2 to $2*1.5$ second.
- The number of retransmissions for one message after the first retransmission is limit and static configured to 4 times. The intervals between these retransmissions are exponentially increase with the base is 2.
- The number of parallel sending messages to one destination is also static configured with the parameter NSTART. The default value is 1. However, increasing this value requires careful estimation of the RTT based application characteristics and network statistics, or an adaptive RTT estimation and a deliberate RTO calculation applied.

The advantage of this solution is that it is simple to be implemented in constrained devices. However, this solution cannot adapt with network traffic changing, especially when the CoAP nodes integrate with Internet which is never stable. On the other hand, the design of the Default CoAP allows to extend with more advanced solutions by modifying base parameters or providing mechanisms to dynamically update the parameters.

4.2 CoAP simple congestion control/advanced - CoCoA

The draft of CoCoA [4] provides a thorough congestion control solution based on parameters introduced by the Default CoAP. CoCoA is supported with different experiments on constrained devices on various network technologies. CoCoA control congestion in a more adaptive way compare to the Default CoAP. Main features of CoCoA include: itemize

- Dynamically change RTT with a Strong RTT estimator and a Weak RTT estimator. The Strong one updates RTT with the time of getting the ACK of a CON message without any retransmission. The Weak one utilizes the time of getting ACK from the time of sending the message till the time of receiving the ACK of the first retransmission. The Weak one probably has ambiguous estimation for RTT because CoAP cannot distinguish ACK of the orinal message or the first retransmission.
- The initial retransmission timeout (RTO) is calculated based on the previous RTO and a new RTO derived from the updated RTT.
- Apply variable backoff factor for calculating the interval to the next retransmissions after the first one. This feature allows CoCoA to adjust the sending rate to be near the rate of the default CoAP after having events impacting the RTO.

4.3 FASOR and other congestion control solutions for CoAP

There has been not yet a generic robust congestion control solution for CoAP applications in different networks. The constrained devices, constrained networks and the integration to the existing Web abilities require various experiments with different settings to to evaluate the robustness of the solution. The author of ORS [?] has indicated the weakness of CoCoA with some experiments. CoCoA does not adapt with high traffic networks where the chance of receiving ACKs is very low then the RTT is not updated with the enviroment. The sender will continue sending waste messages to the congestion points and make the congestion worse. This event is named congestion collapse. FASOR has proposed a solution for this problem based on the way that TCP has overcome congestion collapse.

There are other extensions of CoCoA in the literature to construct a more robust solution for CoAP systems.

5 Conclusion

This article has constructed an overview of the risks of DoS attacks and the work in the literature of congestion control solutions for CoAP systems. The constrained resources and environments and the integration to Internet ability require more researches and experiments to have more best practices and guidelines to ensure the security and quality of the CoAP applications.

References

- [1] User Datagram Protocol. RFC 768, August 1980.
- [2] Kevin Ashton. That 'Internet of Things' Thing, 2009.
- [3] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells. CoAP congestion control for the Internet of things. *IEEE Communications Magazine*, 54(7):154–160, 2016.
- [4] Carsten Bormann, August Betzler, Carles Gomez, and Ilker Demirkol. CoAP Simple Congestion Control/Advanced. Internet-Draft draft-ietf-core-cocoa-03, Internet Engineering Task Force, February 2018. Work in Progress.
- [5] Center for Applied Internet Data Analysis (CAIDA). State of ip spoofing, 2019. (Accessed on 02/28/2019).
- [6] Cisco. Cisco Security Notice: CVE-2013-5211, 2014.
- [7] Lars Eggert, Gorry Fairhurst, and Greg Shepherd. UDP Usage Guidelines. RFC 8085, March 2017.
- [8] Rainer Vosseler Federico Maggi and Davide Quarta. The Fragility of Industrial IoT's Data BackBone Security and Privacy Issues in MQTT and CoAP Protocols, 2018.
- [9] Sally Floyd. Congestion Control Principles. RFC 2914, September 2000.
- [10] IAB, Mark J. Handley, and Eric Rescorla. Internet Denial-of-Service Considerations. RFC 4732, December 2006.
- [11] Ilpo Järvinen, Iivo Raitahila, Zhen Cao, and Markku Kojo. FASOR Retransmission Timeout and Congestion Control Mechanism for CoAP. 20.
- [12] Phil Karn and Craig Partridge. Improving round-trip time estimates in reliable transport protocols. In *ACM SIGCOMM Computer Communication Review*, volume 17, pages 2–7. ACM, 1987.
- [13] Gabriel Montenegro, Jonathan Hui, David Culler, and Nandakishore Kushalnagar. Transmission of IPv6 Packets over IEEE 802.15.4 Networks. RFC 4944, September 2007.
- [14] Denis Rand. So you think iot ddos botnets are dangerous bypassing isp and enterprise anti-ddos with 90's technology, May 2018.

- [15] Matt Sargent, Jerry Chu, Dr. Vern Paxson, and Mark Allman. Computing TCP's Retransmission Timer. RFC 6298, June 2011.
- [16] Zach Shelby, Klaus Hartke, and Carsten Bormann. The Constrained Application Protocol (CoAP). RFC 7252, June 2014.
- [17] William Stallings. *Data and Computer Communications*. Prentice Hall Press, Upper Saddle River, NJ, USA, 10th edition, 2013.
- [18] William Stallings. *Network security essentials: applications and standards*. Pearson, 2016.

A Review of 2-Dimensional DNA Origami

Shiting Long

shiting.long@aalto.fi

Tutor:

Vinay Gautam

Pekka Orponen

Abstract

DNA nanotechnology as a notion to create scalable geometric DNA nanostructures using DNA as building blocks and DNA self-assembly as a guiding mechanism was ground-breaking [1]. This notion has inspired nanoscale system design and the nanoengineering of artificial systems ever since it was addressed. More attention was drawn to this field after the invention of 'DNA origami' [2], providing that a more inexpensive and profitable method was offered to practically manipulate the process of DNA self-assembly. With the development of technology, scientists have been able to assemble 2-dimensional(2D) DNA nanostructures from elementary geometric shapes like triangles to arbitrary user-defined images with immense complexity and novelty. This review paper aims at heralding the recent studies of 2D DNA origami while recounting the history of DNA nanotechnology.

KEYWORDS: *DNA nanotechnology, DNA origami, DNA tiling, DNA self-assembly*

1 Introduction

DNA nanotechnology was conceptually introduced by Nadrian Seeman in 1982 [1]. He proposed that immobile nucleic acid junctions generated from DNA could be assembled into complex and multi-dimensional DNA lattices. Unconventionally, DNA in this context was studied as a non-biological engineering material to build up a large-scale nanosystem by DNA self-assembly, which is a bottom-up approach to construct DNA nanostructures by preparing nucleobases with strict base pairing rules.

Attempts to realize Seeman's proposition were pursued in the following decades. A cube [3] and a truncated octahedron [4] made of DNA were synthesized. However, it was recognized that the structural building blocks in these designs were not solid enough to form multi-dimensional lattices. The double cross-over (DX) motif as a structural building block for DNA complexes, which was proven effective in 1993 [5], remains the central motif in DNA nanotechnology. Together with the concept of DNA tiles [6], which is derived from Wang's theory of mathematical tiling [7], a 2D DNA lattice was designed and experimentally demonstrated [8].

The traditional approach to DNA self-assembly relies on the systematic design of a number of complementary oligonucleotides [9]. In order to obtain results with a moderate amount of errors, the oligonucleotides must have perfect equimolar stoichiometry, but such quality is difficult to obtain. Therefore, this difficulty limited the complexities of the assembled DNA nanostructures. Such problem was not overcome until Rothemund introduced DNA origami, a revolutionary approach for the construction of DNA nanostructures, in 2006 [2]. Rothemund showed that his approach did not need to meet the rigorous criteria that restrict the complexities in the traditional approach to DNA self-assembly. Thus, this high-yield and low-cost new approach has stimulated numerous researches and applications [10–12].

DNA origami is an ideal cornerstone in the applications of DNA self-assembly because it enables the constructions of a wide variety of DNA nanostructures in multiple dimensions. However, this paper only addresses developments of 2D DNA origami.

Soon after the publication of Rothemund's findings, Qian et al. created an analogic China map based on DNA origami, demonstrating that constructing complicated shapes in 2D is achievable [13]. Anderson et al. proved later that arbitrary shapes of 2D DNA origami within the area of 100×100 nm is programmable [14]. The size of 2D DNA nanostructure was successfully expanded to an array with edge dimension of 2 – 3 μ m by tiling strategy [15].

One of the recent developments in 2D DNA origami is programmable random DNA tilings that are able to construct even more remarkable DNA nanostructures. Tikhomirov et al. [16] provided a framework that scales up the complexity and diversity of synthetic molecular devices that are organized by DNA nanostructures only by applying simple local assembly rules. It was shown that with planar networks of random loops, mazes and trees, the complexity of uniquely addressable DNA nanostructures could be increased by 10 to 100 times. Later on, their research continued and demonstrated that even more arbitrary complex nanostructures could be created by fractal assembly [17]. In order to readily access the fractal assembly approach, the paper presented a software tool, FracTile Compiler [18], which is able to produce DNA nanostructures according to the desired input image of a user.

DNA nanotechnology is an outstanding example of engineered molecular self-assembly, which is seen as a bottom-up approach to form massive structures by preparing small molecules with a defined arrangement. Molecular self-assembly has reached a point where it is possible to construct large and complex structures. It benefits the studies of structural biology and biophysics as well as the applications of nanomedicines. However, the assembly of non-biological nanosystems remains in a theoretical proposition and has not yet achieved a breakthrough. This paper believes that the investigation of DNA self-assembly is a search in nature to obtain insights into manufacturing artificial nanosystems with desired features.

The rest of this paper is organized as follows. Section 2 illustrates background knowledge with regards to DNA nanotechnology and defines the terminology used in this paper. Section 3 introduces the gradual change from single to multi-component nanostructure design in 2D DNA origami. Section 4 presents two state-of-the-art approaches to 2D DNA origami. Fi-

nally, section 5 concludes the paper by discussing the main technologies introduced in the paper and analyzing future possibilities in this field.

2 Background

This section starts with the introduction of DNA and its role in nano-engineering, providing the basic knowledge of the primitives in DNA nanotechnology. Following is the key concept in this paper, DNA origami. Finally, DNA tiling is briefly illustrated, because it is an essential theoretical methodology in scaling up the DNA nanostructures through a programmable multi-component system design approach.

2.1 DNA as a substrate in Nanoengineering

DNA is a molecule that has two forms: single-stranded DNA (ss-DNA) and double-stranded DNA (ds-DNA) [19]. A DNA strand, which is also known as a polynucleotide, is composed of multiple nucleotides. The exact nucleobase on a nucleotide differentiates it from others. There are four forms of nucleobases: cytosine (C), guanine (G), adenine (A) and thymine (T).

A ds-DNA is constructed by two strands that are bonded together based on base pairing rules (C with G and A with T). The two strands of a ds-DNA have the same biological information and wind around in opposite directions to each other, resulting in an antiparallel structure as shown in Figure 1 (c). DNA self-assembly can be roughly defined as the autonomous process of strands forming a ds-DNA with base pairing rules. This process may happen when the system of DNA strands has required complementary nucleotides, medium and driving forces.

One spiral coil of a ds-DNA has a length of 3.4nm and a width of 2nm. Notably, the persistence of ds-DNA is higher than ss-DNA [21, 22], which makes ss-DNA easy to manipulate whereas ds-DNA exist to be stable. Moreover, the thermodynamic properties of DNA are predictable in the self-assembly process [23] and reverse engineering of DNA is achievable [24]. The above characteristics qualify DNA as a substrate in nanoengineering.

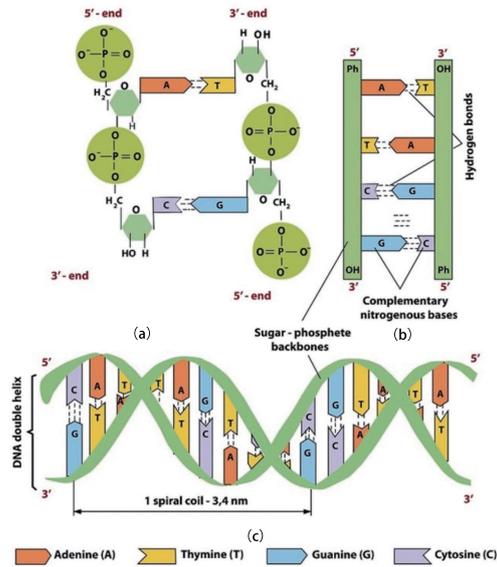


Figure 1. Structure of DNA (a) The molecule view of two bounded DNA strands. (b) The macroscopic view of two bounded DNA strands. (c) The double helix structure of ds-DNA. [20]

2.2 DNA Origami

In DNA origami, a long ss-DNA is chosen as the scaffold strand, and it is folded by hundreds of staple strands which are short synthetic strands. The staples (short for staple strands) are strands that provide complementary nucleotide bases according to the Watson-Crick model for the scaffold.

In previous research, it was found that parallel helices in DNA lattices are not strongly bound [25], hence the stable DNA lattice structure requires ‘stickers’ to clinch helices together. A periodic collection of crossovers created by a staple plays the role of stickers in DNA origami design.

The first step of DNA origami design includes supplying an odd number of half-turns (half of one spiral coil in a DNA double helix) of crossovers. Secondly, a scaffold strand is introduced and folded until it replaces one of the two strands in every helix. During this step, a seam where the folding path does not cross appears. Thirdly, staples are created to stabilize the structure and strengthen the seam. As shown in Figure 2 **d**, a nick emerges where the two staples meet. Hence, pairs of adjacent staples are merged to create larger binding domains for staples (see Figure 2 **e**).

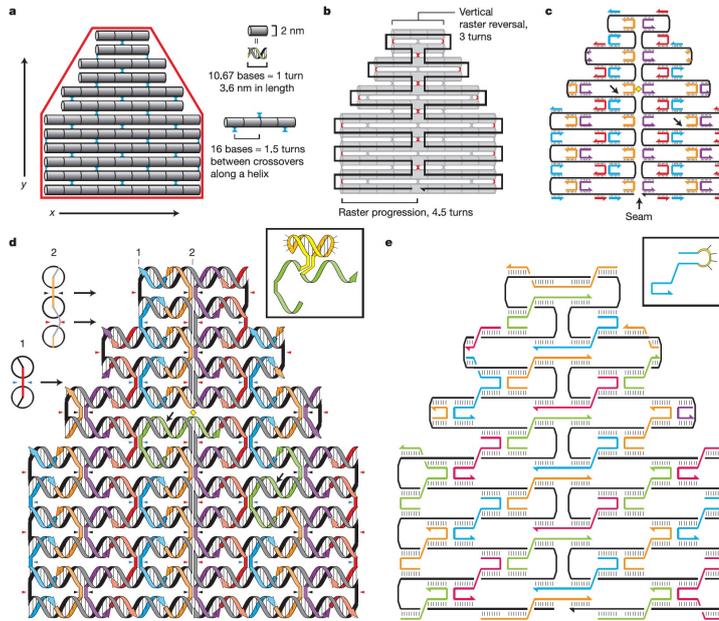


Figure 2. Design Process of DNA Origami **a.** The intended shape of DNA origami is outlined by the red and crossovers marked in blue are introduced. **b.** A scaffold strand is folded and replaces one of the two strands in every helix. More crossovers (marked in red) are introduced. **c.** A first design of the DNA origami. **d.** The same state as **c** with strands drawn as helices. **e.** The final design of the DNA origami. [2]

Rothemund [2] has created six shapes including a star, figures with holes and basic geometric figures using M13mp18 as a scaffold strand. In addition, he demonstrated that patterning and combining DNA origami is possible by depicting staple strands as binary pixels.

2.3 DNA Tiling

DNA tiling was first introduced as a technique to algorithmically assemble 2D DNA crystal [6]. This technique is derived from the mathematical tiling theory [26] with DX molecular DNA tiles acting as building blocks. Therefore, it is possible that nanoscale periodic nanostructures can be created from patterning the tiles.

DAO (double crossover, antiparallel, odd spacing) and DAE (double crossover, antiparallel, odd spacing) molecules are stable DX molecules [5] and are thus used as DNA tiles in [6]. A DAO molecule has four strands, each participates in both helices; whereas a DAE molecule has five strands, only three of them participate in both helices (see Figure 3 **d**). Each corner of

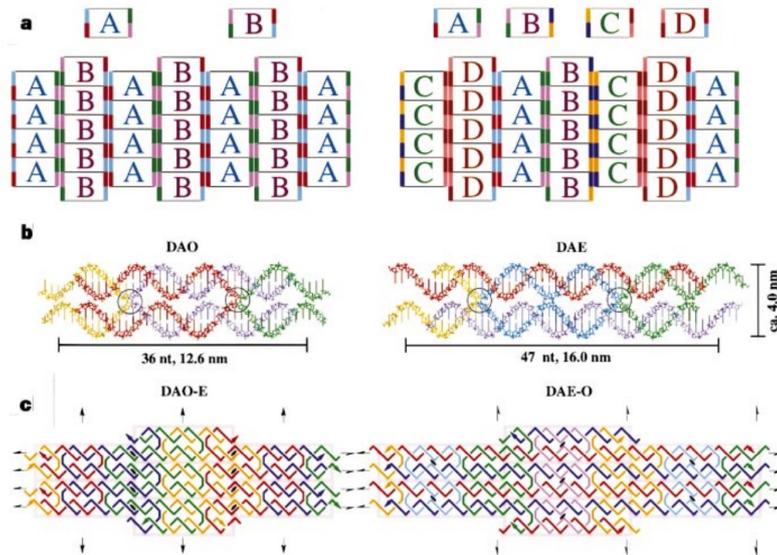


Figure 3. Design of DNA tiles **a.** Logical structures of DNA tiling using two and four types of DNA tiles, respectively. **b.** A DAO molecule and a DAE molecule. **c.** The topologies produced by DNA tiles, where each grey section is a DNA tile. [6]

a DNA tile has a single-stranded sticky end, enabling other DNA tiles to bind with. Therefore, specific binding of different tiles is controlled by the types of nucleotides of the sticky ends.

Although DNA tiling was proposed as rather a theoretical strategy than a practical solution to produce complex nanostructures, the idea has long been an inspiration to build up the complexity of DNA lattices [16, 17, 27].

3 Programmable 2D DNA Origami from Single to Multi-Component Nanostructure Design

DNA origami is by far the most popular way of assembling 2D DNA nanostructure. However, nanostructures assembled using solely this approach suffer from size limitations. As Rothmund suggested [2], patterning and combining DNA origami with different shapes have low success rates, which is probably due to poor stoichiometry and insensitive concentrations of extended staples. That is to say, high complexity of a single-component 2D DNA nanostructure is supported by DNA origami, whereas this nanostructure cannot be scaled up arbitrarily as difficulties exist when designing a multi-component system.

Combining DNA origami and DNA tiling can break the size limitations. DNA origami tiles usually are DNA origami formed tiles, each contains sticky ends to associate with other tiles. Periodic arrays of 2D DNA origami tiles have been created successfully by Liu et al. [15], proving that the construction of a multi-component nanostructure by DNA origami tiling is achievable.

Although Liu's finding freed the size restraint of 2D DNA origami, it did not increase the complexity of possible nanostructures because regular DNA tiles have the same capability as the DNA origami tiles in the context of assembling nanostructures with periodic patterns.

To generate greater diversity and pattern complexity of a multi-component nanostructure, Tikhomirov et al. [16] presented a strategy using random DNA tilings to create algorithmic arrays of DNA origami tiles. Compared to regular DNA tiles considered as algorithmic arrays, DNA origami tiles have more programmable interactions and can accommodate more complex internal patterns, and thus can lead to more diverse and convoluted outcomes. Later on, fractal assembly was presented to construct even more arbitrary nanostructures with greater sizes [17].

4 State-of-the-art Studies on 2D DNA origami

4.1 Random DNA Tilings by Programmable Disorder

Herein the paper discusses the details of the random DNA tiling approach presented in [16]. The approach is guided by stochastic Truchet tiling, which introduces randomness to DNA origami tiles (arrays) that act as Truchet tiles. A Truchet tile is a square tile decorated with a kind of pattern that is not rotationally symmetric [28], and it may result in different patterns if it turns to a different orientation (See Figure 4 a). Therefore, Truchet tiles can form various patterns when they are randomly placed.

A DNA origami tile in the random DNA tiling approach is designed with a maximal surface area (See Figure 4 b) so that it can support more pat-

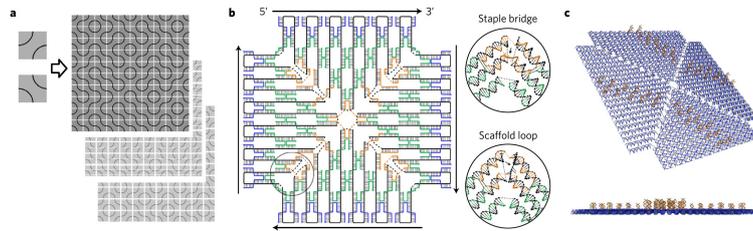


Figure 4. Implementation of Truchet Arrays **a.** Truchet tile with two arcs, which has two possible orientations. Such tile can form an arbitrary tiling as shown on the right. **b.** A DNA origami tile constructed by four isosceles triangles. **c.** A 3D Model of the DNA origami tile, which is designed for calculations. [16]

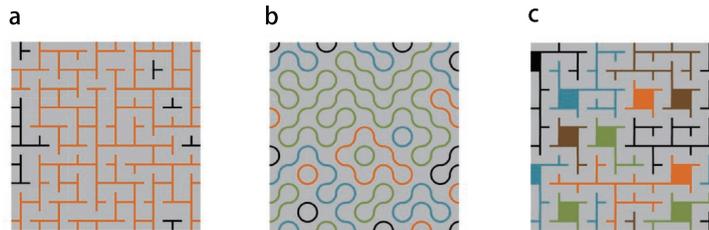


Figure 5. Forms of Planar Network **a.** An example of a maze network. **b.** An example of a loop network. **c.** An example of a tree network. [16]

terns. Note that the tile has a four-fold rotational symmetry, enabling it to rotate to four orientations. Each tile has two inputs and two outputs so that it can also interact with other tiles.

The randomness in the orientations of the tiles is introduced by a framework that creates programmable disorder. Such framework can construct three forms of a planar network (See Figure 5): loop (closed lines with no branches), maze (lines with loops and branches) and tree (lines with branches but no loops) [16]. The process of defining a random array in this framework includes choosing an arbitrary type of tile, setting random orientation to each tile to create a form of network, and fitting the tiles to a finite grid.

Tikhomirov et al. demonstrated that uniquely addressable DNA nanostructures with higher complexity than before were successfully created using the random DNA tiling approach. In addition, their research motivated the circuit and device design with desired functions.

4.2 Fractal Assembly

In their previous work [16], Tikhomirov et al. provided a successful scaling-up strategy using DNA origami tiles, but the rotation symmetry of Truchet arrays does not support true arbitrariness and it is impossible to create arrays larger than a grid size of 4×4 (see the supplementary note of [16]). Therefore, fractal assembly as a new method to overcome the limitation was purposed [17].

Fractal assembly is a hierarchical multistage assembly process which enables DNA origami tiles to create arbitrary patterns with large size. Three rules are guiding the local assembly of DNA tiles: 1) giving and receiving; 2) rotation; 3) edge code. ‘Giving and receiving’ rule indicates that for each of the four tiles, the two interactive edges of a tile should be either both giving or both receiving (See Figure 6 **d**). ‘Rotation’ rule indicates that for each of the four tiles, two tiles have the same orientation while the others are rotated 90° (See Figure 6 **d**). ‘Edge code’ indicates that each edge of a tile has a coding of 0s and 1s, where 0 represents a scaffold loop, and 1 represents a staple (See Figure 6 **e**). These rules are applied recursively in all stages of the fractal assembly, and thus guarantee the stability and self-similarity of the self-assembly process.

DNA origami tile designed in Figure 6 **b** was used to demonstrate the fractal assembly process. In contrast with setting Truchet tile pattern on top, here staples of DNA origami are considered as uniquely addressable pixels (‘on’ if with modification or extension, ‘off’ if not). Thus, 136 pixels are derived from a DNA origami tile and arbitrary binary patterns are constructible using these DNA origami tiles.

A software tool, FracTile, was developed by Peterson [18] to demonstrate the fractal assembly process. A user can upload an image to FracTile and a corresponding DNA tiling will be generated according to the user’s specifications with details of DNA origami tiles information. Fractal assembly shows that it is possible to scale up the complexity of DNA nanostructures to arbitrary binary patterns and that top-down approach to generate DNA nanostructure is feasible.

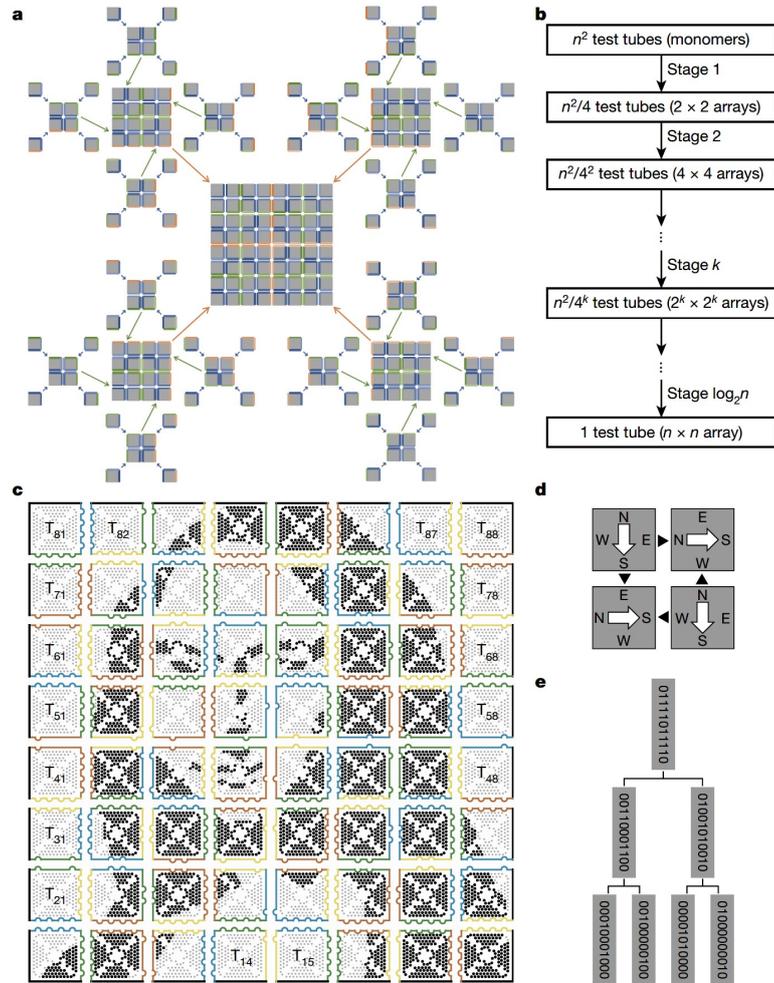


Figure 6. Fractal Assembly Process **a.** A three-stage assembly process that generates an 8×8 DNA tiling. Note that the colored edges are illustrated because two edges can be bonded if and only if they share the same color. **b.** Generating $n \times n$ tiling in $\log_2 n$ stages. **c.** DNA origami tiles that form a pattern of Mona Lisa using fractal assembly. The black dots and grey dots denote ‘on’ and ‘off’ staples, respectively. Indentations and bumps of the edges of each tile represent receiving and giving edges, respectively. Note that the four edges of a tile are colored: North with blue, South with orange, East with green and West with yellow. **d.** Abstract tiles to demonstrate ‘giving and receiving’ and ‘rotation’ rule. **e.** An edge code that derives to multiple edge codes in the next stages. [17]

5 Conclusion

The groundwork of DNA nanotechnology [1], which Seeman considered as the greatest success at the time, was first rejected in Nature by editors who questioned the role of biology in Seeman’s work. Indeed, peeling off the biology layer of DNA to study it as building blocks in nanoengineering was unconventional, not to mention the notion was simply an idea. DNA nanotechnology was not regarded as a pragmatic field so that little attention was paid to it. Fortunately, the essence of science is that it can

always attract scientists to believe in what others may find pointless.

A tipping point for DNA nanotechnology came when DNA origami was introduced. Since then, numerous scientists dived into the field of building nanosystems with DNA. DNA origami appeals to the outside world of academics because it provides an outlook of constructing complex DNA nanostructures with a high-yield and low-cost methodology.

Given that this paper emphasizes on reviewing 2D DNA origami, this paper briefly discusses two recent studies based on this technology, showing that remarkable 2D DNA nanostructures have been created. In 2016, algorithmic 2D arrays of DNA origami were successfully created [16], in which programmable disorder was added to the nanostructures. Further, in 2017, staples in DNA origami tiles were treated as pixels [17]. Thus, it enabled arbitrary binary patterns to be expressed through DNA nanostructures.

Researches by Tikhomirov et al. [16,17] are outstanding examples of demonstrating 2D DNA origami is effective in scaling up the complexity of 2D DNA nanostructures. However, the problem of if 2D DNA origami is limited in size remains. When the DNA nanostructure version of Mona Lisa in Figure 6 was shown to the world, reporters complimented that it was the smallest version of Mona Lisa. Unfortunately, it is not a compliment to the scientists, because they want to enlarge the grid size so that the nanostructures can be volumized. Although nanostructures having a grid size of 8×8 have been created successfully, the synthesizing process had a low yield [17]. 2D DNA nanostructures of larger grid size are not yet created successfully. Therefore, whether the size of DNA nanostructures can be increased while the current complexity maintains is still an open question.

References

- [1] N. C. Seeman, "Nucleic acid junctions and lattices," *Journal of Theoretical Biology*, vol. 99, pp. 237–247, 1982.
- [2] P. W. K. Rothemund, "Folding DNA to create nanoscale shapes and patterns," *Nature*, vol. 440, pp. 297 EP –, Mar 2006. Article.
- [3] J. Chen and N. C. Seeman, "Synthesis from DNA of a molecule with the connectivity of a cube," *Nature*, vol. 350, no. 6319, pp. 631–633, 1991.
- [4] W. M. Shih, J. D. Quispe, and G. F. Joyce, "A 1.7-kilobase single-stranded DNA that folds into a nanoscale octahedron," *Nature*, vol. 427, pp. 618 EP –, Feb 2004.
- [5] T. J. Fu and N. C. Seeman, "DNA double-crossover molecules," *Biochemistry*, vol. 32, pp. 3211–3220, Apr 1993.
- [6] E. Winfree, F. Liu, L. A. Wenzler, and N. C. Seeman, "Design and self-assembly of two-dimensional DNA crystals," *Nature*, vol. 394, pp. 539 EP –, Aug 1998. Article.
- [7] H. Wang, "Proving theorems by pattern recognition — ii," *The Bell System Technical Journal*, vol. 40, pp. 1–41, Jan 1961.
- [8] R. F. Service, "DNA nanotechnology grows up," *Science*, vol. 332, no. 6034, pp. 1140–1143, 2011.
- [9] J. K. Nangreave, D. Han, Y. Liu, and H. Yan, "DNA origami: a history and current perspective.," *Current opinion in chemical biology*, vol. 14 5, pp. 608–15, 2010.
- [10] H. Yan, S. H. Park, G. Finkelstein, J. H. Reif, and T. H. LaBean, "DNA-templated self-assembly of protein arrays and highly conductive nanowires," *Science*, vol. 301, no. 5641, pp. 1882–1884, 2003.
- [11] A. Kuzyk, R. Schreiber, Z. Fan, G. Pardatscher, E.-M. Roller, A. Högele, F. C. Simmel, A. O. Govorov, and T. Liedl, "DNA-based self-assembly of chiral plasmonic nanostructures with tailored optical response," *Nature*, vol. 483, pp. 311 EP –, Mar 2012.
- [12] E. S. Andersen, M. Dong, M. M. Nielsen, K. Jahn, R. Subramani, W. Mamdouh, M. M. Golas, B. Sander, H. Stark, C. L. P. Oliveira, J. S. Pedersen, V. Birkedal, F. Besenbacher, K. V. Gothelf, and J. Kjems, "Self-assembly of a nanoscale DNA box with a controllable lid," *Nature*, vol. 459, pp. 73 EP –, May 2009.
- [13] L. Qian, Y. Wang, Z. Zhang, J. Zhao, D. Pan, Y. Zhang, Q. Liu, C. Fan, J. Hu, and L. He, "Analogic china map constructed by DNA," *Chinese Science Bulletin*, vol. 51, pp. 2973–2976, Dec 2006.
- [14] E. S. Andersen, M. Dong, M. M. Nielsen, K. Jahn, A. Lind-Thomsen, W. Mamdouh, K. V. Gothelf, F. Besenbacher, and J. Kjems, "DNA origami design of dolphin-shaped structures with flexible tails," *ACS Nano*, vol. 2, no. 6, pp. 1213–1218, 2008.

- [15] W. Liu, H. Zhong, R. Wang, and N. C. Seeman, "Crystalline two-dimensional DNA-origami arrays," *Angewandte Chemie International Edition*, vol. 50, no. 1, pp. 264–267.
- [16] G. Tikhomirov, P. Petersen, and L. Qian, "Programmable disorder in random DNA tilings," *Nature Nanotechnology*, vol. 12, pp. 251 EP –, Nov 2016. Article.
- [17] G. Tikhomirov, P. Petersen, and L. Qian, "Fractal assembly of micrometre-scale DNA origami arrays with arbitrary patterns," *Nature*, vol. 552, pp. 67 EP –, Dec 2017.
- [18] P. Petersen, "Fractile compiler," 2017. Available: <http://qianlab.caltech.edu/FracTileCompiler>.
- [19] V. K. Gautam, *Dynamically Controlled DNA Tiles: Looking Beyond the Self-assembly of Static Two-dimensional DNA-based Computational Structures*. PhD thesis, Norwegian University of Science and Technology, 2017.
- [20] S. Cheriyaedath, "Structure of DNA." Available: <https://www.news-medical.net/life-sciences/Structure-of-DNA.aspx>.
- [21] G. S. Manning, "Three persistence lengths for a stiff polymer with an application to DNA b-z junctions," *Biopolymers*, vol. 27, no. 10, pp. 1529–1542.
- [22] J. Yan and J. F. Marko, "Localized single-stranded bubble mechanism for cyclization of short double helix DNA," *Phys. Rev. Lett.*, vol. 93, p. 108108, Sep 2004.
- [23] J. SantaLucia, "A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics," *Proceedings of the National Academy of Sciences*, vol. 95, no. 4, pp. 1460–1465, 1998.
- [24] J. N. Zadeh, C. D. Steenberg, J. S. Bois, B. R. Wolfe, M. B. Pierce, A. R. Khan, R. M. Dirks, and N. A. Pierce, "Nupack: Analysis and design of nucleic acid systems," *Journal of Computational Chemistry*, vol. 32, no. 1, pp. 170–173.
- [25] P. W. K. Rothemund, A. Ekani-Nkodo, N. Papadakis, A. Kumar, D. K. Fygen-son, and E. Winfree, "Design and characterization of programmable DNA nanotubes," *Journal of the American Chemical Society*, vol. 126, pp. 16344–16352, Dec 2004.
- [26] B. Grünbaum and G. C. Shephard, *Tilings and patterns*. New York, NY: Freeman, 1987.
- [27] N. C. Seeman, "DNA in a material world," *Nature*, vol. 421, pp. 427 EP –, Jan 2003.
- [28] S. Truchet, "Mémoire sur les combinaisons," *Memories of the Royal Academy of Sciences*, vol. 1704, pp. 363–372, 1704.

Tor: Attack and Defense

Nam Xuan Nguyen

xuan.2.nguyen@aalto.fi

Tutor: Lachlan Gunn

Abstract

Tor is the most widely known anonymous communication network today, obscuring the real identities of millions of clients on a daily basis. Since Tor was first introduced and deployed, many works have been published which enhance its security, performance, and usability. Given the significance of this research area, we aim to provide the audience with an overview of current research about attack techniques to break Tor protection as well as methods to block them. We focus on the most powerful and realistic attacks that are Website fingerprinting and End-to-end confirmation attacks as they are continuous unresolved threats.

KEYWORDS: *Tor browser, Tor network, Tor protocol, anonymity system, Website fingerprinting, End-to-end confirmation attacks.*

1 Introduction

In the early days of the Internet, users could communicate with each other in relative obscurity. This belief was so popular that it could be found in a famous comic made by Steiner et al. (1993): *On the Internet, nobody knows you're a dog* [4]. However, Internet anonymity has changed a lot

which today ISPs routinely provide their users' personal information to a third party, including marketers and governments.

Nevertheless, there are many situations that require preserving anonymity and privacy on the Internet. In [17], Edman et al. enumerate some examples such as military cyber soldiers, law enforcement agents, and private citizens. Military cyber soldiers desire to conceal their secret locations from which they send and receive packets on the Internet. Intelligence officers may need to monitor suspicious network systems without the awareness of system administrators. Furthermore, many people like to freely surf the Web without worrying about surveillance and their personal detail being collected. Fortunately, these requirements can be met by anonymity systems.

An anonymity system is a communication layer providing protocols and tools to hide users' identities and prevent computer systems network traffic from being analyzed. These systems allow Internet users to communicate with a high probability that tracing their IP address is impossible.

Although there has been much research on anonymity systems, most of them only exist on paper. A few systems such as Tor [7], I2P [3], and Freenet [2] are available for users. In fact, for many years, Tor has been the most popular system for concealing identity on the Internet. In 2013, this system had about 4 million users [6]. Currently, more than 2 million daily users connect to the Internet using Tor. Tor is free, open-source software and an open network for the purpose of anonymous online activity. This software enables individual clients to freely browse the Web and prevent these websites from tracking customer's location. It also provides Tor *onion services* for users who need to deploy their web services without revealing their location.

Despite the fact that Tor was originally designed for protecting US naval forces communication, it is currently used for many other purposes [9]. Normal users use Tor to protect their privacy from unethical marketers, anonymously research sensitive topics as well as to skirt surveillance and circumvent censorship. From the journalists' point of view, Tor can be adopted for their safety. SecureDrop [5], an open-source whistleblower submission system, integrates Tor as part of it to ensure that the document sharing processes are anonymous. Similarly, in China, journalists employ Tor to report about local events in the hope of the more equitable and transparent society. Tor also helps business executives, military and IT professionals to complete their tasks in secret.

As well as Tor, I2P [3] and Freenet [2] have also been widely used throughout the years. I2P is another anonymous communication network which runs on peer-to-peer and low latency relay networks. However, unlike Tor, I2P is optimized to build a private network on the top the Internet [1]. Although Freenet [2] is also a peer-to-peer platform, it is a high latency network that is quite different from Tor. Its main function is to provide an anonymous distributed encrypted data store. For these reasons, it can be seen that Tor is the most suitable option for people to anonymously access public web services, therefore it has become the most popular anonymity system so far. On the other hand, due to the privacy benefits that Tor provides, criminals also make use of it for nefarious purposes [17, 25]. Therefore, intelligence agencies such as American National Security Agency (NSA) and the British GCHQ continuously seek for Tor network weaknesses and perform various techniques to capture Tor user IP addresses [8]. On the other hand, Tor developers and supporters ceaselessly improve Tor, make it stronger and less vulnerable. The war between both sides contains interesting technical details which can be used for making better Tor or anonymity systems in the future as well as other types of technology in the field of cybersecurity.

This paper discusses attack techniques which have been used to bypass Tor protection and the corresponding methods to block these attacks. These attacks are based on weaknesses discovered from the time Tor was introduced to the present, including fingerprint attacks and end-to-end confirmation attacks.

Following the introduction section, the second section discusses the technical details of Tor. The third section discusses attack techniques and corresponding mitigation approaches. The last section concludes the paper and discusses the methods that can be used to protect as well as to attack Tor in the future.

2 Tor

This section presents an overview of Tor from a technical point of view. Tor is a circuit-based low-latency anonymous communication service which overcomes several weaknesses in the original Onion routing system [15]. Onion Routing allows to established anonymous connections through a sequence of *onion routers* [26]. An onion router can only identify its next and previous routers along the route. This mechanism hides the senders'

identity from the receivers as well as both the senders' and receivers' real IP addresses from the eavesdroppers. However, the onion routers along the route have to be selected at connection setup and the list of routers is sent along with the data. The data is encapsulated in a layered data structure called *onion*. Each onion layer contains the next router information in a route. When receiving an onion, the onion router decrypts the corresponding layer to obtain the IP address of the next hop, then it forwards the embedded onion to that router. The last onion router transfers data to the destination. Besides Onion routing, Tor has added several improvements which allow it to better protect the users' identity.[15].

3 Attacks and Defences

Several attacks on Tor are recorded or proposed by researchers these are designed to undermine users' anonymity [10]. Many attacks are passive attacks, including attacks from AS-Level adversaries [17] and website fingerprinting attacks [19]. These attacks allow the adversaries to monitor the traffic without manipulation in order to compromise the Tor users' identity. The other major attack type is the active attack [10], such as End-to-End confirmation attacks, path selection attacks, sniper attacks, side channel attacks, traffic and timing attacks, which require attackers to modify Tor network traffic. Although most attacks only exist on paper, website fingerprinting and end-to-end confirmation attack have continuously been a real threat to the anonymity provided by Tor. For that reason, this section focuses on these two techniques, especially in the processes that they evolve from ideas to the real-world attacks.

3.1 Website fingerprinting attacks

Web fingerprinting (WF) is a special traffic analysis technique which allows eavesdroppers, such as system administrators, ISPs and the governments, to extract users' behaviors without decrypting the data [23]. They observe the encrypted traffic and form patterns using certain features of the communication data, including the packet size, the timing and the volume of transferred data. Although the data remains encrypted, the adversaries can match the obtained patterns to a database of known pattern, e.g., list of specific websites. In the case that the local eavesdroppers and wiretapper can observe the connection between the original senders

and the first onion node of the Tor network, this attack may map each user to the websites he/she visited, thus breaking one of Tor privacy guarantees.

The term *web fingerprinting* was originated by Hintz et al. in [19]. He performed a proof-of-concept attack on a free web proxy called *Safe Web*, in which he distinguished between 5 websites using the similarities of object sizes and the detection rate was from 45% to 75%. In order to recognize a larger number of websites, machine learning algorithms are used in many works. In [18], Herrmann et al. introduced an approach which employs a Multinomial Naïve-Bayes classifier to match the encrypted traffic pattern to a corresponding pattern in a closed-world database which contains 775 websites. Closed-world database is a set of known items. However, the approach only achieved a recognition rate of 2.96% that is insufficient to identify most websites. In comparison, another approach based on Support Vector Machines (SVMs) resulted in a better recognition rate, 55%, on the same database [23]. This work facilitates the classification by selecting the volume, time, and direction of the traffic as the classification features. Then, SVM classifier was exploited with these features, thus leading to a superior result (55% compared to 2.96%).

Subsequently, many researchers have proposed various methods to improve this attack, boosting the true positive rate (TPR) as well as reducing the false positive rate (FPR). In [12], Cai et al. presented a novel technique for calculating the similarity of packet traces obtained when monitoring the process of loading web pages of Tor browser. These traces are converted into strings, thus being comparable. Then, the authors applied the Damerau-Levenshtein distance [12] as the metric to compare them. The reason for this choice is that DamerauLeven-Levenshtein allows dynamic operators, such as insertions, deletions, substitutions, which are suitable for situations in the network, including packet drops, re-transmissions, re-orderings, and gentle changes in web page content. For testing purpose, the authors selected a set of 100 web pages from Alexa Top 1000 web pages. They performed experiments which identified the web pages in the 100-webpage set that a victim visited. As a result, the accuracy rate is ranging from 52.2% to 83.7% depending on the methods which protect Tor connections, such as randomized pipelining, randomized pipelining + randomized traffic and no protection.

Although accomplishing very high TPR, the works above remain a relatively high FPR. In addition, These works only test the attack under the

closed-world model, thus not testing outside of a pre-defined list of web pages. However, a technique which was designed in [27] has successfully reduced the FPR to far below 1%, which is a practical level to use in real life. The researchers also present the performance of their techniques under the open-world setting, thus being more realistic compared to the previous works. In [27], the authors applied the k-Nearest Neighbour classifier with the multi-modal property of web pages. This technique is used to monitor a list of 100 web pages and identify a subset of web pages that a user is visiting. It results in a TPR of 85% while its FPR stays at a low level of 0.6%.

The attack designed by Wang et al. [28] achieves a significant TPR over 95% in the open-world setting, while the FPR is under 0.2%. Their technique also showed considerable improvement that achieved a TPR of 91% in the closed-world dataset. In both cases, the websites in the database are also selected from Alexa top 1000 and Alexa top 100 websites as the previous works. As a result, these experimental results can be compared to previous works. In order to attain this substantial result, Wang et al. analyzed the Tor network traffic in Tor cells instead of packets at the TCP/IP level. Additionally, the authors proposed new distance-based metrics to more accurately measure the similarity level of traffic patterns. These new metrics include modified versions of optimal string alignment distance and Damerau-Levenshtein distance, removing substitutions, different costs for incoming/outgoing packets, varying transposition cost, and fast Levenshtein-like distance [28].

Although the TPR and FPR in the experiments mentioned above are impressive even in the open-world setting, their performance may be hindered if the following conditions are unsatisfied [30]. First, the adversaries' classification training dataset has to be collected under the same conditions as those of the client. Second, the Tor browser completely loads a page before continuing to load another. Next, the start and end positions of a web page have to be known. Finally, noise is found in the communication data. In an effort to realize the capabilities of web fingerprint attack, in [30], it is shown that the adversaries can launch this attack under realistic conditions by operating additional techniques to tackle the unsatisfied conditions. In detail, for the limitations coming from the first condition, attackers only need a small amount of data to attack their targets, thus leading to the fact that it is reasonable to maintain the freshness of the data. Similarly, the problems with the second and the third

conditions can be handled by a technique called *splitting*. It allows to discriminate between different web pages which are loading in parallel or sequentially. In [30], two splitting methods are demonstrated, including time-based and classification-based splitting. Then, the problem with background noise can be handled developing a noise classifier. This classifier will remove various types of noise, and in the paper, they are audio streaming and file download noise.

3.2 Website fingerprint protection

This section describes the various approaches to prevent the website fingerprint attack. In general, these approaches focus on methods to modify packets, so that the network traffic from and to different websites is indistinguishable, which blocks the attackers to identify the destination websites of Tor users. The mitigation methods are described in the **Table 1**. However, these techniques are shown to have their own limitations, and each of them can only prevent particular attack techniques [29]. None of them can absolutely protect Tor users from WF attacks. On the other

Defense	Description
Packet padding [16]	Obscuring the true length of packets by attaching redundant bytes to them.
Traffic morphing [32]	Modifying the network traffic coming from and to the destination website in order to mimic the packet length distribution of a target website
HTTP Obfuscation [21]	Only on client-side. Choosing a uniformly random τ , $0 < \tau < \ell$, and splitting the packet into two packets of size τ and $\ell - \tau$ (ℓ is incoming packet length and $\ell \neq MTU$).
Background noise [23]	Random data can be added as background noise to each page load, which makes the attackers more difficult to identify the target web page.
Tor pipelining & request order randomization [24]	The only technique that has been implemented by Tor developer. Enabling HTTP pipelining, choosing a randomized request instead of the first-come request from the request queue, and regularly randomizing the maximum connection limit.
Buffered Fixed-Length Obfuscator [16]	Constantly sending packets at unchanged intervals with a fixed length. Dummy data is sent in the case no data needs to be transferred.

Table 1. Description of various WF defenses

hand, in [31], Wang et al. claim that their work, Walkie-Talkie, is able to defend against all known WF attacks so far. Walkie-Talkie consists of two modules: half-duplex communication and burst molding. Half-duplex (communicating in both directions, but not simultaneously) is employed to generate *burst sequences*. A burst is a group of all incoming or outgoing Tor cells, fix-size pieces of data sent by Tor browser, and using burst sequences allow burst molding to mimic the non-sensitive web pages without adding any redundancy.

3.3 End-to-end confirmation attacks

End-to-end confirmation is continuously an unresolved threat to Tor users. Tor developers originally had no intention to design Tor to completely block this attack because it can defeat all low-latency anonymity systems. A low-latency system design was selected as Tor users need to surf websites not only anonymously but also smoothly and quickly. Therefore, this section shows the general idea of this attack and emphasize various techniques which make it become a real threat. In order to perform this attack, attackers are required to compromise both entry and exit nodes. This can be successfully done by various methods, e.g., in [11], adversaries may set up a few high-bandwidth nodes, and it is likely that new Tor clients' connections are established through these nodes. Next, attackers can observe a set of incoming traffic and a set of outgoing traffic and aim to match a target outgoing connection to corresponding incoming connection. To achieve this goal, in [11], Bauer et al. suggest that the attackers can divide the duration they record the traffic into adjacent time windows. In each window, the attackers count the number of packets of each traffic. For a combination of an incoming and outgoing traffic streams, the Pearson correlation coefficient π of these two traffic streams can be calculated by the following formula:

$$\pi = \frac{\sum_k ((x_k - \mu)(x'_k - \mu'))}{\sqrt{\sum_k (x_k - \mu)^2} \sqrt{\sum_k (x'_k - \mu')^2}}$$

In this formula, x_k and x'_k denote the number of packets in incoming and outgoing traffic, respectively, for window k^{th} while μ and μ' are the means of the packet counts of the two traffic. The value π ranges between -1 and 1, and if $\pi > t$ (a pre-defined threshold) then both traffic streams belongs to the same connection.

One technique to improve these attacks described in [22] is called **Compromising Anonymity Using Packet Spinning**. This attack creates

loops in circuits (a circuit is a combination of an entry node, middle node, and the exit node) which become denial-of-service attacks targeting other legitimate onion routers. Therefore, these routers are unavailable so the malicious onion routers have more chance to be selected in circuits.

The technique proposed in [11] can also be used to enhance the power of end-to-end attacks. It is called **Low-Resource routing attack** that allows the malicious onion routers lies about their network resource. They appear to have the high-bandwidth capacity and high up-time, so they are more likely to be selected as the entry or exit nodes. Another attack is proposed in this paper. By recognizing patterns in the algorithm of building Tor circuit, this attack reveals the Tor path even before any payload data goes from the victim to the destination websites.

3.4 End-to-end confirmation protection

Since this attack affects all low-latency anonymity system, no defense can perfectly protect Tor users from all circumstances against this attack. The only method to completely prevent it is to change the Tor design which makes Tor slow down a lot and be unable to use. This section discusses a few works which make the adversaries more difficult to attack Tor users. The goal of these defense techniques is to raise the similarity between traffic of different clients as well as to reduce the correlation of traffic flows from the same user. In [14], the authors proposed a packet delay technique which prevents end-to-end confirmation attacks in mix networks [13], another type of anonymity system. This approach delays packets before forwarding them to the next mix. Each packet has a random delay which is different from the delay of other packets. In other words, the delay is a random variable and this variable follows the exponential distribution, which provides the theoretically the best anonymity in comparison to other statistical distributions. However, this protection can be defeated by another attack proposed in the same paper [14].

Another method can be employed to prevent an end-to-end attack that is called *defensive dropping* [20]. In this approach, the client sends a legitimate traffic stream along with a dummy traffic stream at constant rates. The dummy packets are dropped at the middle nodes, which are not entry or exit nodes, therefore only legitimate traffic stream comes to the exit node. This decreases the correlation between the incoming data stream to the entry node and the outgoing data stream from the exit node of the same user. In addition, as clients send packets at a constant rate, this

technique also boosts the correlation between traffic streams of different clients, thereby making them more indistinguishable.

4 Discussion and conclusion

Previous sections reviewed about two most powerful attacks against Tor. For WF attacks, it is shown that Walkie-Talkie can successfully block all known WF attack techniques so far. This defense is effective and easy-to-use, and it only requires minimal padding to confuse the attacker [31]. However, the key idea of WF attacks is to distinguish between traffic communication data stream of different websites, and this can be attained by introducing supervised machine learning algorithms as they allow to distinguish more effectively between objects rather than traditional classification algorithms. Therefore, an obvious approach to improving WF attack techniques is to apply the latest machine learning algorithms, including random forest or neural network. While attackers regularly have plenty of time and resource to continuously test the latest algorithms to find the most effective methods for WF attack, this type of attack remains an undeniable threat to Tor users in the future.

Although end-to-end confirmation attacks are capable of revealing the true identity of Tor users, it is a firm requirement for attackers to have an enormous resource to attack a specific user. To perform this attack, the adversaries need to control both entry and exit nodes. Therefore, in order to control both nodes which connections of a particular Tor user go through, the attackers need to control many Tor nodes by setting up many malicious nodes or compromising as many legitimate nodes as possible. Normal attackers are unable to achieve that.

References

- [1] An Introduction to Tor vs I2P. <https://www.ivpn.net/privacy-guides/an-introduction-to-tor-vs-i2p>. (Accessed on 02/28/2019).
- [2] Freenet. <https://freenetproject.org/author/freenet-project-inc.html>. (Accessed on 02/28/2019).
- [3] I2P Anonymous Network. <https://geti2p.net/en/>. (Accessed on 02/28/2019).
- [4] On the Internet, nobody knows you're a dog - Wikiwand. https://www.wikiwand.com/en/On_the_Internet_nobody_knows_you're_a_dog. (Accessed on 02/28/2019).
- [5] Share and accept documents securely - securedrop. <https://securedrop.org/>.

(Accessed on 02/28/2019).

- [6] Tor (anonymity network) - Wikiwand. [https://www.wikiwand.com/en/Tor_\(anonymity_network\)](https://www.wikiwand.com/en/Tor_(anonymity_network)). (Accessed on 02/28/2019).
- [7] Tor browser. <https://www.torproject.org/projects/torbrowser.html.en>. (Accessed on 02/28/2019).
- [8] Tor stinks presentation. <https://edwardsnowden.com/docs/doc/tor-stinks-presentation.pdf>. (Accessed on 02/28/2019).
- [9] Who uses Tor? <https://www.torproject.org/about/torusers.html.en>. (Accessed on 02/28/2019).
- [10] Mashael AlSabah and Ian Goldberg. Performance and security improvements for tor: A survey. *ACM Computing Surveys (CSUR)*, 49(2):32, 2016.
- [11] Kevin Bauer, Damon McCoy, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Low-resource routing attacks against Tor. In *Proceedings of the 2007 ACM workshop on Privacy in electronic society*, pages 11–20. ACM, 2007.
- [12] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. Touching from a distance: Website fingerprinting attacks and defenses. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 605–616. ACM, 2012.
- [13] David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.
- [14] George Danezis. The traffic analysis of continuous-time mixes. In *International Workshop on Privacy Enhancing Technologies*, pages 35–50. Springer, 2004.
- [15] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, Naval Research Lab Washington DC, 2004.
- [16] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-boo, I still see you: Why efficient traffic analysis countermeasures fail. In *2012 IEEE Symposium on Security and Privacy*, pages 332–346. IEEE, 2012.
- [17] Matthew Edman and Bülent Yener. On anonymity in an electronic society: A survey of anonymous communication systems. *ACM Computing Surveys (CSUR)*, 42(1):5, 2009.
- [18] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naïve-Bayes classifier. In *Proceedings of the 2009 ACM workshop on Cloud computing security*, pages 31–42. ACM, 2009.
- [19] Andrew Hintz. Fingerprinting websites using traffic analysis. In *International Workshop on Privacy Enhancing Technologies*, pages 171–178. Springer, 2002.

-
- [20] Brian N Levine, Michael K Reiter, Chenxi Wang, and Matthew Wright. Timing attacks in low-latency mix systems. In *International Conference on Financial Cryptography*, pages 251–265. Springer, 2004.
- [21] Xiapu Luo, Peng Zhou, Edmond WW Chan, Wenke Lee, Rocky KC Chang, and Roberto Perdisci. Httpos: Sealing information leaks with browser-side obfuscation of encrypted flows. In *NDSS*, volume 11. Citeseer, 2011.
- [22] Paul P Maglio and Teenie Matlock. Metaphors we surf the web by. In *Workshop on Personalized and Social Navigation in Information Space*, pages 1–9. Citeseer, 1998.
- [23] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website fingerprinting in onion routing based anonymization networks. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, pages 103–114. ACM, 2011.
- [24] Mike Perry. Experimental defense for website traffic fingerprinting. *Tor project Blog*. <https://blog.torproject.org/blog/experimental-defensewebsite-traffic-fingerprinting>, 2011.
- [25] John S Quarterman, Peter F Cassidy, and Gretchen K Phillips. Method and system for detecting distributed internet crime, October 15 2013. US Patent 8,560,413.
- [26] Michael G Reed, Paul F Syverson, and David M Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected areas in Communications*, 16(4):482–494, 1998.
- [27] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 143–157, 2014.
- [28] Tao Wang and Ian Goldberg. Improved website fingerprinting on Tor. In *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*, pages 201–212. ACM, 2013.
- [29] Tao Wang and Ian Goldberg. Comparing website fingerprinting attacks and defenses. Technical report, Technical Report 2013-30, CACR, 2013. <http://cacr.uwaterloo.ca/techreports> . . . , 2014.
- [30] Tao Wang and Ian Goldberg. On realistically attacking Tor with website fingerprinting. *Proceedings on Privacy Enhancing Technologies*, 2016(4):21–36, 2016.
- [31] Tao Wang and Ian Goldberg. Walkie-talkie: An efficient defense against passive website fingerprinting attacks. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*, pages 1375–1390, 2017.
- [32] Charles V Wright, Scott E Coull, and Fabian Monroe. Traffic morphing: An efficient defense against statistical traffic analysis. In *NDSS*, volume 9. Citeseer, 2009.

Designing solutions for pervasive displays: A survey

Andreas Hitz

andreas.hitz@aalto.fi

Tutor: Maria Montoya-Freire

Abstract

Pervasive displays are ubiquitous in our everyday life in many places, such as airports, train stations or shopping centers. A wide range of stakeholders is involved, including display owners, advertisers and obviously users. This paper gives an overview of the most important concepts of pervasive displays and covers challenges, current and future application scenarios, and new developments. It focuses in detail on the design aspects and requirements of such displays, including audience behavior and engagement, interaction techniques, system software, and display evaluation, by examining various literature sources. The paper's contribution is to give an overview of the wide and relevant area of research based on previous papers and contributions, while simultaneously demonstrating problems, possibilities, and the importance of designing suitable solutions for pervasive displays.

***KEYWORDS:** pervasive displays, digital signage, application scenarios, audience behavior, engagement, interaction techniques, system software, evaluation*

1 Introduction

Pervasive displays and digital signage are increasingly becoming more popular, due to a rise in demand of requesting information in all kinds of locations and occasions. They are present in many public places, such as train stations, airports, shopping centers, museums or central squares. Their main purpose is usually advertisement, but especially in public transport they are also used to display (real-time) information and announcements [6].

Recent developments in the areas of interaction and sensing technologies have opened up new technologies for the future design of such displays: context-based applications allow to take audience and situation into account to provide personalized information. Screen devices are more connected to large collaborative broadcasting systems that are interacting with each other, thus creating whole communication platforms [1].

However, these new developments also bring new challenges along, especially in the design of solutions, such as showing the right content, engaging users, or creating economically valuable solutions. The goal of this paper is to show application scenarios, analyze the design of pervasive displays, and demonstrate future developments in the research area.

The paper is structured as follows: after this introduction, important keywords are defined, and some challenges of public displays are briefly summarized. In the following, a few current application scenarios are described. The main part analyzes the different design aspects of pervasive display solutions: audience behavior and engagement, interaction techniques, system software and evaluation. Following a selection of possible future developments, a concise conclusion is drawn.

2 Fundamentals of pervasive displays

2.1 Definition, roles and functions

Digital signage is a form of electronic signage that uses digital technologies such as LCD, LED or projection to interactively show content on pervasive displays (see Figure 1 for an exemplary setting). Their function is to deliver content such as digital videos, weather data, images or simple text to their potential users, i.e., people who walk past and look at them. Such displays are usually located in public spaces, for instance, train sta-

tions, bus stops, stadiums, museums, retail stores, restaurants, cinemas or other buildings [4].



Figure 1. Example scenery with public displays [6]

Depending on the use case and the target audience, they fulfill all kinds of purposes: entertainment, merchandising, marketing or promotional advertising, and guidance. For instance, displays can show information about timetable schedules or retail products, or assist at indoor navigation in shopping malls. Furthermore, displays can show personalized digital content, or even influence the behavior and decision-making of users towards a certain product [1].

Display viewers are the people that see and interact with the displays; usually, these are pedestrians that are passing by the displays, or people waiting in areas surrounded with public screens. **Display advertisers** are firms that want to use the public displays for their own purposes, e.g., advertising and marketing. **Display owners** are usually companies, such as public transport providers or the owners of shopping centres.

The electronic displays are usually organized within a centrally managed network, which makes them individually addressable to show tailored content according to the needs of the display owner or advertiser. Other important technological aspects for displays are interactivity and context-awareness [19].

2.2 Challenges

Designers face several issues and challenges to develop useful solutions for pervasive displays. One of them is to provide a truly meaningful, valuable service for users who pass by pervasive displays that really makes them stop and observe it more closely. It is difficult to not overdo it and limit the amount of displayed information; the content should attract

users and grab their attention, without completely overstraining them and disturbing their original actions. The quantity and intensity of interactions should not become too much, as this can become a potential safety risk in public spaces [19]. An approach to solve this has been taken in the "Proxemic Peddler" project, which aims to tune the content of public displays in response to how passersby are attending them [22].

After the users have seen the displayed content and information, designers need to make sure that there are ways how this information can be carried away, so that it does not disappear immediately, but rather stays in peoples' minds. A persistent effect on the users needs to be measurable [15].

Engaging users, especially communicating the interactivity of screens, can be difficult. Another issue is the scheduling of content: when, how often, how long and in which order should it be displayed? In addition, the timing and form of transitions, context-sensitivity or content shown based on the interaction of the viewers with the displays are some of more questions that need to be answered. Taniguchi [19] mentions also that in the long run, user-generated content should be supported by public displays. Privacy [9] and accessibility [6] are important aspects that need to be considered as well.

3 Application scenarios

In the following, a few current application scenarios for public displays are presented [4].

3.1 Advertising

Advertising is a common use case of public displays. The display owners sell advertisement space to their customers, whose money is used to finance the display network. In places such as clothing stores or electronics warehouses, where people are making a purchase anyway, point-of-sale displays are used to advertise additional, related products in the meanwhile. Since public displays often come in large-scale sizes, they have an impact already due to their pure dimensions. Another essential factor that influences the success of a public display is its location and correct positioning, as this mainly determines how many pedestrians are possibly passing by there daily and have the chance to see it [2].

As people are exposed to a large number of ads in all kinds of situations every day, a successful solution must combine ads with valuable content to the viewer. A big challenge for advertisers is as well to ensure how viewers can take away the information from the display - this bit is naturally easier in print advertising, and also in the web, where links, behind which more information can be found, fulfill this function [2].

3.2 Information boards

A classic, popular use case of public displays are boards that simply present information: in airports, these have usually been split-flap systems, but they are nowadays increasingly replaced by complex connected systems with large display numbers [6]. Their information content is mainly to show the status of the system or inform about conditions such as the travel situation in city traffic networks, thus representing an informative rather than an interactive component.

3.3 Signage

The main purpose of signage has usually been to support the viewers with navigation hints and warnings, mostly used in traffic in the form of street signs. Due to sinking costs, the static signs are being replaced by dynamically changing objects ¹ [13]. In addition, screens are also employed to display meeting schedules in office environments that are placed in front of the corresponding conference rooms. Similarly, navigation inside buildings can also be realized, for instance in combination with mobile phones that read beacons [12].

4 Design of pervasive display solutions

This chapter contains detailed information about the design aspects and requirements of public displays: audience behavior, which includes concepts such as engaging users and grabbing their attention; interaction techniques, such as touch, mid-air gestures or mobile devices; system software, which includes content creation, scheduling, visualization and management; and evaluation tools and methods.

¹<https://broadsign.com/>

4.1 Audience behavior and engagement

To design a useful public display, it is important to know how viewers engage with and behave in relation to them - the created content, selected location, and user interface design need to be carefully chosen.

Behavior models

There are several models to describe audience behavior and engagement. These take two factors into account: *spatial aspects* (the behavior in terms of the viewer's spatial relationship to the display) and *temporal aspects* (the changing level of engagement over time). The relationship between users is another important factor that needs to be considered in this context. Four of the models are briefly introduced in the following (see Figure 2).

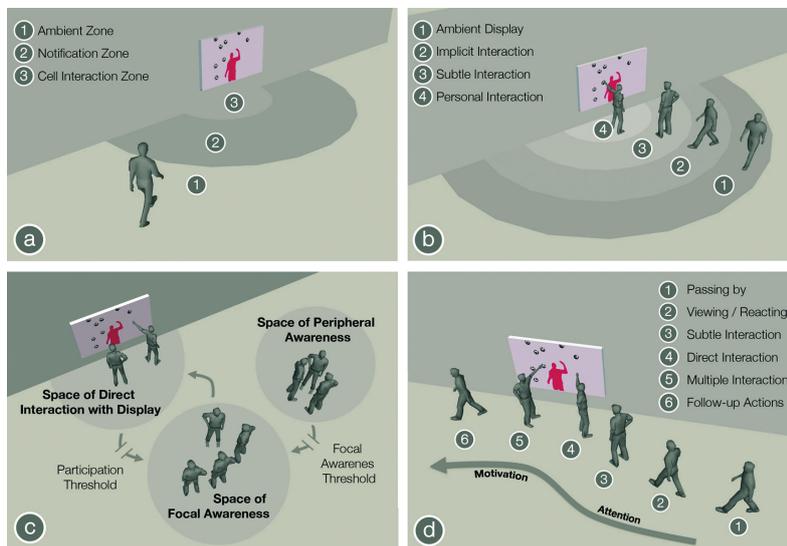


Figure 2. Public display interaction models [8]

The **three zones model** (a) [18] is a spatial zone model that differs between the ambient, notification and cell interaction zones. The **extended three zones model** (b) [20] advances this theory slightly by adding an additional zone. These models only consider information presentation and the interaction of a single user, but not transitions between zones, multiple users and the relationship between them. The **public interaction flow model** (c) [5] differs between spaces of awareness: peripheral awareness is the space where people are simply socializing; in the space of focal awareness, the attention shifts to the display; this is eventually followed by the direct interaction space. The **audience funnel model** (d) [14] investigates interaction phases and the probability of transitions between them. It differs between passing by a screen, viewing or reacting

towards it, subtle, direct and multiple interaction, and follow-up actions.

Audience engagement

Concerning audience engagement, there are different phases as well. Success is eventually reached if users perform interactions with the screens. As a first step for that, **understanding attention** is essential. Public displays are not owned by users, but instead, the screens need to compete for the peoples' attention. The right amount of interaction needs to be found - especially in the context of pervasive computing, it can be quickly too much. In public interaction, the presentation of oneself, the control over one's own personal data, social behavior and personal space are meaningful factors. Use of stimuli or physical objects can be approaches to address the problem of displays being ignored and to advertise interactivity [15].

Furthermore, **managing attention** is crucial. Alt [1] mentions a few concepts from other research literature: *behavioral urgency* attracts the users' attention, e.g., if moving objects appear abruptly or luminance contrast changes take place, these stimuli require immediate action and therefore make the viewer use the screen more likely. The *honeypot effect* describes that a crowd of people interacting with a screen attracts even more people [16]. The phenomenon of *change blindness* means that people have difficulty in observing obvious major changes of situation under certain circumstances; this fact can be used to minimize the distraction a display creates when content is updated [17].

Communicating interactivity is a big challenge, as people are usually used to public displays being solely utilized for advertisement purposes, and therefore they are not considering a possible interactivity component. Various techniques can be used to compensate for that [1]: a *call-to-action*, which can be a simple text label motivating the user to interact with the display; an *attract sequence*, i.e., constantly moving content; *analog signage* in combination with digital; *user representation*, i.e., showing the user itself in any way on the screen; or *immediate usability*, which means that previously mentioned techniques are combined to show that immediate, quick interaction with the screen content is possible [17].

People often do not use public displays on purpose, but rather spontaneously. To motivate **further engagement**, researchers and developers especially need to think about *why*, and not only *how* a system is used, as there is a difference between goal-oriented usage of displays compared to simple self-fulfilling usage. Motivators for the engagement of the audi-

ence are challenge and control, curiosity and exploration, choice, fantasy and metaphor, collaboration and influence [1].

A suitable location of the public display is essential for the audience's engagement as well. Cities, offices, universities, conferences, cultural sites or shops are locations that have been considered in literature about public displays [4]. Most studies, however, test their solution primarily only in a lab environment.

4.2 Interaction techniques

Features of public displays that are only made possible due to sophisticated interaction technologies equipped in the screen include navigation, content take-away (e.g., on mobile devices or to an e-mail address), or content upload. Three ways of interacting with a display are introduced in the following: through touch, mid-air gestures, or with the aid of mobile devices [8].

Touch screens are nowadays popular especially in mobile and handheld devices. One issue with touch at public displays is that users need to be able to reach the screen at all times, while simultaneously the displays generally need to be protected, visible and easily maintainable; these challenges are naturally bigger than at smaller mobile screens. In addition, users often do not expect interactivity, and due to multi-touch being used increasingly in modern devices, expectations towards devices in use are rather high, although these technologies are harder to realize and implement at large screens. Possible applications include information about cities and local travel, for instance, close to train or bus stations [3].

Mid-air gestures are another way of interacting with public displays. Especially in conditions where touch is not suitable, gesture-based interaction can be an advantage. However, selections and text entries are difficult. Gestures are used to manipulate objects shown on the screen or execute certain sequences of commands. It is necessary that users are familiar with the used gestures, and that these are both coherent, easy to recognize - which is difficult due to the large display size - and teach - which is challenging as not all gestures might be intuitive. Usually, prior knowledge is needed to understand them [21].

Mobile devices, which have become a tool used in daily life worldwide, simplify the interaction with public displays. As a possible scenario, technologies such as Bluetooth, Wi-Fi and GPS, which are used for exact positioning and location tracking, is utilized so that the users can select

the closest display to show their content there. This ensures *personalization*, as the display then reflects the interests of the user. As a result, a complete application that lets the user control screens according to their wishes is built; however, lack of privacy can be an issue here [9]. The *interaction* can take place through different ways: the mobile device can act as a remote control and forward the device input mechanisms to the display, or the display is used as a co-display. The phone then triggers a virtual machine with more computational power, which mirrors the phone content by sharing it wirelessly to the bigger screen. *Information take-away* is supported through QR codes or RFID tags, so that the user can collect information straight from the display to his device [17].

4.3 System software

Public displays are evolving to complex systems with many components, which need to be understood by a wide range of involved stakeholders, including display owners and manufacturers, IT professionals, advertisers, researchers and users. Signage software has various functional requirements, due to its wide range of features, including content distribution and scheduling, real-time play out of content, control of specialized display hardware, analytics, and user interaction [8].

Another important aspect is the reliability and maintainability of the displays - consequently, remote monitoring and management is a requirement. Failures are quickly visible and can have serious side effects; it is therefore necessary to have components which are operating reliably and embedded into physical displays. Furthermore, software systems need to be situation-dependent and adapt to environment and audience to deliver more personalized and relevant content. The ability to replicate content on various screens is another desired feature [11]. However, the available budget for development efforts is a limiting factor.

Taniguchi [19] gives more insight about different trends in digital signage with a focus on scheduling: **networked digital signage** describes large open networks of public displays that allow the content providers to submit, schedule, deliver and display contents according to their needs and wishes. **Context-aware digital signage** can display content according to the current status and adapt to changes of people, places and objects around the display. Approaches for scheduling include constraint-based, recommendation-based, advertisement optimization, display-owner oriented and hybrid.

4.4 Evaluation of pervasive displays

Another aspect of research is how to evaluate public displays. The authors of [3] compare various existing studies with regard to their research questions and approaches and used methods and tools.

Evaluating public displays is challenging, as there are no overall valid guidelines for their design, and many different objectives need to be fulfilled, such as increasing attention, optimizing interaction times, or finding the best interaction techniques. Studies have evaluated several **research areas**: audience behavior, as described above; user experience, since higher user experience likely leads to larger motivation to use a public display application; user acceptance, including incentives and motives to interact with a display; user performance, including error rates and task completion times; display effectiveness, i.e., counting the number of people passing by a display, looking at it or interacting with it; privacy; social impact, i.e., the engagement of people with displays within social interactions [3].

Study types in the context of public displays include *descriptive studies*, which evaluate what is happening in a certain situation and are good in the early stage development, whereas *experimental studies* feature statistical significance tests and are made to proof hypotheses, either in a lab or a real world setting. **Study paradigms** are either aimed to inform about the design of a prototype, or to evaluate an existing prototype. This can be done via observing social settings or asking users via methods and tools such as interviews, questionnaires or in groups. *Lab studies* aim to evaluate a system within a controlled environment, while *field studies* take place in a public setting, are shorter and focused towards a single research question. *Deployment-based research* introduces technology into a social setting, and after feedback and involvement is obtained, the deployment is iteratively improved [3].

The authors of [3] give some **guidelines** for the design of public displays: they claim that due to the huge impact of the content, it is essential to test various forms. Furthermore, understanding the users is highly important, as public displays naturally have different users at different times. Additionally, common problems, such as the lack of user attention or knowledge of display interactivity, need to be evaluated at an early stage.

5 Future developments

While applications of today often only replace or improve traditional analog signs and static systems, they are likely to have completely new functionality in the future. The new use cases, which are exemplary explained in the following, are sensitive to the environment and its viewers, and shrink the boundaries between entertainment, communication and signage that just fulfills the purpose of showing information.

5.1 Emergency services

The authors of [10] create an ideated scenario, where a child gets lost inside a shopping mall and is subsequently looked after with the aid of public displays showing a picture of the child, and continuously spreading it to further areas. The main issue in applying this scenario to the reality is that systems at different places are not connected to each other. However, public displays have a high potential to efficiently reach a large number of people, in contrast to alert systems that use warnings over TV or radio, as these are rather ineffective and only use one static, non-movable source for information delivery. The disadvantage of such solutions is though that the reliance on technology becomes fairly high, and that it is difficult to completely secure such systems against malicious threats and attacks.

5.2 Influencing behavior

The issues of traditional advertisement campaigns are that in paper format, they can only deliver static content, or in TV ads, they are unidirectional - it is therefore difficult to analyze how well some content fits a particular audience. On the contrary, interactive displays can show context-sensitive information and only present it when it is needed. This makes targeted campaigns with topics such as health, well-being and politics possible [10]. Due to location tracking taking place in large scale, privacy is again an issue that needs to be approached [9].

5.3 Personalization and self-expression

With devices and services evolving from pure information sources to personal assistants, the delivery of personalized information becomes more important in the context of public displays as well. The viewers can use their own handheld device first to issue a query to a system, and then the

screen as a supporting device to view the corresponding results. Use cases include, amongst others, personalized travel recommendations, restaurant menus, or the display of interesting upcoming events [10]. In addition, new ways of associating identity, personalization, and membership to certain groups or clubs are possible, for instance by developing interactive games [7] that use new, innovative techniques to recruit members for clubs or to get to know people around.

5.4 Substituting mobile devices

Small screens limit the usefulness of mobile devices - large, detailed content can not be displayed properly. Therefore, public displays can be used instead and connect with the mobile devices via internet. One potential usage scenario is if a doctor urgently needs to make a medical decision based on large ultrasound images, but is currently not in his office; in this case, he can use a nearby situated public display to view the images and see more details. Using appropriate measures, the sensitive private data of the patient should only be displayed on the smaller, mobile screen [23].

6 Conclusion

Advantageous characteristics of digital signage are that information can be easily distributed and pushed towards the user, content can be truly context-dependent, as public displays are embedded and used in the real world, content is interactive and multimedial, and rather easy and quick to update decentrally. Digital videos or other media open new possibilities for both users and advertisers. Displayed information is dynamic and can be updated frequently. Public displays and signage enhance public spaces, and ongoing research is covering several dimensions, ranging from technologies to social sciences.

However, it requires a lot of effort already at the requirements engineering and initial design stages to promote the possibility of engaging with the displays and encourage the users to interact with the content, especially as a lot of the content is usually not specifically personalized for the current user, thus making it less relevant. Furthermore, it is challenging to track user actions and measure how many people perceive and interact with the displays, while still ensuring privacy. Nevertheless, public displays play an essential role in the future infrastructure of public areas.

References

- [1] Florian Alt. *A Design Space for Pervasive Advertising on Public Displays*. University of Stuttgart, 2013.
- [2] Florian Alt, Jörg Müller, and Albrecht Schmidt. Advertising on public display networks. In *Computer, IEEE*, 45(5), pages 50–56, 2012.
- [3] Florian Alt, Stefan Schneegaß, Albrecht Schmidt, Jörg Müller, and Nemanja Memarovic. How to evaluate public displays. In *PerDis 2012 - Proceedings of the 2012 International Symposium on Pervasive Displays*, pages 1–6, 2012.
- [4] Carmelo Ardito, Paolo Buono, Maria Francesca Costabile, and Giuseppe Desolda. Interaction with large displays: A survey. In *ACM Computing Surveys*, Vol. 47, No. 3, Article 46, pages 1–38, 2015.
- [5] Harry Brignull and Yvonne Rogers. Enticing people to interact with large public displays in public spaces. In *Proceedings of the IFIP International Conference on Human-Computer Interaction*, pages 17–24, 2003.
- [6] Jorgos Coenen, Niels Wouters, and Andrew Vande Moere. Synchronized wayfinding on multiple consecutively situated public displays. In *PerDis 2016 - Proceedings of the 5th ACM International Symposium on Pervasive Displays*, pages 182–196, 2016.
- [7] Travis Cox, Marcus Carter, and Eduardo Velloso. Public display: Social games on interactive public screens. In *OzCHI '16 Proceedings of the 28th Australian Conference on Computer-Human Interaction*, pages 371–380, 2016.
- [8] Nigel Davies, Sarah Clinch, and Florian Alt. Pervasive displays: Understanding the future of digital signage. In *Synthesis Lectures on Mobile and Pervasive Computing*, pages 1–130, 2014.
- [9] Nigel Davies, Marc Langheinrich, Sarah Clinch, Ivan Elhart, Adrian Friday, Thomas Kubitzka, and Bholanathsingh Surajbali. Personalisation and privacy in future pervasive display networks. In *CHI '14 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2357–2366, 2014.
- [10] Nigel Davies, Marc Langheinrich, Rui Jose, and Albrecht Schmidt. Open display networks: A communications medium for the 21st century. In *Computer, IEEE*, 45(5), pages 58–64, 2012.
- [11] Maria Montoya Freire, Venkata Praneeth Tatiraju, Mohit Sethi, and Mario Di Francesco. Replication of web-based pervasive display applications. In *PerDis 2016 - Proceedings of the 5th ACM International Symposium on Pervasive Displays*, pages 1–38, 2016.
- [12] Christian Kray, Gerd Kortuem, and Antonio Krüger. Adaptive navigation support with public displays. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 326–328, 2005.
- [13] BroadSign International LLC. Cloud-based digital signage software solutions, 2019.

- [14] Daniel Michelis and Jörg Müller. The audience funnel: Observations of gesture based interaction with multiple large displays in a city center. In *International Journal of Human-Computer Interaction*, 27(6), pages 562–579, 2011.
- [15] Jörg Müller, Florian Alt, Daniel Michelis, and Albrecht Schmidt. Requirements and design space for interactive public displays. In *Proceedings of the 18th ACM international conference on Multimedia (MM '10)*, pages 1285–1294, 2010.
- [16] Jörg Müller, Robert Walter, Gilles Bailly, Michael Nischt, and Florian Alt. Looking glass: A field study on noticing interactivity of a shop window. In *CHI '12 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 297–306, 2012.
- [17] Peter Peltonen, Esko Kurvinen, Antti Salovaara, Giulio Jacucci, Tommi Ilmonen, John Evans, Antti Oulasvirta, and Petri Saarikko. It's mine, don't touch!: interactions at a large multi-touch display in a city centre. In *CHI '08: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1285–1294, 2008.
- [18] Norbert Streitz, Carsten Röcker, Thorsten Prante, Richard Stenzel, and Daniel van Alphen. Situated interaction with ambient information: Facilitating awareness and communication in ubiquitous work environments. In *Proceedings of the 10th International Conference on Human-Computer Interaction*, pages 1–5, 2003.
- [19] Yukinobu Taniguchi. Content scheduling and adaptation for networked and context-aware digital signage: A literature survey. In *ITE Transactions on Media Technology and Applications 2018*, 6(1), pages 18–29, 2018.
- [20] Daniel Vogel and Ravin Balakrishnan. Interactive public ambient displays: Transitioning from implicit to explicit, public to personal, interaction with multiple users. In *Proceedings of the 17th Annual ACM symposium on User Interface Software and Technology*, pages 137–146, 2004.
- [21] Robert Walter, Gilles Bailly, and Jörg Müller. Strikeapose: Revealing mid-air gestures on public displays. In *Proceedings of the 2013 ACM Conference on Human Factors in Computing Systems*, pages 841–850, 2013.
- [22] Miaosen Wang, Sebastian Boring, and Saul Greenberg. Proxemic peddler: A public advertising display that captures and preserves the attention of a passerby. In *PerDis 2012 - Proceedings of the 2012 International Symposium on Pervasive Displays*, pages 1–7, 2012.
- [23] Adam Wolbach, Jan Harkes, Srinivas Chellappa, and M. Satyanarayanan. Transient customization of mobile computing infrastructure. In *Proceedings of the First Workshop on Virtualization in Mobile Computing*, pages 37–41, 2008.

Neural network training with Adam, Nesterov and Standard Gradient Descent

Lasse Kärkkäinen

lasse.karkkainen@aalto.fi

Tutor: Alex Jung

Abstract

Feed-forward multilayer networks are examined from bottom up, starting with gradient derivation, alongside with Python-Numpy implementation based on plain linear algebra. Standard gradient descent, Adam optimizer and Nesterov method are implemented and compared in practice.

KEYWORDS: deep neural net, gradient descent, Nesterov accelerated gradient, Adam optimizer, Numpy

1 Introduction

In the recent years we have seen a boom in AI research and all the exciting new things that make headlines appear to be based on neural networks. As such, neural networks should be a basic tool in every data scientist's toolbox. It turns out that such models are in fact quite simple to implement as a neural classifier is implemented from scratch, using only basic linear algebra and the Python programming language.

A neural net designer is faced with a choice of learning algorithm, typically taken to be either Adam optimizer or simple gradient descent, and hyper parameters such as step size and update function. Simple trial

and error becomes tedious, and finding optimal parameters analytically is hard for multi-layer networks and convex problems.

In this work, a neural network is implemented from scratch, using surprisingly little code, and shown to operate on real problems. In particular, three different optimizers are implemented and compared in practical problems. The work is intended to be educational and suitable for anyone with strong prior knowledge on Python and linear algebra, but necessarily no experience with neural networks.

The reader is encouraged to focus more on either the mathematical background in section 4, or on the practical implementation in section 5, depending on personal preferences. In particular, the algebraic approach may feel heavy for those less mathematically inclined.

This report builds on top of and uses parts of my earlier work *Deep Neural Softmax Classifier in Numpy* [5].

2 Literature

Linear systems can be solved with ease but the rectifier in-linearity of neural networks generally make them impossible to solve for algebraically, and thus iterative methods such as gradient descent are employed. Generally it is recommended that the parameters such as learning rate and update methods are chosen by trial and error, as proving optimality by analytical methods is hard. It is typically recommended to choose initial weights and biases as random numbers close to zero with the assumption that this signals initial lack of confidence and will lead to fast convergence, although also other approaches exist. Learning rate is often chosen as a small constant or decreased over training epochs, although adaptive methods are also possible. [1][7]

For some convex optimization problems it is possible to analytically solve for update parameters that provide optimal learning rate. Standard gradient descent, Nesterov momentum and Heavy-ball method are investigated in [6] where authors find more optimal parameters than those popularly used. However, the authors note that their results remain limited to integral quadratic constraints.

In contrast to the textbook advice of near-zero initialization of weights, more recently it has been found that deep networks require more thought to initialization and optimizers, and better approaches have been proposed.[3][8]

Adam adaptive moment optimization [4] has gained widespread use

since it was proposed in 2015, and further work has looked into integrating it with Nesterov algorithm [2], already known for its good performance.

3 Methods

A multilayer feed-forward network is implemented in Python-Numpy, using basic linear algebra to implement standard gradient descent, Nesterov method and Adam optimizers. The modules in listing 1 are assumed to be loaded.

```
1 from math import pi
2 from numpy import array, linspace, ones, random, sin, tanh
```

Listing 1. Symbols used in code fragments. Please refer to Python and Numpy documentation for details.

4 Algebraic basis of neural networks

4.1 Artificial neurons

A **perceptron** (an artificial neuron) is the basic building block of neural nets. It is nothing more than a linear transform of inputs into a linear output, that is optionally further transformed by a rectifying function to produce a single output for each input sample.

A **neural layer** bundles multiple perceptrons together for multiple outputs. Many samples $\mathbf{X} \in R^{N \times d_{in}}$ are typically processed together, for a transformed output $\mathbf{Y} \in R^{N \times d_{out}}$, so the transform becomes $y_{tj} = f(\sum_i^{d_{in}} x_{ti}w_{ij} + b_j)$, or in matrix form $\mathbf{Y} = f(\mathbf{XW} + \vec{b}^T)$.

This is called a *fully connected* layer because each perceptron is connected to all input dimensions.

Neuronal weights $\mathbf{W} \in R^{d_{in} \times d_{out}}$ and biases $\vec{b} \in R^{d_{out}}$ are found in model training (usually by *gradient descent*). The choice of rectifier f depends on application.

4.2 Rectifiers

Rectifiers serve two important purposes. On the model output they may be used to limit the range of output, e.g. to 0..1 for probabilities. More importantly, they are the only non-linear component in a neural network,

and since a combination of linear transforms reduces to a single linear transform, deep networks would be of little use. The use of rectifiers, also called **activation functions**, allows building rich estimators by adding more layers. The normal choices are shown in Figure 1.

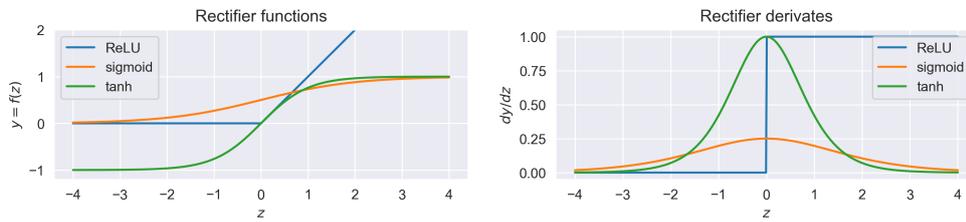


Figure 1. The inlinearities and ranges of the popular rectifier functions are apparent. For purposes of gradient descent, the derivatives shown on the right are crucial.

Because ReLU's derivate is zero for any negative input, gradient descent may be unable to progress. For this purpose, Leaky ReLU is sometimes used in its place, allowing a fraction of the negativity to "leak" through.

Additionally, softmax rectifier is used on the final layer in multi-class classification where classes are exclusive and exhaustive. Unlike the scalar rectifiers, softmax works on vectors to normalize the output so that all classes sum to one.

4.3 Gradient derivation

A function's gradient $\partial f / \partial \mathbf{Z}$ indicates, for $f(\mathbf{Z})$, the direction from \mathbf{Z} where the function's value increases fastest, and the gradient's length indicates the steepness of change. Losses are to be minimized, which is accomplished by following gradients of the *negated loss function*. The following derivations are written in scalar form with true values r_{tk} , predictions y_{tk} (network output) and rectifier inputs z_{tk} .

A **linear regressor** with *Mean-Squared-Error loss* serves as a warm-up to deriving gradients:

$$L = \frac{1}{NK} \sum_t \sum_k (y_{tk} - r_{tk})^2 \quad (1)$$

$$\frac{\partial L}{\partial y_{tk}} = s(y_{tk} - r_{tk}) \quad (2)$$

$$s = \frac{2}{NK} \quad (3)$$

The **logistic regressor** is a classifier that uses sigmoid with *cross-entropy loss function*:

$$L = -\frac{1}{NK} \sum_t^N \sum_k^K (r_{tk} \log_2 y_{tk} + (1 - r_{tk}) \log_2(1 - y_{tk})) \quad (4)$$

$$\frac{\partial L}{\partial z_{tk}} = \frac{\partial L}{\partial y_{tk}} \frac{\partial y_{tk}}{\partial z_{tk}} \quad (5)$$

$$= -s \left(\frac{r_{tk}}{y_{tk}} - \frac{1 - r_{tk}}{1 - y_{tk}} \right) y_{tk} (1 - y_{tk}) \quad (6)$$

$$= s (y_{tk} - r_{tk}) \quad (7)$$

$$s = \frac{1}{NK \ln 2} \quad (8)$$

Gradient for the **softmax classifier** is derived from cross-entropy loss by plugging in the softmax function:

$$L = -\frac{1}{N} \sum_t^N \sum_k^K r_{tk} \log_2 y_{tk} \quad (9)$$

$$= -\frac{1}{N} \sum_t^N \sum_i^K r_{ti} \log_2 \frac{\exp z_{ti}}{\sum_j^K \exp z_{tj}} \quad (10)$$

$$= -s \sum_t^N \sum_i^K r_{ti} \left(z_{ti} - \ln \sum_j^K \exp z_{tj} \right) \quad (11)$$

$$s = \frac{1}{N \ln 2} \quad (12)$$

$$\frac{\partial L}{\partial z_{tk}} = -s \left(r_{tk} - \sum_i^K r_{ti} \frac{\exp z_{tk}}{\sum_j^K \exp z_{tj}} \right) \quad (13)$$

$$= s (y_{tk} - r_{tk}) \quad (14)$$

For exclusive and exhaustive classes $\sum_i^K r_{ti} = 1$. The K classes' dependency on each other (because of softmax normalization) is hidden and the sum terms disappear when the gradient is written in terms of y_{tk} .

All three cases reduce into a very simple form where only the scaling factor s is different. In standard gradient descent and in Nesterov method the step size has equivalent effect to such scaling of gradients. Adam optimizer is completely insensitive to gradient scaling (assuming $s > 0$). For the rest of the text, the scaling factor is ignored by redefining each loss function so that it is divided by its respective scaling factor. Then for each of these three cases

$$\frac{\partial L}{\partial z} = y_{tk} - r_{tk}. \quad (15)$$

Noting that $\mathbf{Z} = \mathbf{XW} + \vec{b}^T$, we find the following update equations for a single layer:

$$g_{b_j} = \sum_t^N g_{tj} \quad (16)$$

$$g_{w_{ij}} = \sum_t^N x_{ti} g_{tj} = \mathbf{X}^T \mathbf{G} \quad (17)$$

$$g_{ti}^X = \sum_j^K g_{tj} w_{ij} = \mathbf{G} \mathbf{W}^T \quad (18)$$

$\mathbf{G}^X = g_{ti}^X$ are the gradients respect to \mathbf{X} . If there are multiple layers, these gradients are passed down to the prior layer as its g_{ij}^Y . The prior layer calculates the new gradients before performing the updates above by applying the chain rule, e.g. for tanh: $g_{tj} = g_{ij}^Y (1 - y_{tj}^2)$. For the logistic and softmax classifiers' output layer this was already handled while deriving the loss gradient, and in the case of linear regression there is no output rectifier, i.e. $\mathbf{Y} = \mathbf{Z}$, so no adjustment is necessary.

Once gradients have been calculated in backpropagation, the optimizers are not concerned with network structure. All parameters are handled as individual scalars, and as such it is more practical to collect all parameters (weights and biases of all layers) into a single vector, collectively marked $\vec{\theta}$, and parameters' corresponding gradients (g_w and g_b calculated above) into identically sized \vec{g} . Such notation will be followed in all that follows.

4.4 Standard gradient descent

Given model's loss function gradient $\vec{g}_t = \nabla L(\vec{\theta}_t | \mathbf{X}, \mathbf{Y})$ against each parameter θ at time step $t = 1 \dots N$, with training data \mathbf{X} and labels \mathbf{Y} , updates are performed using a fixed step α as

$$\vec{\theta}_{t+1} = \vec{\theta}_t - \alpha \vec{g}_t \quad (19)$$

Alpha is taken to be a small positive number such that the parameters gradually migrate downhill in the loss function, leading to a local optimum. Due to the way of updates, data points on steep parts of the loss function, that have large gradients, affect updates more, while points on plateaus have very little effect. Thus, the updates do not effectively consider points far from the decision boundary, even if misclassified by the model. Only those data points that are in the uncertainty region of the model have a significant effect.

Further, the distance to the global minimum is not considered, only the slope. Due to this, using large α the model will overshoot and may fail to converge, and using smaller values the model takes more time to converge than is necessary.

Various momentum methods have been proposed to address these limitations.

4.5 Nesterov accelerated gradient

If a number of successive updates all occur in the same direction, it would seem sensible to do larger updates to skip some iterations. On the other hand, if the target is overshoot, successive updates are likely to be in opposing directions, and an average should converge near the optimum in the middle.

Nesterov method introduces velocity \vec{v} of parameter changes. The velocities are updated as if the standard gradient updates were acceleration. A friction coefficient $\mu \in [0, 1]$ is added to decay existing velocity so that oscillation is reduced.

$$\vec{v}_{t+1} = \mu\vec{v}_t - \alpha\vec{g}_t \quad (20)$$

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \vec{v}_{t+1} + \mu(\vec{v}_{t+1} - \vec{v}_t) \quad (21)$$

Updates are still dependent on the gradient's scale just like in simple gradient descent, but acceleration to the goal speeds up convergence.

4.6 Adam accelerated method

The **Adaptive moment estimation** introduced in [4] is now the most popular method of neural network training.

All operations are performed element-wise for $\vec{\theta}$, \vec{g} , \vec{m} and \vec{v} , which are written in scalar form below without showing the vector indices.

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (22)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (23)$$

$$k_t = \frac{\sqrt{1 - \beta_2^t}}{1 - \beta_1^t} \quad (24)$$

$$\theta_t = \theta_{t-1} - \alpha k_t \frac{m_t}{\sqrt{v_t} + \epsilon} \quad (25)$$

In order to understand Adam, consider setting $\beta_1 = \beta_2 = \epsilon = 0$, so that the updates reduce to $\pm\alpha$ step towards the optimum:

$$\theta_t = \theta_{t-1} - \alpha \frac{g_t}{|g_t|} \quad (26)$$

This is also what occurs on the very first step even if the beta coefficients are above zero due to the *un-biasing factor* k_t (Eq. 24). Un-biasing is used

to allow faster initial convergence because the moments are initialized with zero.

Applying slightly-below one beta values ($\beta_1 = 0.9$ and $\beta_2 = 0.999$ are recommended by the paper) causes averaging of g and g^2 over time. In effect, the updates are *moving mean* normalized by *slowly moving stddev from zero*. The factor k_t starts at 0.32 and reaches 0.99 in about 4000 steps, using recommended beta values.

ϵ is chosen as a very small positive value to avoid division by zero, although larger values could be useful to avoid very large leaps if the average momentum m_t becomes much greater in magnitude than the squared average v_t .

Due to adjusting each parameter separately of the gradient magnitude, the steps do not strictly follow the gradient's direction. Still incremental updates of each parameter towards the minimum indicated by the corresponding partial derivative leads to optimal solution. The averaging gradually decreases the step length if overshooting occurs.

5 Practical example

All neural networks are functions that convert input data into output data, using a series of linear transforms intertwined with non-linear rectifiers. A network with one input variable and one output variable can be taught to approximate any real-valued function, as shown in figure 2. Listing 2 shows the corresponding code.

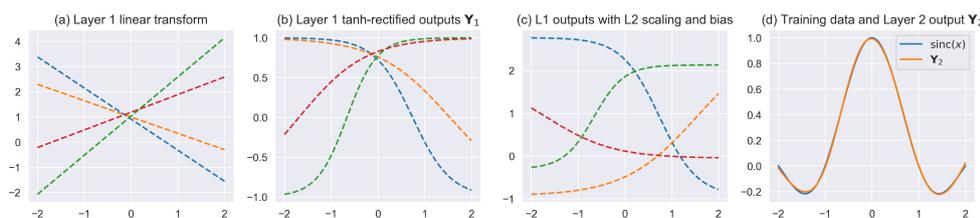


Figure 2. The forward pass of a network taught to estimate the sinc function. The x axis in each plot shows the network input, transformed to y values in various stages. (a) four curves are created out of one input because the weights and biases have four columns. (b) applies tanh to produce layer 1 outputs, which are fed to layer 2. (c) shows layer 2 weights and biases applied, although in practice the matrix multiplication instantly sums the four hidden dimensions into one output, shown in (d) where the network output Y_2 closely matches the training data. The way that the hidden dimensions (hyphenated curves) are used depends on the initial parameter values as well as the training algorithm, so different assignments are found on different runs. In this instance, the curves displayed in (c) are clearly asymmetric despite having a symmetric target function.

```

1 W1, b1, W2, b2 = parameters
2 # Forward pass
3 Y1 = tanh(X @ W1 + b1)
4 Y2 = Y1 @ W2 + b2

```

Listing 2. A two-layer network. In Numpy, operator @ denotes dot product. W_1 converts the $N \times d$ input data X into an $N \times d_1$ matrix. The b_1 vector is added to each row of the matrix. The tanh function operates element-wise, producing layer 1 output Y_1 . The second weight multiplication converts the data to $N \times d_2$. The network output Y_2 is obtained after adding the second bias to every row.

The purpose of network training is to find values for the parameters (figure 3), such that the forward pass produces useful output.

The training data is typically a set of data points as matrix X where each row represents a sample, and matrix Y where the corresponding row shows the desired network output for that sample. In this example, 200 points of the sinc function are calculated (listing 3).

```

1
2 X = linspace(-2, 2, N).reshape(N, 1) # Nx1 input
3 Y = sin(pi * X) / (pi * X)         # Nx1 output

```

Listing 3. Training data where $Y = \text{sinc}(X)$ for X values between -2 and 2.

The optimization begins with random weights and biases set to all ones (listing 4). Random initialization is necessary for networks of two or more layers, so that hidden variables assume different values, and start estimating separate things while the network is being optimized.

```

1 # Dimensions: input -> hidden -> output
2 d, d1, d2 = X.shape[1], 4, Y.shape[1]
3 # Initial parameters
4 parameters = array([
5     random.normal(size=(d, d1)), ones(d1), # W1, b1
6     random.normal(size=(d1, d2)), ones(d2), # W2, b2
7 ])

```

Listing 4. Parameter initialization. Weights are unit normal, biases all ones. The network dimensions are determined by the shapes of the weight matrices, chosen to match the input, any hidden dimensions and finally the output dimension. In each weight matrix, the number of rows must match the layer input, and the number of columns the layer output dimensionality.

$$\begin{array}{c}
 \left[\begin{array}{c}
 \left[\begin{array}{cccc}
 -1.235 & -0.649 & 1.553 & 0.701
 \end{array} \right] \\
 \left[\begin{array}{cccc}
 0.912 & 0.994 & 1.031 & 1.182
 \end{array} \right] \\
 \\
 \left[\begin{array}{c}
 1.857 \\
 -1.85 \\
 1.217 \\
 -0.965
 \end{array} \right] \\
 \left[\begin{array}{c}
 0.914
 \end{array} \right]
 \end{array} \right]
 \begin{array}{l}
 \mathbf{W1} \\
 \mathbf{b1} \\
 \\
 \mathbf{W2} \\
 \mathbf{b2}
 \end{array}
 \end{array}$$

Figure 3. Numpy array `parameters`, containing weight matrices and bias vectors. Values have been found using the example training code, and are visualized in figure 2.

```

1  alpha = 0.001                # Learning rate
2  for t in range(10000):       # Training epochs
3      W1, b1, W2, b2 = parameters
4      # Forward pass
5      Y1 = tanh(X @ W1 + b1)
6      Y2 = Y1 @ W2 + b2
7      # Backpropagation
8      G = Y2 - Y                # Squared error gradient
9      Gb2 = G.sum(axis=0)
10     GW2 = Y1.T @ G
11     G = G @ W2.T              # Backpropagate to layer 1
12     G *= 1 - Y1**2           # Tanh derivate
13     Gb1 = G.sum(axis=0)
14     GW1 = X.T @ G
15     gradients = array([GW1, Gb1, GW2, Gb2])
16     # Perform updates (standard gradient descent)
17     parameters -= alpha * gradients

```

Listing 5. Training a two-layer network by standard gradient descent.

Listing 5 shows the full training code. The code used in this example places no restriction on the number of dimensions, so it can be used with high-dimensional data and more than one output. A large number of training epochs are performed, and in each epoch all data points are first taken through the forward pass. Then the loss function gradient is calculated using the difference between network output and desired output (line 8). Parameters live in a data structure (figure 3) for which Numpy can perform element-wise arithmetic. Gradients against each parameter, calculated on lines 9-10 and 13-14, are stored in an identically shaped structure on line 15. This allows the simple update equation on line 17, where all parameters are updated by standard gradient descent.

Standard gradient descent simply moves all parameters slightly to the opposite direction of the gradient, in an attempt to reduce loss. It is worth

noting that the loss function itself need not be evaluated during optimization.

Finally, only a small modification is required to use another optimizer. Adam optimizer is shown in listing 6 and Nesterov momentum method in listing 7.

```
1 alpha = 0.001 # Learning rate
2 beta1, beta2, epsilon = 0.9, 0.999, 1e-8 # Adam settings
3 m = v = 0 # Adam 1st and 2nd
  moments
4 for t in range(1, 10000): # Numbering starts at 1
  (forward pass and backpropagation code omitted)
5 # Perform updates (Adam optimizer)
6 m = beta1 * m + (1 - beta1) * gradients
7 v = beta2 * v + (1 - beta2) * gradients**2
8 unbias = (1 - beta2**t)**.5 / (1 - beta1**t)
9 parameters -= alpha * unbias * m / (v**.5 + epsilon)
```

Listing 6. Gradient descent replaced by Adam optimizer. Forward pass and backpropagation remain the same as in listing 5. The initially zeroed moments m and v take on update the same shape as parameters and gradients.

```
1 alpha = 0.001 # Learning rate
2 mu = 0.9 # Nesterov coefficient
3 v = 0 # Velocities
4 for t in range(10000): # Training epochs
  (forward pass and backpropagation code omitted)
5 # Perform updates (Nesterov momentum)
6 v_prev = v
7 v = mu * v - alpha * gradients # GD + momentum
8 parameters += v + mu * (v - v_prev) # Nesterov acceleration
```

Listing 7. Nesterov momentum method. Forward pass and backpropagation remain the same as in listing 5. The initially zeroed v takes on update the same shape as parameters and gradients.

5.1 Classification of 2d points

While working in one dimension is a useful simplification, it quickly becomes apparent that this isn't the strongest area for neural networks. Before jumping into higher dimensions, it is useful to evaluate the code just written with a few more dimensions. Also, instead of regression, where the outputs are real-valued linear output of the last layer, a softmax rectifier is attached, turning the network into a classifier where multiple outputs signify the probabilities of the sample belonging to each of the classes. In the case of softmax, contrary to sigmoid, the probabilities are forced to sum to one, i.e. the classes are exclusive and exhaustive.

A generated set of two-dimensional points where model operation can be visually inspected as a development of the classification "heat maps" as seen in figures 4, 6 and 8 where the three optimizers may be visually compared, although it needs to be noted that there is a lot of variance from one run to another. Visually it would seem that Adam is a clear winner on this problem where the earlier example code is extended from two to eight layers of feed forward network. The training statics for Adam, Nesterov and SGD (figures 5, 7 and 9, respectively) tell a similar story.

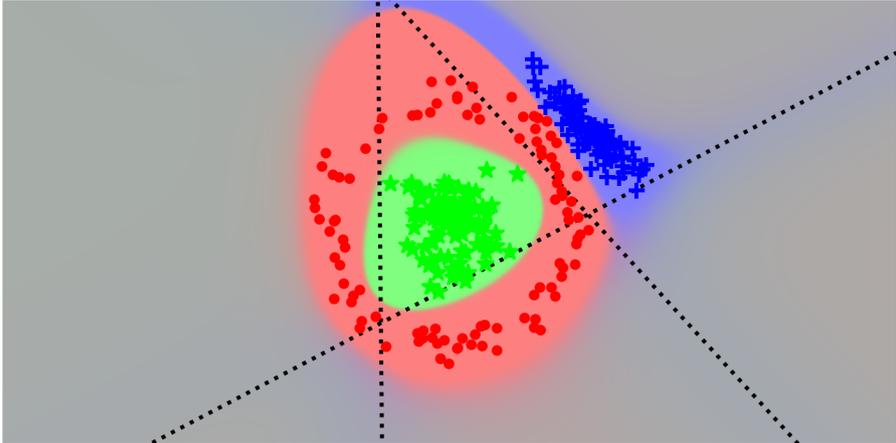


Figure 4. A colored heatmap shows classes predicted at each coordinate (x, y) , using Adam optimizer with 300 randomly generated points $x, y \rightarrow red|green|blue$, each class having a distinct distribution. *Prior injection* used in training makes the surrounding areas appear grey, to signify equal class probabilities. Dotted lines show the first layer decision boundaries but classification only loosely follows those because there are total of 8 layers.

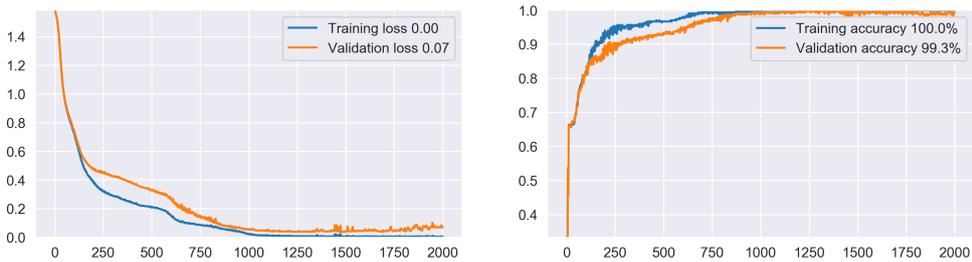


Figure 5. Decrease of error over training epochs, measured for the training points shown in figure 4 and also for an identically generated validation set. The graph shows no difference between training and validation. Loss signifies the amount of information not encoded by the prediction. In three-class classification $\log_2(3) \approx 1.58$ bits equals blind guessing, and that is also the limit imposed on the y axis.

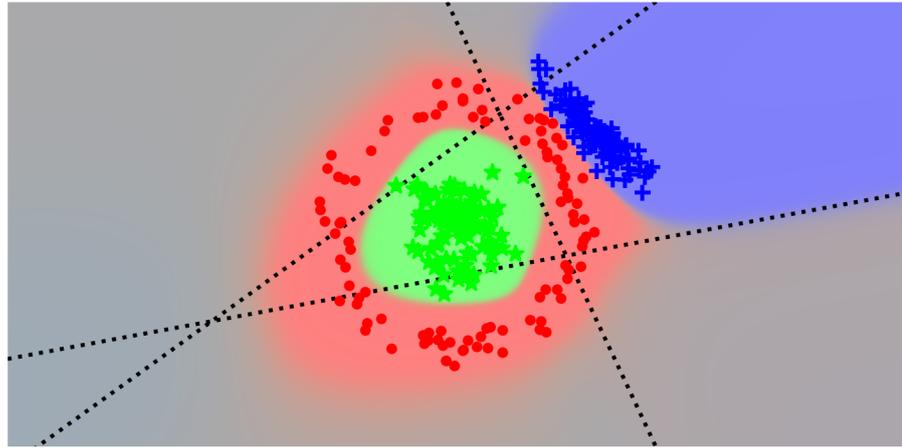


Figure 6. Nesterov converges with similar results.

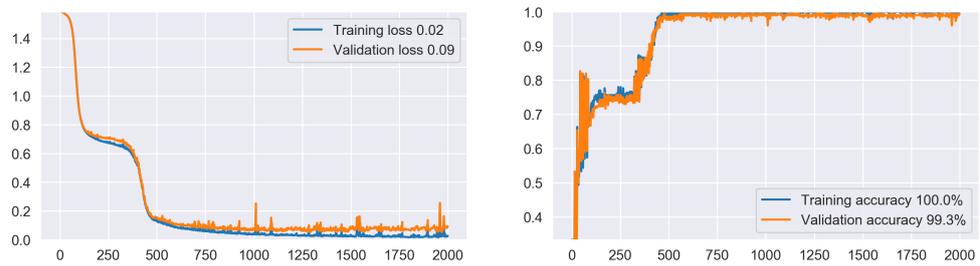


Figure 7. Nesterov's learning rate had to be decreased by a magnitude compared to Adam, to avoid spurious spikes. Despite that, Nesterov converges quickly.

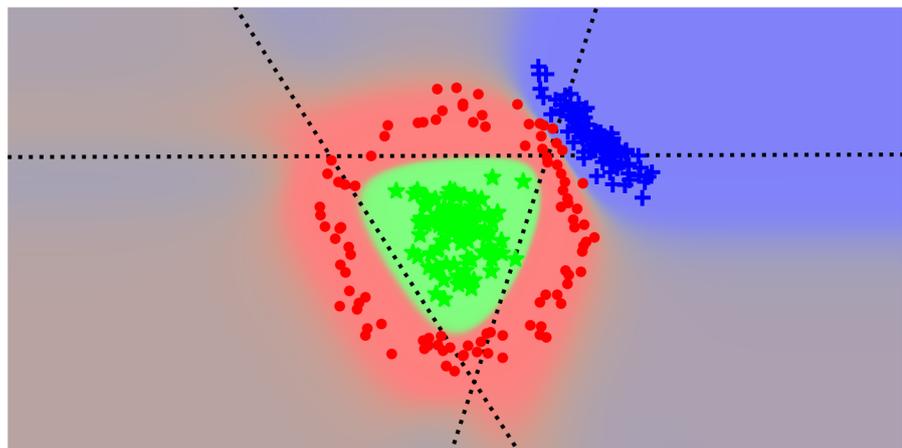


Figure 8. Plain gradient descent could do as well as the others but it would take more time.

6 Analysis

In the testing done, Nesterov and Adam have consistently converged much faster and more reliably than SGD. For some of the problems tested, in

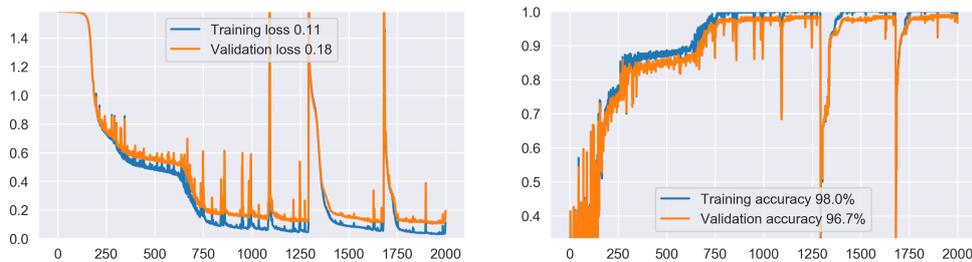


Figure 9. Stochastic gradient descent with same alpha as Adam (0.001) shows spurious spikes and slow convergence.

particular a 1-layer network with sigmoid output and MSE loss, Nesterov appears to converge much faster than Adam can, while on others (deep softmax classifiers) Adam has converged faster. In all problems tried, Adam seems to converge much more reliably than either SGD or Nesterov, when comparable parameters are used.

Quite possibly SGD and Nesterov break down due to large spurious adjustments that occur when a data point hits a very steep gradient, and after such mishap the network may not be unable to converge to any useful result. Adam seems more resistant to such mishaps, presumably because it in principle limits the length of the steps taken, which may also explain its slower convergence when it simply cannot reach the necessary parameter values in time, despite making all steps in the right direction.

The tests done and discussed here are rather preliminary. Due to many free model parameters, where not even the learning rates are directly comparable, no quantitative results of convergence can be presented. Obtaining such results would require brute force approach of parameter trials or other methods not possible within this work.

7 Discussion

The focus of the work shifted far towards the fundamentals in particular due to reviewer's comments, where the original content was found to be too advanced for the general audience of CS students. Although most of the actual work on neural networks nowadays is done on higher level libraries such as PyTorch and Keras, I find working directly with linear algebra useful in understanding why the "black boxes" in such frameworks behave the way they do, even if one doesn't poke the internals directly.

Although other authors have done comprehensive analysis of parameter initialization and optimizer settings for optimal convergence in var-

ious niches, and various methods of automatic hyper-parameter tuning are employed on the field, this work was limited to manual testing on a CPU-based system where each test run took considerable time. Thus, the original goal of comprehensive parameter testing was not attained.

Instead, full gradient derivation and the most popular optimizers are shown along with practical example code which programmers should find more friendly than the algebra. Further, the implemented network attains reasonable results in regression tasks with program code that can fit on a single page, which itself is fascinating.

Overall, the work turned more educational and introductory than a scientific paper, especially since no novel results are presented for vigorous review.

References

- [1] Ethem Alpaydin. *Introduction to Machine Learning*. Massachusetts Institute of Technology, 2nd edition, 2010.
- [2] Timothy Dozat. Incorporating nesterov momentum into adam. 2016.
- [3] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. 2010.
- [4] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. 2015.
- [5] Lasse Kärkkäinen. Deep neural softmax classifier in numpy. 2018.
- [6] Laurent Lessard, Benjamin Recht, and Andrew Packard. Analysis and design of optimization algorithms via integral quadratic constraints. 2015.
- [7] Stuart J. Russell and Peter Norvig. *Artificial Intelligence, A Modern Approach*. Pearson, 3rd edition, 2015.
- [8] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. 2013.

Pointer Analysis: Revisiting Prevalent Techniques with Current Developments, Challenges, and Uses

Gilang Mentari Hamidy
gilang.hamidy@aalto.fi

Tutor: Hans Liljestrand

Abstract

Pointer analysis has been discussed for more than two decades and is generally considered well-established in compiler technology. In particular, pointer analysis is commonly used to aid the software optimization process, as well as identify possible software defects that may lead to a security vulnerability. Despite its importance and significant role in computer programming, there is a limited discussion on this topic recently, along with limited literature covering pointer analysis as a general subject. This paper aims to cover and review the foundation of pointer analysis, by discussing the background, the established algorithms including Andersen's and Steensgaard's algorithms, comparison between the algorithms and the implementation and use-cases in the modern compiler.

KEYWORDS: pointer analysis, compiler, program analysis, programming language

1 Introduction

Programming languages incorporate the concept of reference which allows an object to be referred in other parts of the program without having to copy the entire object content. It enables flexibility inside a program to refer to a specific object in different program points. Some programming

languages typically implement reference as a pointer, which is essentially a memory address of the object location. Pointer is very flexible as you can manipulate it directly using arithmetic operators, and point it to any memory location. Despite their flexibility, pointers introduce complexity in program flow, as well as security issues which can impact the reliability of software. In order to mitigate these issues, a specialized program analysis is required.

Static pointer analysis is a program analysis that estimates the runtime value of a pointer variable [1]. It provides a safe approximation of the possible runtime value of a pointer, and can be useful for other program analysis purposes, such as optimization and defect analysis. Every programming language implementation that uses pointer to support, for instance, raw pointer or type-safe reference, can benefit from pointer analysis.

This paper provides background knowledge that justifies the importance of pointer analysis (section 2), discusses well known algorithms (section 3), explores challenges in improving pointer analysis performance and precision (section 4), reviews current implementations and use-cases in modern compilers (section 5), and finally, concludes with future work suggested by this review (section 6).

2 Background

The pointer is a representation of a memory location in a program and allows a program to store and manipulate a memory location within its logic. It can refer to a location of data or instructions, and it is an essential component in the machine instruction. This section will briefly cover about pointers in programming languages (subsection 2.1), and pointer analysis (subsection 2.2).

2.1 Representing Memory Address in Programming Language

A variable represents a value in a computer program. It allows the programmer to assign values and perform a computation. The programmer typically specifies the variable scope, such as global, local, or heap, not the storage location to store the variable. Instead, the compiler identifies and select the best location to hold a variable [1]. Thus, a variable does not always require a memory location, such as when all variables can be

fitted in available CPU registers. Nevertheless, a variable must be stored in memory in many cases, in which case the compiler assigns it a memory address automatically during code generation.

Several programming languages, such as C and C++, allow the programmer to obtain the memory address of a variable. C and C++ provide the *address-of operator* (&) which denotes an expression of obtaining a variable address, i.e., a *pointer* to the variable. In C++, a pointer declares information about the underlying object type that it refers to [17]. It helps the compiler to select a suitable operation for dereferencing or manipulating the pointer, as well as detecting incorrect use of pointer operations.

A pointer does not necessarily point to data. A *function pointer* allows the abstraction of a function address and enables programmer's control over program flow more flexibly [11]. Several frequent use-cases of a function pointer includes defining a callback function to be passed to another function and dynamically loading an external library at runtime.

The comprehensive type information expressed in program source code is almost entirely erased when the program is compiled to a machine language. The compiler transforms pointer operation and execution into a semantically equivalent direct or indirect addressing instruction [1]. The compiler performs memory allocation automatically and supplies the generated addresses and offset as instruction operands. No apparent semantic information about the type will be stored unless it is explicitly expressed in the program logic, for example, the Run Time Type Information (RTTI) feature in C++ [17].

2.2 Pointer Analysis

During the compilation stage, the compiler may need to predict the possible runtime value of a pointer. Static analysis can be used to analyze the program structure and identify the possible pointer values based on program semantics. This analysis is called *pointer analysis*. For every pointer in the program, a pointer analysis produces a *points-to set*, which is a set of possible object referred by the pointer at runtime according to the program semantics, which is typically a pointer assignment operation [1].

Pointer analysis and *alias analysis* (also known as a static analysis of pointers) are being used interchangeably. However, alias analysis seeks to answer a different question, whether different pointers may point to the same value [6]. Alias analysis is considered a subset of pointer analysis

as it does not produce a complete collection of a possible runtime value of the pointer, rather it only determines whether two pointers may refer to the same object. Alias analysis produces *must-alias* response when both pointers are guaranteed to point the same object, *no-alias* response when both pointers are guaranteed to not point to the same object, and *may-alias* response when the definite relationship cannot be concluded.

According to Rice's theorem, static code analysis, including pointer analysis, is considered an undecidable problem [10]. Pointer analysis involves large problem sets, consists of all pointers in the program, variations in the control flow, and data dependencies between instructions, leading to a decrease of analysis efficiency. Therefore, many pointer analysis algorithms provide an estimation rather than the exact result. They utilize statistical and probabilistic algorithms to approximate runtime values of pointers. Various algorithms consolidate multiple approaches to minimize the gap between performance and precision [6]. Although many algorithms have been published, there is no single unified solution for every case.

Various use-cases of pointer analysis have been presented over the years. Compilers rely on pointer analysis to optimize a program, for instance, register allocation, constant propagation, and code scheduling [4]. The LLVM compiler uses pointer analysis to perform Loop Invariant Code Motion (LICM), which optimizes load and store operation in loops [5]. A pointer security mechanism, such as Code-Pointer Integrity (CPI), can also benefit from pointer analysis by providing information to identify sensitive pointers to protect [8]. Moreover, pointer analysis is crucial to supply information about pointer information for other code analysis, such as control flow and data flow analysis [10].

3 Pointer Analysis Algorithm

Multiple approaches have been proposed to deal with pointer analysis problems. Generally, pointer analysis algorithms can be classified based on their sensitiveness to program behavior, e.g., control-flow and context [7]. Flow-sensitivity describes whether an algorithm includes program control flow in its analysis. Context-sensitivity, on the other hand, describes an algorithm which accounts for interprocedural relationships in analysis. Even though there are several other factors which define a pointer analysis algorithm, including type-sensitivity and field-sensitivity,

these two characteristics are commonly discussed in pointer analysis algorithms.

3.1 Andersen's Algorithm

Andersen [2] proposed an algorithm to build a points-to set using constraint rules that are applied to the program. The algorithm defines constraints for program structures, including declarations, definitions, and statements. The entire program is tagged with the respective constraints. The constraint itself describes a program operation which affects a pointer value, such as assigning pointer with a memory address. The algorithm performs constraint solving to build the points-to set that satisfies them.

```
int a, b, c,  
    *p1, *p2, *p3,  
    **pp1, **pp2;  
p1 = &a;  
p2 = &b;  
p3 = &c;  
pp1 = &p1;  
pp2 = pp1;  
*pp2 = p3; // p1 = &c  
pp1 = &p2;  
*pp1 = &a; // p2 = &a
```

Code 1. Sample program with pointer assignments in C

Practically, the algorithm examines every pointer assignment instruction and update the points-to set information of the left-hand side pointer with the right-hand side value. Consider the code 1, the first instruction assigns the address of variable a into pointer p1, therefore, a is added into p1 points-to set. Table 1 shows the progression of the analysis for every instruction point. Points-to set for every pointer is produced after every instruction is processed. Figure 1 shows the directed graph representation of the points-to set result.

Andersen's algorithm is a context-sensitive algorithm, where it differentiates each function call depending on the passed arguments and generate the constraint for the function call according to the called function points-to set. It is to include the possibility of the argument being involved in determining the return value of the function, which may differ across different function calls.

Instruction	p1	p2	p3	pp1	pp2
p1 = &a	{a}	{}	{}	{}	{}
p2 = &b	{a}	{b}	{}	{}	{}
p3 = &c	{a}	{b}	{c}	{}	{}
pp1 = &p1	{a}	{b}	{c}	{p1}	{}
pp2 = pp1	{a}	{b}	{c}	{p1}	{p1}
*pp2 = p3	{a, c}	{b}	{c}	{p1}	{p1}
pp1 = &p2	{a, c}	{b}	{c}	{p1, p2}	{p1}
*pp1 = &a	{a, c}	{a, b}	{c}	{p1, p2}	{p1}

Table 1. Generated points-to set result per instruction using Andersen’s algorithm

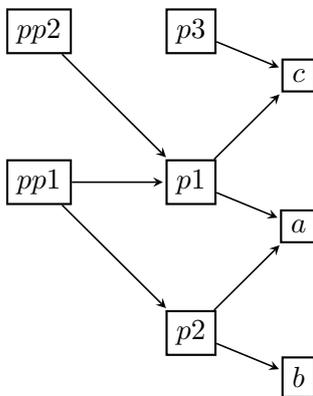


Figure 1. Points-to graph representation generated by Andersen’s algorithm

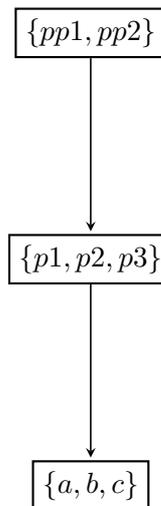


Figure 2. Points-to graph representation generated by Steensgaard’s algorithm

The order of instructions is not considered by the algorithm. The constraint do not reflect any information about program flow and the algorithm simply merges two points-to sets which satisfies the constraints. As a result, two different program flow may have identical points-to sets.

Assuming the program constraint generation has a linear complexity to the program size, the constraint solving algorithm has a cubic worst-case complexity ($\mathcal{O}(n^3)$) due to pointer dereferencing and function call context-sensitiveness. Pointer dereferencing requires the constraint to be propagated against all possible pointed objects. A function call requires an examination of its context according to the supplied argument which has substantial variation between different call context and possibly multiplies the number of constraints to process.

Instruction	p1	p2	p3	pp1	pp2
p1 = &a	{a}	{}	{}	{}	{}
p2 = &b	{a}	{b}	{}	{}	{}
p3 = &c	{a}	{b}	{c}	{}	{}
pp1 = &p1	{a}	{b}	{c}	{p1}	{}
pp2 = pp1	{a}	{b}	{c}	{p1}	{p1}
*pp2 = p3	{a, c}	{b}	{a, c}	{p1}	{p1}
pp1 = &p2	{a, b, c}	{a, b, c}	{a, b, c}	{p1, p2}	{p1, p2}
*pp1 = &a	{a, b, c}	{a, b, c}	{a, b, c}	{p1, p2}	{p1, p2}

Table 2. Generated points-to set result per instruction using Steensgaard’s algorithm

3.2 Steensgaard’s Algorithm

Steensgaard proposed a pointer analysis algorithm which has linear time complexity [16]. It uses a *storage shape graph* to model the runtime storage where the nodes represent storage locations or objects, while the edges represent pointer directions [3]. The storage graph describes location abstraction and potential runtime values from the connectivity between the nodes. The nodes may also be interconnected with each other, which represents an object being composed of different object types, e.g., declaration of a pointer to a pointer to an integer (`int **`).

Similar to Andersen’s approach, Steensgaard introduces several constraint to pointer statements in the program, including assigning and dereferencing a pointer. The constraint define a well-typed state to conform to the storage model graph and maintain the state in every program point, i.e., no conflict between nodes in the storage graph can emerge. The goal of the algorithm is to reduce the graph into smaller disjoint sets by merging multiple equivalent nodes with equal constraint.

Table 2 shows the progression of the algorithm and the merging behavior of Steensgaard’s algorithm. On first few instructions, the analysis behaves similarly with Andersen’s algorithm. However, starting from sixth instruction, the algorithm merges other unrelated pointer because the algorithm determines that they have equivalent constraint. This simplification improves the efficiency of the algorithm, in which the algorithm does not have to traverse multiple indirection to update the points-to set, with the cost of precision. Figure 2 shows the graph representation of the final points-to set.

Steensgaard utilizes the Union-Find data structure to perform the anal-

ysis in almost linear time. Every program instruction is only evaluated once, and the algorithm does not take program flow, such as conditional and loop, into account. Steensgaard's algorithm is also context-insensitive as it considers all function calls to be identical regardless of its calling context. A function call is determined only by its return value, which therefore will be represented as a single node in the storage graph.

4 Performance and Precision

Several studies have been discussing the performance and precision issues on implementing pointer analysis [6, 7, 9]. Researches are focusing on the trade-offs between them to improve the usability of the analysis. There is no golden standard on the balance between them, and it depends on the uses of pointer analysis itself. Several use cases require a more precise result, while some other requires faster analysis.

4.1 Criteria on Performance

Performance of pointer analysis can be evaluated by performing the analysis using actual source code input. The time performance greatly impacts the scalability of the analysis, and hence, deployability in a real-world settings. An algorithm with complexity larger than quadratic, such as Andersen's algorithm, may be impractical when dealing with a larger program.

Several studies compare the performance of pointer analysis algorithms, including Andersen's and Steensgaard's algorithms [7, 15, 14]. The studies are conducted using different test cases, which are based on actual applications. On a program with fewer lines of code, Andersen's algorithm performs almost as fast as Steensgaard's [15], which has a linear complexity [16]. For larger program sets, however, Steensgaard's algorithm yields a more consistent result, whereas Andersen's algorithm performance is highly dependent on the test case. This is a result of context-sensitivity behavior of the algorithm which multiplies the number of pointer sets on every different calling context.

Other than time complexity, the space complexity is also a challenge in developing a pointer analysis. Andersen's algorithm uses more space compared to Steensgaard's algorithm because of its context sensitivity [16]. Steensgaard's algorithm simplify this approach by treating all different

calling contexts as a single type of abstract location, hence, no redundant information is stored for every different calling context.

4.2 Analysis Precision

Pointer analysis precision indicates the accuracy of the generated points-to sets against the real values during the execution. The accuracy can be measured using the size of points-to sets. It is assumed that for every value in a points-to set result, there will be at least one value that represents the correct pointer value at runtime. Given that pointer analysis algorithms typically use formal rules against program semantics, the assumption will hold as long as the rules are correct [2]. Therefore, it is safe to assume that an algorithm is more precise if it yields smaller points-to set.

Shapiro and Horwitz carried an experiment on Steensgaard's and Andersen's algorithm precisions [15], in which they used C source code from multiple programs with different sizes, ranging from one hundred two twenty thousand lines. Andersen's algorithm is considered to be more precise than Steensgaard's algorithm because it yields smaller points-to sets. This is presumably due to the context-sensitivity of Andersen's algorithm, compared to Steensgaard's approach of assuming every call instruction to be equal.

Precision is affected significantly by the analysis characteristic. Although flow-sensitive analyses deliver a more precise results, they have severe scalability issues and often fail to deliver a time-efficient result for large program [12]. Since the need for precision does not outweigh the cost of processing, many approaches chose to disregard flow-sensitivity altogether.

William Landi as reported by Hind [6] observed that the precision of pointer analysis plays an important role to distinguish its practicality. Program optimization, for instance, requires the analysis to produce a correct result for every program input. However, a highly precise result is not necessarily required for every use case. False-positive results can be tolerated within a certain limit without sacrificing its benefits.

There are, however, several proposed methods which improve the precision over the state-of-the-art analyses. For example, Shapiro and Horwitz proposed to improve Steensgaard's approach by introducing categorization in the algorithm [15]. Each variable in the program is assigned with one category, and two variables will only be merged into a single points-to

set if and only if both variables belong to the same category. The precision can be further improved by iteratively executing the analysis with unique categorization on each run. They reported around 6 to 25% improvement over the original Steensgaard's algorithm depends on the test program.

5 Implementation in Modern Compilers

Modern compilers uses pointer analysis mainly for performing program optimization. Therefore, it is usually implemented in compiler back-ends¹, which means it operates on an intermediate representation rather than the original source code. This enables the analysis to be language agnostic rather than exclusive to a specific programming language. However, this might introduce some limitations as some semantic information is lost when the source code is transformed into intermediate code, thus restricting the analysis to distinguish high-level program semantics, such as data structure information.

5.1 Pointer Analysis Implementation in LLVM

LLVM provides an alias analysis interface where the API is structured for querying aliasing information, not a full pointer analysis points-to sets. This design decision dated back to the early development of LLVM², which focused on implementing a stateless constant-time alias analysis infrastructure for determining alias information between two pointers. Despite this API design, it allows implementing points-to analysis algorithm on top of it, although accessing the points-to set information is challenging without any provided API to obtain the points-to result set.

Currently, LLVM provides both Steensgaard's and Andersen's algorithm implementation, apart from other alias analysis algorithms. By default, LLVM uses Basic Alias Analysis, which is the stateless alias analysis implementation, and provides other alias analyses including Type-Based Alias Analysis and ScalarEvolution-based Alias Analysis.

Andersen's and Steensgaard's algorithms are implemented using a directed graph to describe the pointer, which is represented by a node; as-

¹Compiler is usually comprised of a modular structure of front-end and back-end, where the communication between the two uses intermediate representation [1]. Clang and LLVM pair is a notable example of this architecture.

²As examined from the commit history of the LLVM alias analysis infrastructure (lib/Analysis/AliasAnalysis.cpp, Commit ID: 7d58f8d)

signment between pointers, which is represented by an edge; and its value or referred object, which is a stored data in a node. Since both Steensgaard's and Andersen's algorithms are constraint and set based, they can be represented as graph and computed using graph-based algorithm. For example, the Andersen's algorithm implementation traverses the graph to propagate pointer values to every pointer assignment, i.e., pointer value data in a node is copied to every other connected node. On the other hand, Steensgaard's algorithm implementation uses Context Free Language (CFL)-reachability solution algorithm to transform the graph into an equivalent set by using a special data structure called *stratified set* [19].

It is theoretically possible to reconstruct points-to set information from alias information by naively querying all pointer pairs in the program. Must-alias and may-alias responses from the analysis inform that both pointers possibly point to the same value, thus it is possible to construct disjoint sets from that information. This workaround can be implemented in LLVM using the existing alias analysis implementation. However, this approach is inefficient as it is a combinatoric problem, which increases the problem set in quadratic scale apart from the efficiency of the analysis algorithm itself. Consequently, this approach is not applicable in some cases.

5.2 Use Cases

Several compile-time optimizations rely on pointer analysis algorithms for points-to sets and pointer aliasing information. One example that is implemented in LLVM is Loop Invariant Code Motion (LICM) [5]. This optimization attempts to reduce loop size by moving instructions that does not produce observable behaviors. The alias information is used to identify if dereferenced pointer inside the loop is not aliasing any other pointer that is written inside the loop. If the compiler is able to prove this, it can optimize the loop by pulling out the pointer dereference as its value will never change throughout the loop iteration.

Software security can benefit from pointer analysis to identify possible software bugs, for example, detecting Use After Free (UAF)³ vulner-

³Use After Free (UAF) is caused by an instruction trying to use a pointer obtained from heap memory allocation function (e.g., malloc, operator new) which has been released by calling the respective function (e.g., free, operator delete) [13].

abilities which Yan et al. use Andersen's algorithm to analyze possible candidate of UAF pointer access [18]. The detection algorithm uses the context-insensitive variant of Andersen's algorithm to produce a points-to set, which is then passed to the calling context reduction and path reduction mechanism to detect UAF. UAF detection requires a flow-sensitive algorithm which is not typically feasible in general purpose pointer analysis. Therefore, a separate mechanism is required to examine the program flow to identify the possible UAF instruction.

6 Discussion and Further Work

Pointer analysis has been an important program analysis tool in modern compilers. Andersen's and Steensgaard's algorithm is considered as a widely accepted standard for pointer analysis algorithms, and many improvement proposals derived from those approaches. Andersen's algorithm is considered to give a more precise analysis with trade-offs in efficiency and scalability. Whereas, Steensgaard's algorithm performs more efficiently with a trade-off in the precision. The users of the analysis have to consider these factors in order to pick suitable pointer analysis for their cases.

However, there is no recent comprehensive discussion about different pointer analysis algorithms. The most recent comparison study was done by Hind in 2000 and 2001 [6, 7]. Bibliographical literature referring to pointer analysis is mostly referring back to Aho, et al. literature about compiler design as a whole [1]. This indicates that this particular subject is not commonly known in general computer science community and is specific to compiler engineering.

On the implementation side, currently LLVM provides a comprehensive set of alias analysis algorithms, which can be used to support compiler optimization passes or program transformation and instrumentation. However, LLVM does not provide an API to query a points-to set of a pointer, which limits its practicality. A considerable amount of code refactoring will be required to enhance LLVM compiler to support producing points-to set information as the current alias analysis infrastructure itself is already very extensive by itself.

References

- [1] Alfred V. Aho. *Compilers: Principles, Techniques, and Tools (2nd Edition)*. Addison Wesley, sep 2006. URL: <https://www.xarg.org/ref/a/0321486811/>.
- [2] Lars Ole Andersen. *Program analysis and specialization for the C programming language*. PhD thesis, University of Copenhagen, 1994.
- [3] David R. Chase, Mark Wegman, and F. Kenneth Zadeck. Analysis of pointers and structures. In *Proceedings of the ACM SIGPLAN 1990 Conference on Programming Language Design and Implementation, PLDI '90*, pages 296–310, New York, NY, USA, 1990. ACM. URL: <http://doi.acm.org/10.1145/93542.93585>, doi:10.1145/93542.93585.
- [4] Manuvir Das, Ben Liblit, Manuel Fähndrich, and Jakob Rehof. Estimating the impact of scalable pointer analysis on optimization. In Patrick Cousot, editor, *Static Analysis*, pages 260–278, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.
- [5] LLVM Developer Group. Llmv alias analysis infrastructure. URL: <https://llvm.org/docs/AliasAnalysis.html#using-the-aliassetter-tracker-class>.
- [6] Michael Hind. Pointer analysis: Haven't we solved this problem yet? In *PASTE'01*, pages 54–61. ACM Press, 2001.
- [7] Michael Hind and Anthony Pioli. Which pointer analysis should i use? *SIGSOFT Softw. Eng. Notes*, 25(5):113–123, August 2000. URL: <http://doi.acm.org/10.1145/347636.348916>, doi:10.1145/347636.348916.
- [8] Volodymyr Kuznetsov, László Szekeres, Mathias Payer, George Candea, R. Sekar, and Dawn Song. Code-pointer integrity. In *Proceedings of the 11th USENIX Conference on Operating Systems Design and Implementation, OSDI'14*, pages 147–163, Berkeley, CA, USA, 2014. USENIX Association. URL: <http://dl.acm.org/citation.cfm?id=2685048.2685061>.
- [9] Maroua Maalej and Laure Gonnord. Do we still need new Alias Analyses? Research Report RR-8812, Université Lyon Claude Bernard / Laboratoire d'Informatique du Parallélisme, November 2015. URL: <https://hal.inria.fr/hal-01228581>.
- [10] Anders Møller and Michael I. Schwartzbach. Static program analysis, October 2018. Department of Computer Science, Aarhus University. URL: <http://cs.au.dk/~amoeller/spa/>.
- [11] R. Reese. *Understanding and Using C Pointers*. O'Reilly Media, 2013.
- [12] Barbara G. Ryder. Dimensions of precision in reference analysis of object-oriented programming languages. In Görel Hedin, editor, *Compiler Construction*, pages 126–137, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [13] Robert C. Seacord. *Secure Coding in C and C++*. Addison-Wesley Professional, 2005. URL: <https://www.amazon.com/Secure-Coding-Robert-C-Seacord/dp/0321335724?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimb05-20&linkCode=sm2&camp=2025&creative=165953&creativeASIN=0321335724>.

- [14] Marc Shapiro and Susan Horwitz. The effects of the precision of pointer analysis. In Pascal Van Hentenryck, editor, *Static Analysis*, pages 16–34, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.
- [15] Marc Shapiro and Susan Horwitz. Fast and accurate flow-insensitive points-to analysis. In *Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '97, pages 1–14, New York, NY, USA, 1997. ACM. URL: <http://doi.acm.org/10.1145/263699.263703>, doi:10.1145/263699.263703.
- [16] Bjarne Steensgaard. Points-to analysis in almost linear time. In *Proceedings of the 23rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '96, pages 32–41, New York, NY, USA, 1996. ACM. URL: <http://doi.acm.org/10.1145/237721.237727>, doi:10.1145/237721.237727.
- [17] Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley Professional, 4th edition, 2013.
- [18] Hua Yan, Yulei Sui, Shiping Chen, and Jingling Xue. Spatio-temporal context reduction: A pointer-analysis-based static approach for detecting use-after-free vulnerabilities. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, pages 327–337, New York, NY, USA, 2018. ACM. URL: <http://doi.acm.org/10.1145/3180155.3180178>, doi:10.1145/3180155.3180178.
- [19] Qirun Zhang, Michael R. Lyu, Hao Yuan, and Zhendong Su. Fast algorithms for dyck-cfl-reachability with applications to alias analysis. *SIGPLAN Not.*, 48(6):435–446, June 2013. URL: <http://doi.acm.org/10.1145/2499370.2462159>, doi:10.1145/2499370.2462159.

Access Control for The Internet of Things

Christian Yudhistira

christian.yudhistira@aalto.fi

Tutor: Mohit Sethi

Abstract

Authorization in the Internet of Things (IoT) is one of the fundamental factors in implementing IoT devices on an established communication network. In fact, current authorization solutions for web applications require high computational power. As many IoT devices are categorized as a constrained device, the implementation of authorization solutions requires different approaches.

In this paper, we compare two on-going authorization frameworks for IoT devices based on OAuth 2.0. The first solution is Authentication and Authorization for Constrained Environments (ACE) and the second solution is Device Authorization Grant for Browserless Device. Both frameworks are designed to enable clients to obtain an access token in a specific use case.

KEYWORDS: *IoT, authorization, OAuth 2.0, ACE, Browserless*

1 Introduction

The evolution of social Internet services such as blogs, photo sharing, and social media has fostered the growth of web applications that rely on the content of these Internet services [9]. The protected content hosted by online resource servers can typically be accessed by third-party web applications (clients) using Application Programming Interfaces (APIs). In the traditional authentication model, the client requests access for the protected resource by authenticating with the server using credentials (username and password) of the resource owner. Nevertheless, this solution creates potential problems to the privacy of the resource owner since the resource owner is unable to restrict scope and duration access to an authenticated client. Therefore, it is essential to establish identity management standards across online services which provide an alternative to sharing credentials of the resource owner.

The Single Sign-On (SSO) mechanism provides a central authentication and authorization authority which allows an authenticated client to access protected resources using an access token. The central authorization authority is usually assigned to a service provider. Nowadays, the Open Authorization 2.0 (OAuth 2.0 / OAuth) is one of the most widely deployed SSO frameworks in the web application [6]. The OAuth 2.0 is a web-based authorization framework that allows owners of the information to grant access from other entities to their resources (data or services) [6]. Third-party applications usually use the OAuth framework to request an access token and obtain access to owner resources within limited scope and duration. The third-party application is often referred to as a client. As a result of a valid request, the service provider will issue an access token and send it to the client in a JavaScript Object Notation (JSON) format as a replacement of the owner's credential. Originally, the OAuth framework is designed over the Hyper Text Transfer Protocol (HTTP) to support the authorization for web applications [3]. Nevertheless, the design of the OAuth framework allows implementation of OAuth over any protocol other than HTTP [8]. This property of the OAuth framework enables a new implementation of the OAuth framework in an Internet of Things (IoT) which typically has limited computational power. Traditional network protocols (e.g., HTTP and JSON) which are designed for

devices with more available resources could optionally be replaced with constrained Internet protocols, such as Constrained Application Protocol (CoAP) and Concise Binary Object Representation (CBOR) to enable the implementation of OAuth frameworks in the IoT.

The IoT consists of billions interconnected devices, involving both standard Internet devices which have relatively high processing power and small Internet devices which usually have limited processing power. Many of IoT devices are categorized as constrained devices [4], they operate in challenging environments, such as lossy and low-power networks, have limited computational power, and are usually battery-powered, thus requiring significant attention to maintain the energy consumption low. As a part of established communication networks, the implementation of IoT devices also require some aspects of network security components, including user access control and authorization. The existence of these components is even more critical for the IoT device because an adversary can easily exploit services which may compromise physical resources, such as digital locks in the smart home. From the fact that the majority of IoT devices have limited capabilities, the implementation of the OAuth 2.0 framework in the IoT has to deal with the limitations and challenges of constrained devices. The limited computational power of IoT devices may not be sufficient to perform the cryptographic primitives required for message authentications and digital signatures, which may have a negative impact on energy consumption. Moreover, the limited input capabilities of constrained devices may make users unable to enter their own credentials in an appropriate environment. In this paper, we aim to compare two on-going solutions based on OAuth 2.0 framework that are related to access control and authorization in IoT devices. One solution will discuss about Authentication and Authorization for Constrained Environments (ACE) framework based on the OAuth 2.0 [11] and we will compare it with the second solution about OAuth 2.0 Device Authorization Grant for Browserless and Input Constrained Devices [5]. We also suggest typical IoT applications which are most suitable for each authorization framework.

The rest of this paper is organized as follows, Section. 2 presents the relevant background of authorization and access control, Section. 3 explains the comparison between these two framework solutions and Section. 4 concludes the paper.

2 Background Theory

In this section, we provide background information about some authorization and access control technologies which will be compared in the Section. 3.

2.1 OAuth 2.0

The OAuth 2.0 authorization framework enables clients to get access to resources without using the owner's credentials. Instead, an access token is used by the client as a replacement of the owner's credential. In the implementation of the OAuth 2.0 framework, there are three roles which have to take into account:

- *Resource Owner* (RO), is an entity which provides their own services and give permissions to an authorization server to issue an access token;
- *Service Provider* (SP), is the host of user's services which returns a service after validating an access token within the resource request;
- *Service Consumer* (SC), is the client application which requests an access token to the authorization server and transmit it inside the resource request to obtain the protected service;

Each access token has a specific scope to the resource depend on the desired scope that is requested by the client. Therefore, every SC does not have full access to the resource which is not belong to them. This authorization framework is originally designed for use with HTTP [7].

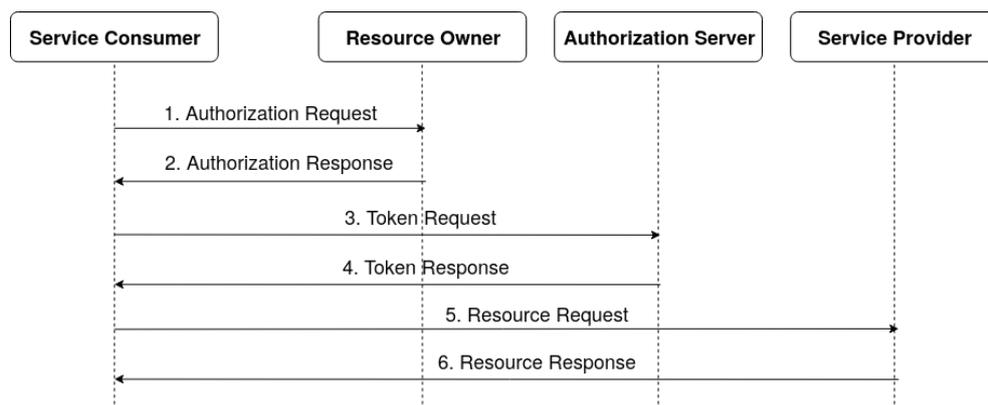


Figure 1. Flow of OAuth 2.0 Framework [8]

Figure. 1 shows the flow of the authorization process of OAuth 2.0 in the form of a sequence diagram as described in the following steps:

1. The Service Consumer sends an authorization request to the Resource Owner.

2. The Resource Owner issues an authorization grant, a credential which represents the resource owner's authorization, and sends it to the Service Consumer. The generated grant includes in one of the four grant type standards [8].
3. The Service Consumer presents the authorization grant to the Authorization Server within a request for an access token.
4. The Authorization Server attempts to validate the authorization grant prior to sending an access token to the Service Consumer.
5. The Service Consumer requests the resource from the Service Provider by providing the access token.
6. The Service Provider attempts to validate the access token and returns the resource after verifying the access token.

For a specific use case, the access token is determined by the interaction between the client, the resource owner, the authorization server and the resource server within a grant type. In the original OAuth 2.0, there are four grant types. The selected grant type is decided when the client request the authorization grant for the first time. This paper [6] described four grant types as follow:

- *Authorization Code*, the authorization code is the most secure flow because it separates secret information that is exposed to the client and the resource owner. It assumes the resource owner and the client application are on separate devices. After the resource owner presents her credentials into the browser, the access token does not pass through the browser. The access token is directly requested by the client to the authorization server using the authorization code that is issued by the authorization server after authenticating the resource owner. As a result of this design, the access token will not be exposed to other entities including the resource owner.
- *Implicit*, in the implicit flow, all the communication is happening through the browser and there is no backend server redeeming the authorization grant for an access token. It assumes the resource owner and the client are on the same device. Instead of transmitting intermediate credentials (such as authorization code) to the client, the authorization server directly sends an access token to the client.
- *Resource Owner Password Credentials*, in this flow, the resource owner presents her credentials (i.e., username and password) for the authorization server to the client. The client directly uses these credentials to obtain an access token from the authorization server.

This mode should only be used when there is a high degree of trust between the resource owner and the client (e.g., the client is part of the device operating system).

- *Client Credentials*, client credentials are typically used to obtain the access token when the client is accessing his/her own resource. This flow allows the client to request an access token using its client's credentials.

2.2 ACE OAuth 2.0

The ACE OAuth 2.0 framework defines one possible implementation of the OAuth 2.0 framework on IoT devices which typically do not have processing capabilities to apply traditional network protocols which are required high computational requirements [1]. Three building blocks enable the implementation of the OAuth 2.0 framework in IoT devices, such as:

1. Constrained Application Protocol (CoAP) [12], this protocol resides in the application layer and mainly handles messaging transfer in a constrained network. The request and response messages between IoT devices are transmitted using this protocol.
2. Concise Binary Object Representation (CBOR) [2], is a data format that is designed for small code and used for encoding the access token. This is used as a replacement of JSON format in the constrained device.
3. CBOR Object Signing and Encryption (COSE) [10], this is a standard security service in the application layer for the CBOR format. This protocol operates as an alternative or complement to transport layer security (e.g., Datagram Transport Layer Security).

In order to satisfy the performance requirements of IoT devices, the existing specification of the OAuth framework should add extensions which do not require high processing power. Therefore, some constrained protocols are used to replace traditional network protocol which may burden resources of IoT devices. CoAP is used in the constrained device to handle messaging in the application layer and optionally replace HTTP. In addition to provide security at the application layer, COSE is used to secure self-contained tokens such as proof-of-possession (PoP) tokens, which is an extension to the OAuth tokens. When distributing access tokens, the CBOR format is applied to replace JSON format that is used in the original OAuth so it will reduce memory footprint in the IoT device.

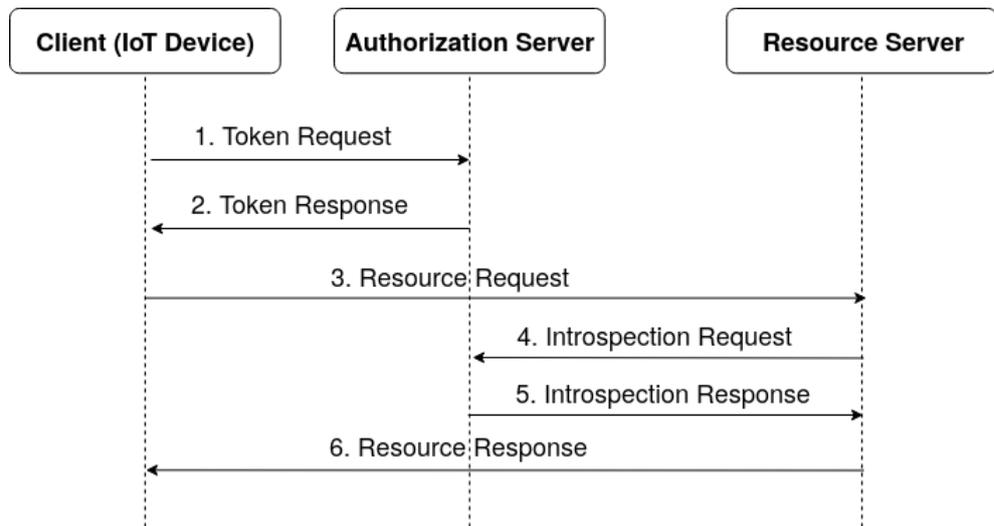


Figure 2. Flow of ACE OAuth 2.0 Framework [11]

The authorization flow illustrated in Figure. 2 describes the interaction between three roles:

1. The Client device makes a token request to the Authorization Server.
2. The Authorization Server returns an access token and optionally a refresh token after successfully process the token request.
3. The Client uses the received access token to request for an access to any resource in the Resource Server.
4. The Resource Server could directly process the request if the request token can be validated on its own or optionally forward the token to the Authorization Server before giving an access to the requested resource.
5. The Authorization Server returns the security parameter which will be used by the Resource Server to decide before giving permission to the Client.
6. The Client obtains the target resource if the request is authorized by the Resource Server.

The ACE framework defines a number of protocol flows that are determined by selected grant types. Based on [11], there are two preferred grant types which typically work best in the IoT environment, such as:

- *Authorization Code*, this flow is a good choice for applications that require interactions between mobile applications with IoT devices. One example is in the smart home, where the end user uses the mobile application to present the user's credentials before gaining access to physical resources which are attached to IoT devices.
- *Client Credentials*, The Client Credential Grant is a good fit for use

with IoT devices where the OAuth client itself is a constrained device.

2.3 Device Authorization Grant for Browserless Device

The original OAuth 2.0 framework has one implementation model which requires the end-user to present his/her credential (username and password) to the authorization server before receiving a granted access to the resource. However, most of IoT devices typically lack browser feature (Browserless) and requiring the end-user to input his/her credentials is impractical. These browserless devices usually include wireless speakers, digital picture frames and printers. The device authorization grant framework enables browserless devices to communicate with a secondary device (e.g., smartphone, tablet) which support browser feature and internet connectivity to send an verification code to the authorization server and obtain authorized access to the target resource.

Based on [5], the implementation of the device authorization grant framework comprises of four mandatory components, such as:

- The client device has an Internet connectivity.
- The client device provides a HTTPS requests capability.
- The client device is able to display or communicate a URI and verification code to a secondary device of the user.
- The user has a secondary device (e.g., personal computer or smartphone) from which they can process the request.

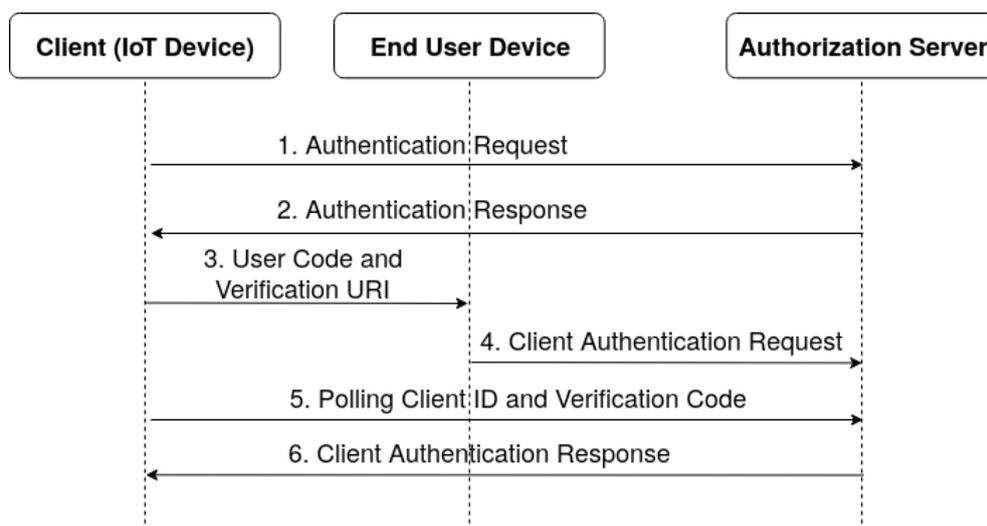


Figure 3. Flow of Device Authorization Grant Framework [5]

The flow which is showed in Figure. 3 can be described in the following steps:

1. The Client device requests access from the Authorization Server and includes its client identifier. The client identifier is a unique id that issued by the authorization server to the registered client [8].
2. The Authorization Server returns a verification code, an end-user code and a verification URI for the end-user device.
3. The Client device forwards the end-user code code and the verification URI to the end-user device. The end-user access the verification URI with his/her secondary device and input the end-user code.
4. The authorization server processes the end-user code and prompts the user to grant the client's access request.
5. The IoT device periodically send client id with the verification code to the authorization server to check if the user has completed the user authorization step.
6. The Client receives the access token if the authorization server manages to validate the verification code provided by the client.

While IoT devices that operate in the ACE framework have limited processing power, the device authorization grant framework has common IoT devices which have higher processing power constrained. This condition allows IoT devices to implement traditional network protocol to send an authentication request to the authorization server.

The client initiates the authorization flow by requesting a set of verification codes from the authorization server by making an HTTP "POST" request to the authorization server. All requests from the device must use the Transport Layer Security (TLS) protocol. In response, the authorization server generates a unique device verification code and an end-user code that are valid for a limited time and includes them in the HTTP response body using the JSON format with a 200 (OK) status code.

3 Framework Comparison

In this section, we compare both the ACE framework and the Device Authorization Grant framework from several aspects. Subsection. 3.1 presents specific use case for each framework and Subsection. 3.2 discusses deployment aspects on each authorization framework.

3.1 Target Use Case

As described in the subsection. 2.1, there are four grant types which define the flow to obtain the access token for the client. Each grant type is selected for a unique use case. The authorization code grant type is designed for a case where the client operates on behalf of the resource owner. However, in case the client acts on behalf of the resource owner and provides very limited input mechanism to the user, it is recommended to use the device authorization grant type.

One typical use case for the authorization code grant is in the smart home application. A real example for this case is a smartphone (client) which operates as a wireless door key and requests access to the authorization server to control a door lock which is integrated into an IoT device (resource server). It assumes that both the client and the resource server have been registered to the authorization server so the following resource request and response come from authenticated entities. In the owner authorization step, the end user (resource owner) presents owner credentials (username and password) to the authorization server. After granting owner authorization, the authorization server sends the authorization code to the client. This authorization code can be used by the client to receive the access token from the authorization server. When issuing the access token, the authorization server adds claims which are used by the resource server to maintain the access scope of every authorized request from the client. After receiving the access token, the end user allows to use her smartphone to control the door lock.

On another use case, an example for the device authorization grant is a wireless speaker (client) which request access to a user account in a cloud music service (resource server), e.g., Spotify and Joox. When the speaker tries to connect for the first time with the authorization server, the speaker forwards the client id of the speaker to the authorization server. As a return, the authorization server sends the verification code, the end-user code and the verification URI to the speaker. This initial process occurs without end-user interaction. After this process, the end user tries to pair her smartphone with the speaker using Bluetooth technology. After both devices have been paired, the speaker will forward the user code and verification URI to the end user (resource owner) whose identity has been registered in the cloud service beforehand. The end user will decide before giving access for her account to the speaker. Mean-

while, the speaker repeatedly polls the authorization server to find out if the user completed the user authorization step. After the end user grants access to the client speaker, music services can be accessed by the speaker afterward.

3.2 Deployment Aspects

For each entity that operates on a framework, as shown in Figure. 2 and Figure. 3, there are numbers of possible cases that may occur during the running process. There is a case when one entity unable to transmit authorization information (e.g., access tokens, public key, and verification code) to another member within the same framework at an unpredicted time.

Local token validation in the ACE framework [11]

In a network which implements the ACE framework, there is a case when the resource server is not connected to the authorization server after the client sending a resource request. Therefore, the resource server cannot retrieve the authorization information associated with each access token from the authorization server. In this case, the self-contained access token has a security property which can be used by the resource server to verify the access locally. The self-contained access control has a "scope" element which contains information about permitted accesses (claims) that can be requested by the client towards the resource server. The "scope" element is issued by the authorization server when the client first requests for the access token. The authorization server encodes authorization information along with a signature into the token and returns it to the client. When the resource server receives the self-contained access token from the client, the authorization information of the access token informs the server that the client has authorized access for resources inside "scope" element. As a result, it is not necessary for the resource server to verify the access token in the authorization server as long as the client sends self-contained access control when requesting access.

Non-browser user interaction in the Authorization Grant framework [5]

For some electronic devices (e.g., wireless speaker and smart home hub) that do not provide display screens for the user interaction, the implementation of the Device Authorization Grant framework requires different user interaction methods. One alternative user interaction for transmit-

ting the verification URI and the user-code is via Bluetooth. This alternative does not require a browser and manual input of the code. Instead, there is an authorization server's companion app which will receive the user code and the verification URI from the paired device and sends this information to the authorization server.

4 Conclusions

This paper compares two on-going authorization frameworks for IoT devices based on the OAuth 2.0 framework. From the standard of The ACE framework, it is typically designed for client devices which have limited processing power. On another side, the standard of the device authorization grant framework [5] specifies the implementation of the framework on devices which do not have a browser feature. Both frameworks enable OAuth clients to obtain authorization access from the resource owner using an access token. The access token associates the client with the owner credential within a limited scope. The security property of the access token avoids the client to obtain unlimited access to the owner resource.

References

- [1] Carsten Bormann, Mehmet Ersue, and Ari Keranen. Terminology for constrained-node networks. Technical report, 2014.
- [2] Carsten Bormann and Paul Hoffman. Concise binary object representation (cbor). Technical report, 2013.
- [3] Eric Y. Chen, Yutong Pei, Shuo Chen, Yuan Tian, Robert Kotcher, and Patrick Tague. Oauth demystified for mobile application developers. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, CCS '14*, pages 892–903, New York, NY, USA, 2014. ACM.
- [4] Simone Cirani, Marco Picone, Pietro Gonizzi, Luca Veltri, and Gianluigi Ferrari. Iot-oas: An oauth-based authorization service architecture for secure services in iot scenarios. *IEEE sensors journal*, 15(2):1224–1234, 2015.
- [5] William Denniss, John Bradley, Jones, and Hannes Tschofenig. OAuth 2.0 device authorization grant. *IETF, Internet-Draft draft-ietf-oauth-device-flow-15*, 2019.
- [6] Daniel Fett, Ralf Küsters, and Guido Schmitz. A comprehensive formal security analysis of oauth 2.0. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1204–1215. ACM, 2016.

- [7] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. Technical report, 1999.
- [8] Dick Hardt. The oauth 2.0 authorization framework. Technical report, 2012.
- [9] B. Leiba. Oauth web authorization protocol. *IEEE Internet Computing*, 16(1):74–77, Jan 2012.
- [10] Jim Schaad. Cbor object signing and encryption (cose). Technical report, 2017.
- [11] Ludwig Seitz, Goeran Selander, Erik Wahlstroem, Samuel Erdtman, and Hannes Tschofenig. Authentication and authorization for constrained environments (ace) using the oauth 2.0 framework (ace-oauth). *IETF, Internet-Draft draft-ietf-ace-oauth-authz-23*, 2019.
- [12] Zach Shelby, Klaus Hartke, and Carsten Bormann. The constrained application protocol (coap). Technical report, 2014.

Distributed data analytics at the edge

Tommi Askola

tommi.askola@aalto.fi

Tutor: Gopika Premsankar

Abstract

The amount of data generated by devices in Internet of Things (IoT) has grown rapidly in past few years. The IoT data is processed in the cloud, for most part in distant data centers. As a result, high latency and high response time become an issue when sending huge amounts of data to the data centers. Edge computing introduces a way to reduce latency by bringing computation resources closer to the end user. This paper focuses mainly on applications that process data on the edge as a survey of state of the art in this area. We focus on existing edge architecture and software solutions to run real-time data analysis on IoT sensor data.

KEYWORDS: edge computing, fog computing, Internet of Things, real-time data analytics

1 Introduction

The amount of data generated by devices in the Internet of Things (IoT) has grown rapidly in the past few years. Also the number of devices connected to the internet in 2020 is expected to be 50 billion whereas in 2003 there were only 500 million [1]. In 2011 International Data Corporation

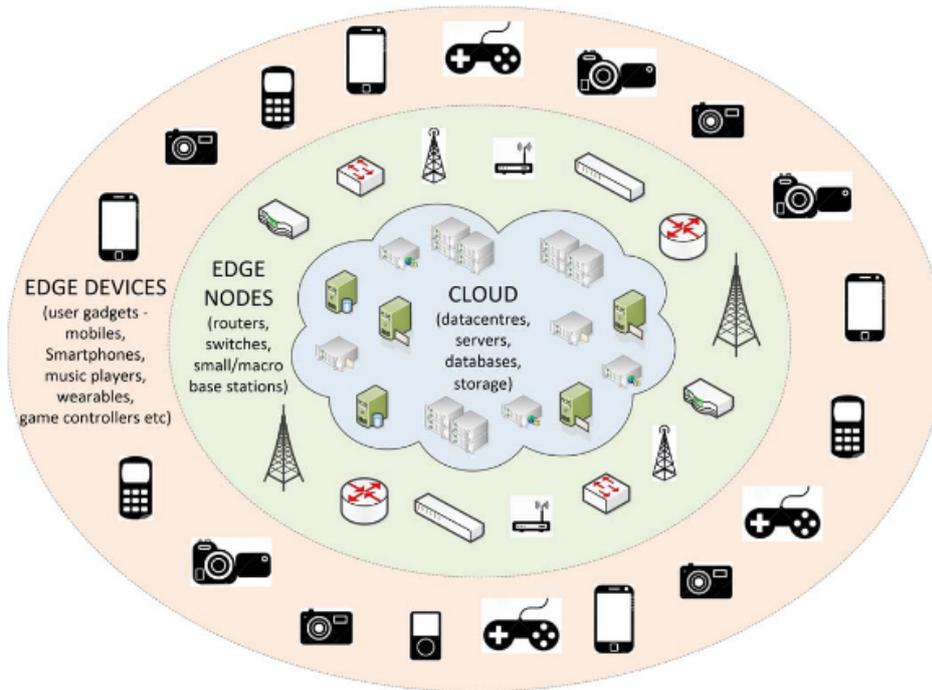


Figure 1. Edge devices and edge nodes that can be used in edge computing [6].

(IDC) estimated that the amount of data produced that year would surpass 1.8 zettabytes (1.8 trillion Gigabytes) and in 2010 it had surpassed 1.0 zettabyte [2]. IoT devices and sensors that create data have limited computing capabilities. Thus, processing of this data is offloaded to the cloud to improve battery life and increase performance of the applications [3]. However, when data is transmitted to distant cloud data centers, the application response time grows. This is a problem in mobile gaming or video streaming when latency should be low to guarantee good user experience. Edge computing [4] and fog computing [5] have been presented as solutions to reduce latency and also handle issues with privacy and data safety. Shi et al [4] define edge computing as a computing that happens in a resource between data sources and cloud data centers. In edge computing the computing happens at the proximity of the data sources, at the edge devices. With the help of edge computing we can reduce latency and can deal with huge amount of data that is growing every year [6]. In Figure 1 edge devices and edge nodes are shown which can be used in the computation instead of the cloud.

In this paper we will look into edge computing, especially in data analytics, and compare existing implementations where edge or fog computing has been used in data analysis. In section 2 we compare state of art architectures in distributed data processing. In section 3 we focus on implemented algorithms and frameworks. Section 4 summarizes the contents

of this paper.

2 System architecture

In this section we focus on existing edge computing-assisted architectures in IoT analysis. We look into existing architectures in the field of smart cities, smart homes and healthcare. This section covers the architectures have been proposed. In section 3 we look into the algorithms and frameworks of the same papers.

A smart city is an urban area where analysis of real-time data is used to achieve sustainable outcomes. Analysis of real-time data from smart cities is used to improve quality of life for example by reducing traffic congestion and energy waste. Smart city is projected to have a massive impact in economics in the coming decades, as it has been estimated that hundreds of millions of jobs will be created to facilitate conversion to smart city. Smart cities have many challenges such as geospatially distributed sensing networks and big data analysis. The big data in smart cities needs to be processed near the sensors at the edge instead of data centers, in order to achieve low latency and location awareness. Location awareness is important in order to react to location based behavior. [7]

Hierarchical distributed fog computing architecture for big data analysis in smart cities has been proposed to guarantee low latency and location awareness. In this architecture, intelligence is distributed at the edge of a layered fog computing network. Tang et al. [7] proposed 4-layer network that is shown in Figure 2. Layer 4 contains sensor nodes that are widely distributed throughout the smart city. They measure physical changes such as temperature and forward measured data to layer 3. Layer 3 consists of many edge devices. Each edge device is connected to a local group of sensors. Layer 3 nodes identify potential threat patterns and perform feature extraction so that the raw data does not have to be transmitted to layer 2. Layer 3 also has a simple and fast feedback control to respond to local and small threats. Layer 2 nodes are connected to tens of layer 3 nodes. Layer 2 nodes use spatial and temporal data to recognize potentially hazardous events and control the infrastructure when such an event is observed. The data analysis of the layers 2 and 3 is forwarded to layer 1. Layer 1 consists of data centers in the cloud. At the top layer very large-scale (city wide) and very delay tolerant (years) computing tasks are performed. [7]

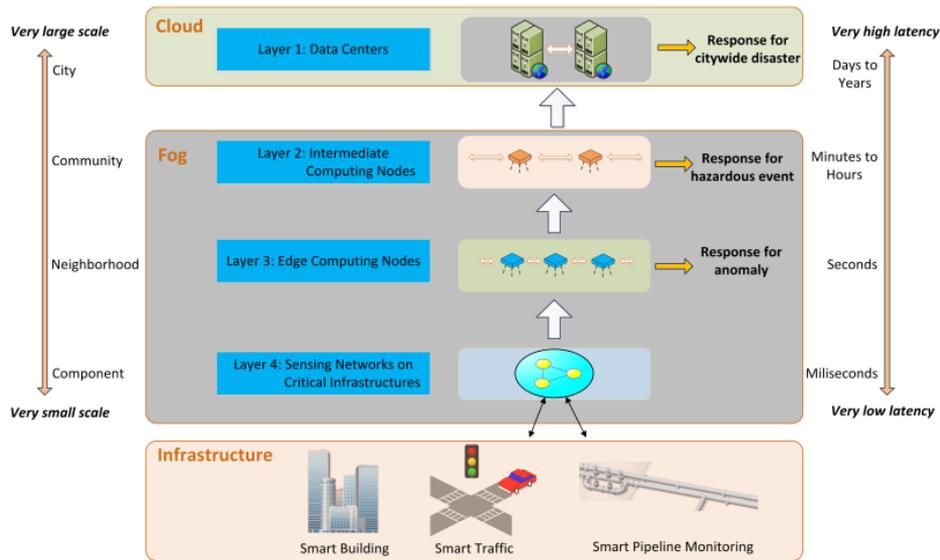


Figure 2. The 4-layer Fog computing architecture in smart cities. [7]

Edge analytics can also be carried out on crowdsourced data. Crowdsourcing can be used for marketing and advertising, locating missing people and public safety. Video feed captured for one reason can be useful for totally unrelated reason. For example, video feed in surveillance cameras can be used for tracking movements of a missing person. However the sheer number of surveillance cameras, police patrol cameras etc. is so huge that all of the raw data cannot be sent to the cloud for processing. Satyanarayanan et al. [8] have proposed an architecture called GigaSight that uses a decentralized edge computing infrastructure in form of virtual machine based-cloudlets. The architecture of GigaSight is illustrated in Figure 3. Cloudlet is defined as a "data center in a box" that brings computation closer to the user. In GigaSight every cloudlet analyzes video streams it receives and sends to the cloud only the recognized features and metadata of the video stream. In addition to analytics, the cloudlets also denature the video streams to preserve privacy. Denaturing is editing out frames or blurring certain objects. [8]

Yassine et al. [9] propose an architecture for IoT big data analytics for smart homes using fog and cloud computing. Smart homes generate huge amount of data from smart devices and appliances. Analyzing this data real-time allows us to deduct information that can be used to improve people's safety, health and economy. For example, by monitoring usage of appliances in a smart home we can notice abnormal activities which could indicate health problems. However, the amount of data produced can be huge. To handle this massive amount of data, the authors have proposed

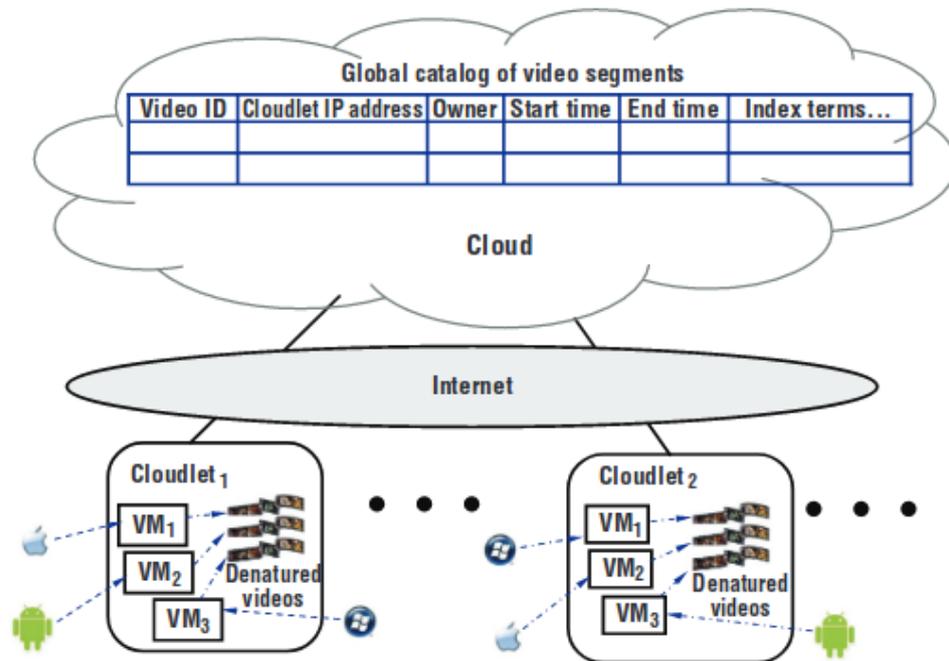


Figure 3. The GigaSight architecture. It uses cloudlets to process the data and sends only features and metadata to the cloud. [6].

a system which uses fog computing. The architecture consists of IoT big data analytics with fog computing and cloud system. The fog nodes allow faster data processing because they are situated closer to the physical location of the smart home. The cloud system is responsible of processing computationally intensive tasks. The data to be analyzed is produced by smart home components such as sensors, devices and appliances. The fog computing nodes handle data pre-processing, pattern mining, classification, prediction and visualization. These functions are responsible for quick analysis of the data and send the results to the cloud or directly to serviced applications. The cloud system is responsible for historical data analytics and storing the data. [9]

Healthcare is a promising area for an IoT application. The IoT can be used in many medical applications such as remote health monitoring, fitness programs, supervision of chronic diseases and elderly care. IoT based applications are expected to increase the quality of life and improve user experience [10]. Edge or fog computing can significantly reduce latency and response time of an IoT application. This can be important when it is critical to react in a short time to a certain event. [11]

Dubey et al. [11] describe how fog computing can improve IoT data processing from wide variety of wearable sensors such as smart watches and wearable ECG systems. These wearable devices collect health data from

human body around the clock; thus, the amount of data produced is huge. Health applications require real time and fast data processing. Collecting, storing and analyzing big data in body sensing network (BSN) is a challenge. It is inefficient and impractical to use traditional cloud architectures to store and process the health data. Also wearable sensors produce, among valuable information, non-deterministic errors and data can be corrupted. So it is time consuming and expensive to send raw medical data to the servers in the cloud. To overcome these issues the authors proposed an architecture with 3 sub-systems. One sub-system consists of a BSN to collect the data, another consists of a fog gateway for on site processing and third consists of a cloud server for storage and back-end analysis. The fog gateway helps to reduce the amount of data to be stored in the cloud as it filters out irrelevant information that is not needed for analysis. The fog gateway has limited storage and computing capabilities compared to the cloud, so the processing in the fog needs to be computationally simple and clinically relevant. The fog gateway converts the data into features or patterns with time-stamps that are sent to the cloud to be analyzed. The used architecture reduces data transmission and storage by processing the data before transmission. [11]

Cao et al. [12] focus on using fog computing in pervasive health monitoring. Pervasive health monitoring is one of the biomedical application areas that utilizes big data. The same problems arise as with traditional methods of sending data to the cloud: high latency and high response time. The authors employ a pervasive fall detection for a stroke mitigation application named U-Fall. The experiment is relevant because falls are a major cause of mortality among old people and stroke patients often have multiple falls in a year. U-Fall splits the data analytics, in this case fall detection, between the edge devices and the server. Fall detection and simple filtering of false alarms is done in the fog computing. Computationally complex filtering of false positives is done in the cloud. [12]

Rahmani et al. [13] proposed a smart e-Health gateway, called UT-GATE, at the edge of healthcare IoT. Their architecture consists of 3 layers: medical sensors and actuators network, edge/fog layer and cloud layer. The first layer has sensors that capture biomedical data from the body and room. This data is then transmitted to the second layer, that acts as a smart e-health gateway, via wireless or wired communication protocols such as Bluetooth or Wi-Fi. The fog is formed from multiple geographically distributed smart e-Health gateways. This second layer receives

data from different subnetworks. The key feature of the fog layer is local data processing as it provides intelligence at the gateway. The fog layer also handles data compression, filtering and fusion. With help of this fog layer the system can react faster to certain events. For example, it can provide fall-detection without sending the data to the cloud to be analyzed and thus save time. The cloud layer implements broadcasting, data warehouse and data analytics. [13]

3 Algorithms and frameworks

In this section we look into the algorithms and frameworks that were used in implementation of those architectures that we looked on section 2.

Tang et al. [7] implemented a prototype of their hierarchical 4-layer network in smart cities. The authors experiment with a smart pipeline monitoring system. In the experiment layer 4 has optical fibers as sensors that measure temperature along the pipeline. At layer 3 machine learning algorithms try to identify threat patterns from the data stream. Also the raw sensor data is discarded and only extracted feature patterns are transmitted to the layer 2. Each layer 2 computing node receives features from multiple of layer 3 edge devices. The layer 2 nodes combine these features and use hidden Markov model (HMM) to process the data. They use Baum Welch learning algorithm for learning and for classification they use maximum a posteriori (MAP) rule. The cloud at layer 1 is built using Hadoop. The results obtained with this prototype are encouraging. The amount of data transmitted was only about 0.02% of the raw data. Also the response time reduced significantly using this architecture. [7]

Yassine et al. [9] conducted a case study with their architecture for IoT big data analytics using fog and cloud computing. They use data which contains millions of records of usage of appliances over a span of two years. The data is first cleaned in the fog nodes using Python scripts with regular expressions (RegEx). After cleaning the data feature pattern mining is applied. This helps to deduct how the smart home occupants are using their appliances. The authors also implement k-mean clustering algorithm to further analyze the usage of appliances. With this algorithm they can deduct the usage level of an appliance during a certain time period. The case study was run in one fog node. The running time of processing the data, that was produced over 2 years, took only few minutes. Thus, one fog node is capable of processing data from more than one

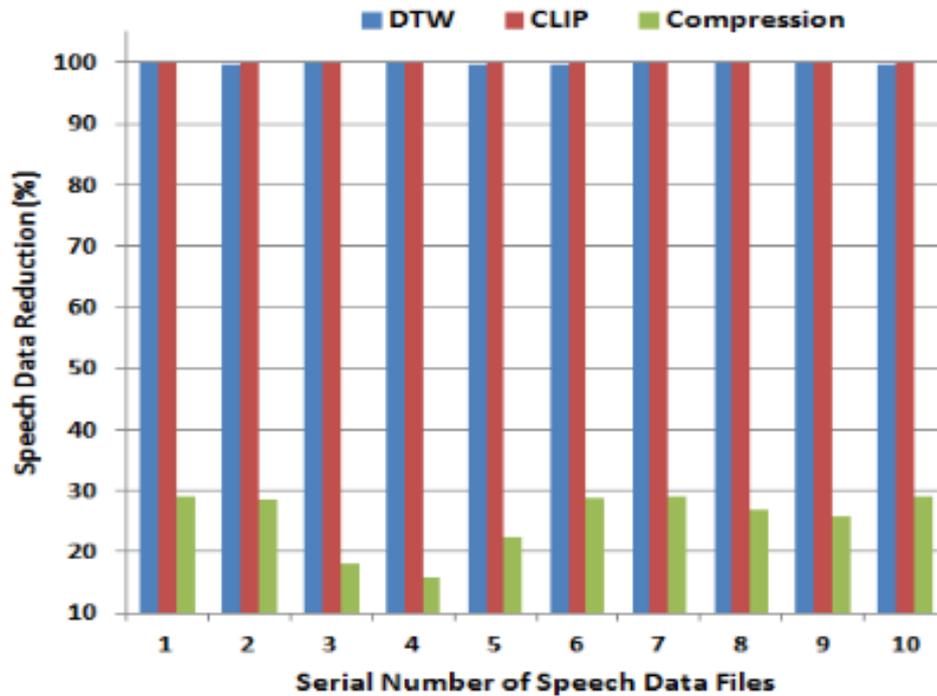


Figure 4. The percentage of data reduction achieved using dynamic time warping (DTW), clinical speech processing (CLIP) and GNU zip compression on 10 speech files. [11]

smart home. [9]

Dubey et al. [11] have different methods for the analysis of the health-care data. They use Dynamic time warping (DTW) for mining patterns in time-series data. They also use Clinical speech processing chain (CLIP) to filter speech data in order to compute the clinically relevant metrics such as loudness and frequency. They also compress and decompress the data using GNU zip in their experiments. The compression was done in the fog processor that sends the compressed data to the cloud. The authors study remote delivery of speech treatment for people with speech disorders. Speech data from smartwatch is transmitted to the fog computer that extracts features and patterns from the data. These features and patterns are then sent to the cloud to be analyzed and used in the treatment by Speech-language pathologists. Figure 4 shows percentage of data reduction with these three methods in speech data. CLIP and DTW have both over 99% data reduction. However, GNU compression is only one of the three methods that is not lossy, meaning the GNU compression can be reversed back to achieve the original speech data. [11]

The fall detection system U-Fall consists of a front-end and a back-end. There are two algorithms running in front-end: RSS detector, ADLs filter. RSS detector uses root-sum-of-squares (RSS) of acceleration magnitude

to detect potential falls. RSS detector can produce a low miss-rate but at the same time many false positives. To filter out these false positives ADLs filter is needed. ADLs filter tries to remove events that are from activities of daily livings (ADLs). In back-end the data processed with orientation filtering and nonlinear time series analysis. The authors implemented U-Fall algorithm using Android as the front-end and Amazon AWS as the back-end. U-Fall was compared to two state-of-the-art fall detection algorithms. U-Fall had good response time, energy consumption, sensitivity and specificity when comparing with the other two fall detection algorithms. [12]

Rahmani et al. [13] implemented a case study of their UT-Gate smart gateway. In the implementation they used free service provided by “heliohost.org” for the remote server. It includes MYSQL with remote access. Server side scripts are implemented in PHP. Compression of the data is done in the fog layer using LZW algorithm. The authors tested effectiveness of their system by implementing a ECG signal processing in the sensor layer, in the fog layer and in the cloud and comparing the energy efficiency and amount of data transmitted. They also apply a ECG threshold-based feature extraction algorithm to the data. Processing the data at UT-GATE is more energy efficiency than processing data at the sensor nodes. About 56% of energy is saved when the processing is done in the fog layer. Processing the data at UT-GATE also decreases the data flow to the cloud by 74%. [13]

Rahmani et al. [13] also implemented Early Warning System with their UT-GATE architecture. The system is a tool for estimating the degree of illness by calculating Early Warning Score (EWS). EWS uses patient’s medical parameters to find abnormal signs. Figure 5 shows the three layers of the system and their functions. At the first layer there are a network of sensors that are divided into three groups: medical sensors, environmental sensors and activity sensors. The second layer receives sensor data via wireless communication. The gateway is a UDP server implemented by Node.js. UT-GATE first uses a bandpass filter with Finite Impulse Response (FIR) to reduce noises from the incoming ECG signal. Python is used for preliminary data analysis. The fog layer performs real-time EWS calculation. If the EWS score is elevated there is a higher probability that patient’s health is deteriorating. In that case medical experts are notified. The data is further compressed with tar and encrypted with Python’s Crypto library. The data is then sent to the cloud server.

The third layer consists of a cloud server and user interface for patients, caregivers and medical experts. The cloud server further processes the data and provides reports, suggestions and possible alerts. [13]

4 Conclusion

In this paper, we collected existing literature in the field of distributed data analytics using edge or fog computing. Especially we were interested of researches that contained implementations of data analytics at the edge. We found out that fog computing helps handling the huge amount of data that devices and sensors produce by bringing computation resources closer to the user and the sensors. Most of the architectures examined were hierarchical, consisting at least of a sensor layer, a fog or edge layer and the cloud. The fog layer is suitable for the pre-processing of the data, data filtering and computationally lightweight analysis. In these architectures the cloud was used for computationally heavy analyzing tasks and as a storage. The amount of data transmitted to the cloud can decrease significantly, when using edge devices to do the analysis instead of the cloud. Also fog computing helps to cut down response times and latency which is important when fast reaction to a certain event is essential. For example, when a person falls down, reacting quickly might save the person's life.

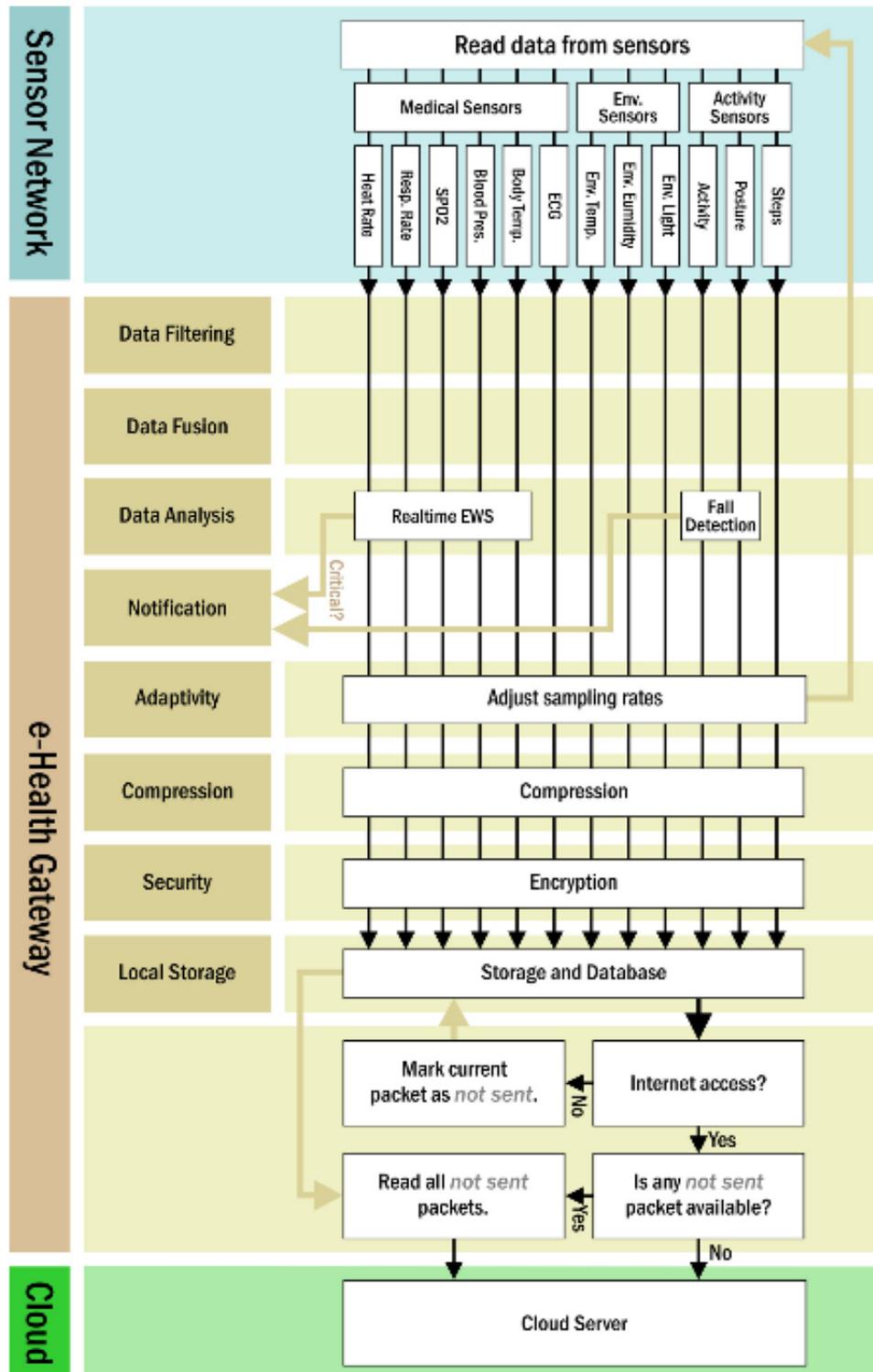


Figure 5. The fog-based EWS System consisting of three layers. [13]

References

- [1] Dave Evans. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper*, 1(2011):1–11, 2011.
- [2] John Gantz and David Reinsel. Extracting value from chaos. *IDC view*, 1142(2011):1–12, 2011.
- [3] Hoang T Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless communications and mobile computing*, 13(18):1587–1611, 2013.
- [4] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, 3(5):637–646, 2016.
- [5] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012.
- [6] Blesson Varghese, Nan Wang, Sakil Barbhuiya, Peter Kilpatrick, and Dimitrios S Nikolopoulos. Challenges and opportunities in edge computing. *arXiv preprint arXiv:1609.01967*, 2016.
- [7] Bo Tang, Zhen Chen, Gerald Heffernan, Tao Wei, Haibo He, and Qing Yang. A hierarchical distributed fog computing architecture for big data analysis in smart cities. In *Proceedings of the ASE BigData & SocialInformatics 2015*, page 28. ACM, 2015.
- [8] Mahadev Satyanarayanan, Pieter Simoens, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, Wenlu Hu, and Brandon Amos. Edge analytics in the internet of things. *IEEE Pervasive Computing*, 14(2):24–31, 2015.
- [9] Abdulsalam Yassine, Shailendra Singh, M Shamim Hossain, and Ghulam Muhammad. Iot big data analytics for smart homes with fog and cloud computing. *Future Generation Computer Systems*, 91:563–573, 2019.
- [10] SM Riazul Islam, Daehan Kwak, MD Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. The internet of things for health care: a comprehensive survey. *IEEE Access*, 3:678–708, 2015.
- [11] Harishchandra Dubey, Jing Yang, Nick Constant, Amir Mohammad Amiri, Qing Yang, and Kunal Makodiya. Fog data: Enhancing telehealth big data through fog computing. In *Proceedings of the ASE bigdata & socialinformatics 2015*, page 14. ACM, 2015.
- [12] Yu Cao, Peng Hou, Donald Brown, Jie Wang, and Songqing Chen. Distributed analytics and edge intelligence: Pervasive health monitoring at the era of fog computing. In *Proceedings of the 2015 Workshop on Mobile Big Data*, pages 43–48. ACM, 2015.
- [13] Amir M Rahmani, Tuan Nguyen Gia, Behailu Negash, Arman Anzanpour, Iman Azimi, Mingzhe Jiang, and Pasi Liljeberg. Exploiting smart e-health gateways at the edge of healthcare internet-of-things: A fog computing approach. *Future Generation Computer Systems*, 78:641–658, 2018.

Data Analytics using Azure IoT Edge

Md.Kamrul Islam

kamrul.islam@aalto.fi

Tutor: Behrouz Jedari

Abstract

Cloud computing allows applications and services with various storage and computing requirements to perform their tasks through powerful centralized servers. However, the architecture of cloud computing constrains applications with different data analysis and processing tasks to transmit their content to centralized servers, which can result in long response delay. Recently, the concept of mobile edge computing (MEC) has been introduced to meet the high-bandwidth and low-latency quality requirements of fifth-generation (5G) networks and Internet of Things (IoT) applications and use cases. The main idea in MEC is to bring the network resources as close as possible to mobile users, thus reduce the traffic in the backhaul network while improving the quality of experience of mobile user. Recently, a number of edge data computing and analysis platforms have been introduced. In this report, we introduce Azure IoT Edge as a prominent MEC platforms and explain its main features. Specifically, we overview the Azure cloud platform, the IoT edge platform, and the IoT edge architectures. Next, we conduct a GPS tracking system, namely GPSEEDGE, to comparatively analyze the performance of cloud and edge services in terms of latency. Finally, we present the experimental results and conclude the report.

KEYWORDS: 5G networks, Mobile edge computing, Internet of Things, Microsoft Azure IoT Edge.

1 Introduction

The Internet traffic is expected to grow enormously in the next few years [10]. According to Cisco global cloud index [21], data produced by people, machines and things will be around 50 zettabytes and IP traffic capacity will increase 10.4 zettabytes. The majority of the Internet data will be generated from mobile devices and Internet of Things (IoT) devices, such as sensors, actuators, smart phones, and wearable devices. For instance, self-driving cars, industrial IoT, and other connecting services (e.g., a connected plane or connected hospital applications) require flawless communication between devices with the centralized cloud data center. Since most of the aforementioned applications require low latency and high computation Internet connectivity to provide low-latency services, the architecture of cloud computing is generally not an optimal solution to meet their service requirements. The main reason is that the cloud computing servers and data centers are generally far from the end users. Thus, The round trip time between a request submitted by devices and a response from the cloud is generally long, which results in poor application performance [24].

A long network latency could hinder the decision-making process for certain types of services, for example a self-driving car has to make decision instantly. A minor delay could cause a fatal accident. Therefore, Communication among those devices need computational powers and strong network performances. Contrary, some of the edge devices have limited computational power which considered a huge bottleneck [19]. Since these applications required the computation power adjacent to the location. Thus, Edge computing could solve the problem by solving the computation adjacent to the communication devices.

Mobile Edge Computing (MEC) is known as an emerging paradigm to provide high-bandwidth and low-latency communications in 5G networks and IoT applications [8]. MEC supports different types of downlink and uplink scenarios (e.g., video streaming and analytics), where the main objective is to take advantages of the storage or processing capacities of low-power small-cell base stations (e.g., pico-cells) within Radio Access Network (RAN) to serve the demands of mobile users in proximity [24] [6].

It is a multi-vendor technology accessible by the service providers and third parties to mitigate cloud computing challenges [20]. Since, MEC is adjacent to the users thus it is supporting applications and services with low latency, and provides high Quality of Service (QoS). Therefore, MEC is enabling the IoT solutions required computation in real-time operations at the network edge [18].

Recently, a number of cloud solution providers, such as AWS and Azure, has introduced edge computing solutions for data streaming and analysis purposes at the network edge. The AWS and Azure platforms name their edge computing solutions Greengrass and Azure IoT Edge, respectively. These service providers have a similar architecture and technology for their edge computing solution [1].

This report is organized as follows. Section 2 presents edge computing and its importance. Section 3 provides an overview of the Azure cloud computing platform. Section 4 describes Azure IoT Edge and its different components. Section 5 elaborates more about IoT edge architectures and a comparative overview of AWS. Section 6 discusses the GPSEEDGE use case. Finally, this report is concluded with the research outcome and futures challenges in section 7.

2 What is Edge Computing

The edge computing concept is not new. Content Delivery Network(SDN) brings the content as close as possible to the users to improve the QoS and Quality of Experience (QoE). Previously, the cloudlet and fog computing technique introduced by the researchers for the data produced at the network edge. Those edge produced data processing in the cloud might not be an efficient solution [21]. As a result, researchers proposed several edge computing methods and techniques to solve the edge data processing challenges.

Since data producing at the network edge, therefore, processing and computing those data at the network edge would be an efficient use of the network. Edge computing defines as a technology enabler, which allowed to perform computation at the network edge. According to Weisong Shi et al., [21] " edge as any computing and network resources along the path between data sources and cloud data centers". Therefore, edge com-

puting becoming the ultimate solution to mitigate the bottlenecks of cloud computing.

2.1 Advantages of Edge computing

There are numerous positive reasons to use edge computing technologies and methods. Following are some reasons discussed briefly to demonstrate the importance of edge computing.

- **Offloading tasks from the cloud:** Cloud computing has been considering an efficient way of processing data since its computational power higher. As mentioned earlier, some applications are required computation at the network edge to make rapid decisions for the services, e.g., self-drive car. Therefore, the round trips of the process data could be hindering its decision-making process since the network bandwidth has not increased [22]. Data processing at the network edge offloads the computational task from the cloud. Later edge device sends the process data to the cloud for further use.
- **Data pulled from the IoT devices:** Almost all the electronic devices will become an IoT data providers and data consumers, e.g., smart city traffic lights, internet-connected fridges. The sheer volumes of raw data generated from those IoT devices could not efficiently handle by the Conventional cloud computing [21] [22]. Thus, most of the data produced from the IoT devices will not transmit to the cloud due to efficient use of bandwidth, security and optimum use of device energy consumption.
- **Role change from a data consumer to data producer:** Over the period evolution of the technologies users' role has also changed from a data consumer to a data producer. For example, mobile phone users consume data from Youtube, Facebook or Instagram but it also contributes as data producer by producing data e.g., photo and video upload, social networking. Thus, edge computing required functionalities to process those data locally before sending to the cloud [9].
- **Proximity to users:** 5G network gives MEC utmost proximity to the users mostly one or two network hops away. As a result, it significantly reduces the end to end latency and alleviates congestion on the core network [6].

3 Azure platform

Microsoft cloud computing business called Azure cloud computing. It is a collective cloud computing service that includes infrastructure as a service (IaaS), platform as a service (PaaS), and software as a service (SaaS) [12]. Azure IaaS is a virtual server in the cloud that has control by the users. Azure data-centers distributed around different continents to provide services globally. Azure IaaS, virtualized servers controlled by the Azure Fabric Controller [23]. There are numerous advantages of using IaaS. For example, it eradicates capital expenses optimizes ongoing cost, enhanced business continuity disaster recovery, rapid innovation, quick response to the business condition, boosting stability, reliability, and supportability, and Superior security[2].

Azure PaaS is the deployment and development environment for web-based applications to provide 360-degree lifecycle support [25]. It includes building, testing, deploying, managing and updating within Azure cloud platform[17]. PaaS brings the benefits of time efficiency, application lifecycle, multi-platform support, geo-distributed development, and cost optimization[17].

SaaS is the application based solution, provided to the consumers, that functioning in a cloud environment. The user of SaaS-based application, use the application over the internet and they do not need to maintain any of the required infrastructures[14]. Office 365 and One Drive is some example of an application that Azure providing as SaaS. Azure SaaS also bring numerous advantages for its users such as eliminate hardware and software implementation and maintenance, Pay-as-you-go, free to use other software, easy mobilization of the workforce, and on-demand data access [3].

4 Azure IoT Edge

Azure IoT edge builds on top of the Azure IoT Hub to develop IoT services. It brings cloud analytics and business logic to the devices level to help business to focus on the insights. Data analytic from IoT solutions derived the business values, but all the data are not equally important for the business. Thus, using the right data is the key to deriving the business values computed by the edge device locally. It could dwindle the bandwidth cost and reduce transferring a big amount of raw data. Locally

clean and aggregated data only send to the cloud.

4.1 Azure IoT Edge component:

Following are the three IoT edge computing components [13].

- **IoT Edge module:** This is a container that can run azure or third party services or developers code. Modules are installed and execute to the local Edge devices.
- **IoT Edge Runtime:** This runs on the Edge devices and maintains modules installed in the individual devices.
- **IoT Edge cloud interface:** This enables developers to manage and monitor IoT devices remotely.

IoT Edge module:

IoT edge module is the smallest execution units for IoT edge which utensil as docker consistent containers. Containers are running the business logic at the edge devices. So module considers as unite of compute, containing the developers own code, parts of services which deployed onto the edge.

There are four conceptual module elements which describe a module development, deployment, and maintenance. [16]. First, a module image is a package containing an application that defines a module. Second, a module instance working as computational units running in the module image on the IoT edge device. IoT edge runtime initiates the module instance. Third, a module identity associated with the module instance. It contains the IoT Hub information including security credentials. Fourth, a module twin is a JSON document that store in the IoT Hub. It contains the state, metadata, configuration, and conditional information of module instance. It is important to describe in more details to understand the IoT edge module better.

Module images and instances:

IoT Edge module image containing an application that taking the advantages from the IoT edge runtime provides management, security, and communication features. Module images support the developers' code, or it is also possible to export from Azure services. Azure supporting images are

available in the cloud, that can be edited, updated, and deployed for the diverse business solutions. A module image installs to a device, when it starts then it triggers a new instance of that module. However, geolocation does not matter for module images, but the individual device would have its own module instance.

Module Identities:

Module identity is a unique identity for each module instance in the IoT Hub. This identity is testable for authentication purpose in the cloud. It makes sure the securities between modules, modules and cloud, and device and modules. However, one device could have multiple module identities.

Module twin:

Module twin is dealing with state management from a device management perspective. Each module configured in a way that a module instance has a comparable module twin. The twin and instance are correlated with each other by the module identity.

IoT edge Runtime:

Azure IoT edge runtime is the heart of the system. It deploys onto the edge devices. Since edge runtime also containerized that gives freedom of movements and update facilities according to the business needs. All the modules running on the edge device managed and monitored by the edge runtime. It also maintained the security standards meaning that all the authentication required by the modules maintained by the edge runtime. Following are the features provided by the edge runtime [15].

- It deploys and updates the workload on the device.
- It is Managing IoT edge security standards on the device.
- It assures that the edge modules are consistently running.
- It checks the module health and reports to the cloud in order to monitoring remotely.
- It facilitates communication between the downstream leaf devices/modules on the IoT edge device and IoT edge device and cloud.

IoT Edge cloud interface

Managing the software life cycle for the enterprise devices is a challenging and complicated job. It becomes vigorously difficult when it deals with millions of heterogeneous IoT devices[5]. Thus creating a workload for a singular type of device and configure it according to the business demand. The configured devices deployed at scale to the millions of devices those are having similar demand [13].

Azure IoT edge and Azure IoT cloud solution are seamlessly integrated among each other to provide one tailor solution to the business needs. Following are the key characteristics of the Azure IoT edge cloud interface.

- Workload configuration and run to a singular type device
- Sending similar workload to the number of devices
- Workload monitoring on the devices

5 Azure IoT Edge computing architecture

Edge computing architecture creates a new paradigm by computation data into the network edge and leverages the bottleneck from cloud computing [7]. Similarly, Azure IoT edge compute data at the network edge by using docker compatible container lightweight virtualization. The edge module can contain Azure function, developers code, and libraries. The docker containers run as a result of the edge module in the IoT edge platform [1]. IoT edge platform is flexible, and it supports five programming languages such as Java, C, Python, C# and Node.js. Its allow the possibilities to deploy Azure services, such as functions, streaming analytic, and Azure machine learning model in the docker containers. Importantly, Azure edge module could be deployed, updated and modified either by using the Azure IoT edge cloud websites interface or Azure command line interface(CLI) [4].

All the IoT devices installed at the Edge runtime. The Edge runtime consists of three components such as IoT Edge security daemon, IoT Edge agent and IoT Edge Hubs. Firstly, the IoT Edge security daemon component starts when the IoT edge device boots and bootstraps a device. Secondly, IoT Edge agent is managing the deployments and monitoring the

modules on the IoT edge device and IoT edge Hub. Finally, the IoT edge Hub maintaining communication between modules and IoT device and between device and IoT Hub [4]. During the communication between the IoT agent and IoT Hub, it uses MQTT and AMQP protocols for messaging. Then the message is routed from the IoT Hub to user stated endpoint. The IoT Hubs assorted the incoming message for blobs storage endpoint and then it writes multiple times in an individual blobs files [1].

5.1 AWS Greengrass for edge services

Amazon is also providing edge computing solution which they name as AWS Greengrass. AWS Greengrass shares akin functionalities and architecture as compared to Azure IoT Edge. In IoT Greengrass, AWS lambda functions can run in the connected device which enables machine learning models, data synchronization, and secure communication with other devices.

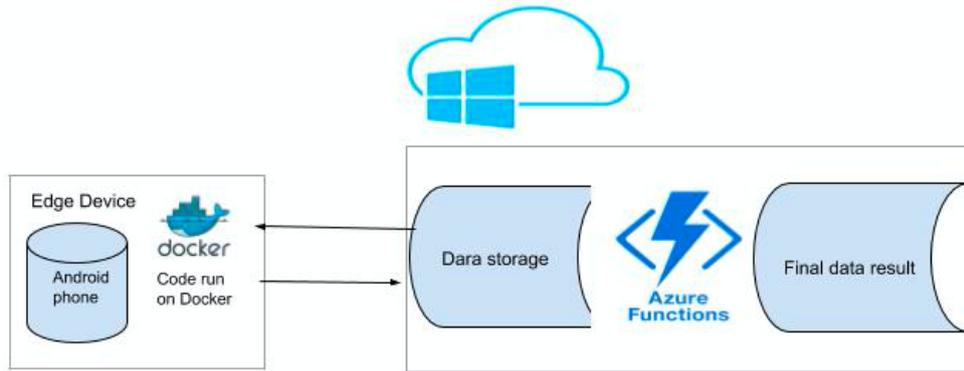
As like Azure IoT Edge, Greengrass also supports multiple programming languages for developers to develop and test their own solution. The device is running on IoT Greengrass core acting as a hub. It is responsible for the communications with other devices those installed AWS IoT device Software Development Kit (SDK). Nevertheless, Greengrass core device, IoT devices SDK could be configured to communicate among each other as a Greengrass group. If Greengrass core loses the connection with the cloud, but device can communicate among each other in the local area network [11]. Greengrass pre-build connectors enhance edge computing functionalists greatly without writing code. It enables devices to connect with IoT edge service fast. In term of security, Greengrass provides its security features along-with hardware-secured messages encryption features [11].

6 GPSEEDGE use case

The GPSEEDGE name comes from GPS and Edge. This section the author describes a simple experiment between Azure edge and cloud computing using GPS technology. The experiment aims to measure the latency difference between the Azure Edge computing vs the Azure cloud computing.

Experimental Setup

Figure 1. GPSEEDGE Architecture

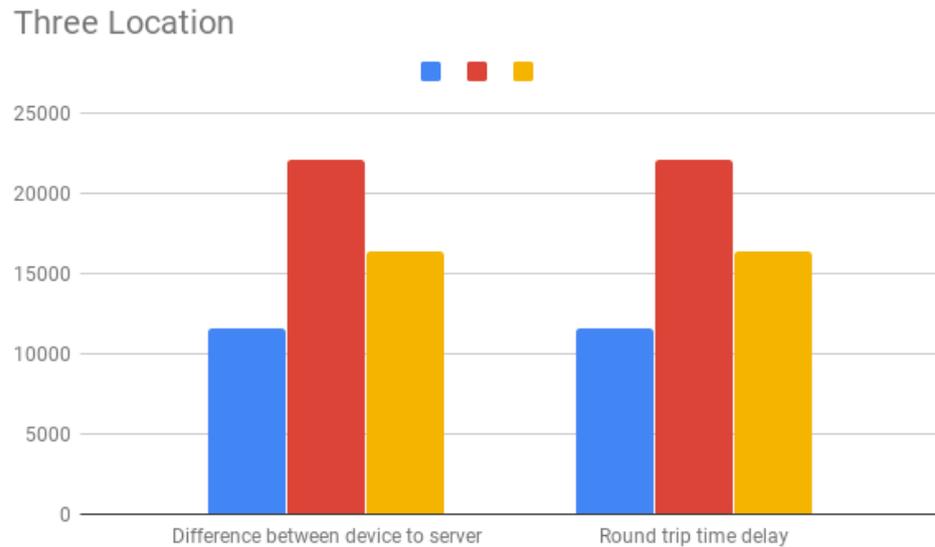


The experiment runs in a simulated computer with GPS and cloud computing functionality. This simulated computer has 8GB RAM, 2.16 GHz quad-core processor power and 64-bit windows operating system. During this experimental process Aalto university standard internet connection used for the round trip of the raw and processed data.

There was two Android apps used as an edge device. Developer codes run on the docker container on the edge devices. First, this two edge device run for a couple of minutes and calculate data from distance A to distance B then it saves the timestamp on the edge device. Second, Those raw data sent to the Azure IoT Hub for cloud computing and storage purpose. Figure 1 shows the GPSEEDGE architecture. Azure IoT Hub cloud calculate a distance from A to B from raw data by using Azure function and logs the timestamp. Last but not least, Azure IoT Hub send those process data back to Android device and log the timestamp.

The experiment runs a couple of minutes and generates various data points. The author only takes three locations data, and its data point to see the difference. Those data point log as a timestamp in the system. The output of those logs come in three different JSON files. The first data point is edge computation timestamp, the second one is Azure cloud computing server timestamps, and the third point is round trip processed data into the edge device. Calculation from these three points gives us the network latency. Network latency could find by deducting timestamps from device to server, and server to the device. For instance, a user moving with the app from distance A to B, the app calculates the distance S and logging the timestamp X. Same data send to Azure IoT Hub and calculate distance A to B and logging the timestamp Y, and finally calculated value

Figure 2. Timestamp difference between device and server (milliseconds)



send to the device and logging a new timestamp Z. The different timestamps between the devices to server X-Y and round trip timestamp X-Z gives the millisecond difference that is the network latency.

6.1 Results and Discussion

The timestamp discussed in section 6 gives us the notion of the importance of edge computing. As mentioned earlier, some of the applications required instant computation data to make a better decision. The experiment runs by the author to show that computation at the edge of the device could give a better quality of service as compared to computation in the cloud. The result shows in the figure 2 that there is a significant difference between the device, server and round trip timestamp.

The graph shows the log time differences between the server and a round trip delay to the edge devices for three locations. The timestamp difference for location one is 11556 and 11562 milliseconds from the device logs and the round trip device logs respectively. Location two and three demonstrate the same pattern. It means that edge computing could save some milliseconds to seconds as compared to cloud computing.

The delay may cause; firstly, the bandwidth because data required some time to travel from the devices to the cloud and come back to the devices. Secondly, Azure IoT Hub batches the messages received from devices. It

writes the result in batches and in a single blob file from multiple messages. It means that Azure IoT Hub holds the messages for a while before it is writing in the blobs files.

7 Conclusion

Cloud computing helps the business to scale it without much thinking about the technological scale. Since the bandwidth, equipment, and associated ICT technologies are evolving at a rapid pace, thus it triggering several new business phenomena. Those new businesses, e.g., self-driving cars, need seamless connectivity, and fast computing power on the fly. Cloud computing could be a solution for those businesses, since cloud data center are far from the actual user locations, therefore it implies some latency to the services which could lead to a fatal incidents. Nevertheless, Edge computing could leverage some of the constraints of cloud computing.

Microsoft Azure edge computing is called Azure IoT edge. Microsoft provides edge service in a combination of its two services, such as Azure IoT Edge and IoT Hub. IoT Edge provides computational capabilities to the edge device to run the business logic at the edge devices and helps to make instant decisions. Later, processed data send to the IoT Hub for storage or further process. This experiment finds significant difference between Azure IoT edge and Azure cloud in term of latency. The IoT edge computation does not have latency which helps devices to make a better decision as compare to cloud computation.

Since this paper only dealing with Azure IoT Edge platform. The future challenge would be to experiment with the same logic to AWS Greengrass to see the latency difference. It will also be a challenge to investigate how Azure and Greengrass will work with 5G technology.

References

- [1] Mike P. Wittie Anirban Das, Stacy Patterson. *Edgebench: Benchmarking edge computing platforms*, 2019.
- [2] Microsoft Azure. What is iaas? <https://azure.microsoft.com/en-us/overview/what-is-iaas/>, 2018. [Online; accessed 7th-April-2019].
- [3] Microsoft Azure. What is saas? <https://azure.microsoft.com/en-us/overview/what-is-saas/>, 2018. [Online; accessed 7th-April-2019].
- [4] Microsoft Azure. Azure iot edge. <https://azure.microsoft.com/en-us/services/iot-edge/>, 2019. [Online; accessed 7th-April-2019].
- [5] Jeff Barnes Bob Familiar. *Business in Real-Time Using Azure IoT and Cortana Intelligence Suite-Driving Your Digital Transformation*. APress, August 2018.
- [6] Intel Labs China. Mobile edge computing towards 5g: Vision, recent progress, and open challenges, 2016.
- [7] Bartolome Rubio Cristian Martin, Manuel Diaz. An edge computing architecture in the internet of things. *Proceedings - 2018 IEEE 21st International Symposium on Real-Time Computing, ISORC 2018*, pages 3; 99–102, 2018-05-31.
- [8] Pekka Kuure Uwe Rauschenbach Fabio Giust Dario Sabella, Alessandro Vailant. Mobile-edge computing architecture: The role of mec in the internet of things., October 2016.
- [9] Manish Parashar Eduard Renart, Javier Diaz-Montes. Data-driven stream processing at the edge, 2017.
- [10] freeCodeCamp.org. Beyond the Cloud: Edge Computing. <https://www.youtube.com/watch?v> 2018. [Online; accessed 7th-April-2019].
- [11] AWS Greengrass. Aws iot greengrass, bring local compute, messaging, data caching, sync, and ml inference capabilities to edge devices. <https://aws.amazon.com/greengrass/> 2019. [Online; accessed 7th-April-2019].
- [12] Robert Greiner. Windows Azure IaaS vs. PaaS vs. SaaS. <http://robertgreiner.com/2014/03/windows-azure-iaas-paas-saas-overview/>, 2014. [Online; accessed 7th-April-2019].
- [13] Kelly Gremban. What is azure iot edge. <https://docs.microsoft.com/en-us/azure/iot-edge/about-iot-edge>, 2019. [Online; accessed 7th-April-2019].
- [14] Erkkka Honkavaara. Implementing enterprise application integration with cloud services; järjestelmäintegraation toteuttaminen pilvipalveluiden avulla. G2 pro gradu, diplomityö, 2017-04-03.
- [15] Chrissie Theano Petersen Shilpa Sharma Karim Oumghar mrohera Robin Shahan Nick Schonning Ben New Kelly Gremban, Beth Harvey. Understand the azure iot edge runtime and its architecture. <https://docs.microsoft.com/en-us/azure/iot-edge/iot-edge-runtime>, 2019. [Online; accessed 7th-April-2019].
- [16] mrohera Robin Shahan Chipalo Street Kelly Gremban, Karim Oumghar. Understand azure iot edge modules. <https://docs.microsoft.com/en-us/azure/iot-edge/iot-edge-modules>, 2019. [Online; accessed 7th-April-2019].

- [17] Alexandre Painchaud. What is microsoft azure platform as a service (paas)? <https://www.sherweb.com/blog/what-is-azure-paas/>, 2018. [Online; accessed 7th-April-2019].
- [18] Soumya Kanti Datta Peter Corcoran. Mobile-edge computing and the internet of things for consumers, October 2016.
- [19] Mario; Taleb Tarik Premsankar, Gopika; Di Francesco. Edge computing for the internet of things. 2018.
- [20] Konstantinos; Mada Badr; Flinck Hannu; Dutta Sunny; Sabella Dario Taleb, Tarik; Samdanis. On multi-access edge computing. pages 25; 1657–1681, 2017-07-01.
- [21] IEEE Jie Cao Student Member IEEE Quan Zhang Student Member IEEE Youhuizi Li Weisong Shi, Fellow and Lanyu Xu. Edge computing: Vision and challenges, 2016-10-01.
- [22] Schahram Dustdar Weisong Shi. The promise of edge computing, 2016-05-01.
- [23] Jian Chen Yuchen Zhou, Fei Richard Yu and Yonghong Kuo. *Communications, Caching, and Computing for Next Generation HetNets*. IEEE Wireless Communications, August 2018.
- [24] Dario Sabella Nurit Sprecher Yun Chao Hu, Milan Patel and Valerie Young. Mobile edge computing a key technology towards 5g, 2015.
- [25] Li Zhuo. Design of online vocal music course based on azure-paas platform, 2017.

Bluetooth LE Locator Security and Privacy

Ngadhnjim Plaku

ngadhnjim.plaku@aalto.fi

Tutor: Tuomas Aura

Abstract

Bluetooth Low Energy (BLE) has emerged as a technology that enables devices to broadcast information that shows their presence in an area. The transmitted signals are received by other devices in the vicinity, which use such information to offer services with location awareness. Since these signals can be easily captured by monitoring devices, as the number of devices that use the technology increases, tracking by adversaries becomes a real threat. Although the specification security features can be implemented in typical BLE systems, the design and implementation of most available protocols do not make use of the cryptographic capabilities. This paper presents an analysis of BLE security, found vulnerabilities, possible attacks that exploit them and their privacy implications.

KEYWORDS: *Bluetooth, BLE, locators, security, privacy*

1 Introduction

Bluetooth Low Energy (BLE) is a wireless communication technology designed as a low-power solution for several applications. This technology has seen increasing usage for proximity-based user engagement and other applications such as tracking lost items, theft detection, proximity-based authentication, and indoor navigation [19]. With the widespread of the technology, there is an increasing need to evaluate its security properties and the privacy of users.

Among the most common usages of the technology are location tracking and proximity detection applications. The most usual implementation is through beacons that are connected to apps, which operate based on prior information about the locations of beacons. The apps are installed on the users' mobile devices and detect the beacons. Figure 1 shows Tile [4], which is an example of such BLE locators. It can be attached to items that are easily lost or forgotten, such as keys or wallet. The locator frequently advertises information, which is picked up from the application whenever the connected mobile phone is in the range of the signal. The application is able to locate the position of the item based on the signal strength and the location of the phone. Therefore, the user can find out the current location of the item, when the phone is within close proximity to the locator, or the location where it was the last time the phone received a signal.



Figure 1. Tile tracker attached to keys [4]

While BLE locators help the owners locate their items in situations such as when they are lost or misplaced, on the other side these devices also bring new security and privacy issues. These issues arise from the implementation of the locators or the mobile application on the user's phone. The data transmitted by the locators might reveal information that violates the privacy of the user if it is connected to the user's personal in-

formation [13]. Potential misuse of this information is tracking the user's location and behavior.

Since the BLE device advertises its presence, an unauthorized and potentially malicious party can use this revealing of the device's presence for more serious privacy and security attacks. Additionally, the technology was designed to be used on devices with low computational capabilities and small batteries. As a result, the protocol was simplified, and there are side effects regarding the privacy of data transmitted over BLE [17]. Some of these issues were avoided in recent specifications, which contain new privacy features with the addition of randomized addresses and other security enhancements [6].

Our goal is to analyze the security and privacy issues that are related to BLE locators. We will give an overview of the technology regarding these issues by reviewing the literature and previous research. This paper is organized as follows. Section 2 introduces the BLE technology and its security features. Section 3 analyses the possible passive attacks that can be used for location and presence tracking. Section 4 continues with the active attacks used to detect the location and proximity of such devices. Section 5 presents a discussion of additional issues regarding the security and privacy of BLE locators. Finally, section 6 summarizes the paper and presents the conclusion.

2 Bluetooth Low Energy

2.1 Overview

Bluetooth Low Energy operates in the unlicensed ISM band at 2.4 GHz [16]. The band is separated into 40 RF channels. Three of these channels are used for initial advertising, while the other 37 are used for communication between devices. The devices that want to establish a communication must use the same physical channel. This is done by transmitting on the same RF channel. To mitigate the effects of a collision, transmissions start with an Access Address.

The packets are formatted as shown in Figure 2. This format is used for both the advertising and data channels. The packet starts with a preamble, which is used by the receiver to synchronize connection parameters. Then follows the Access Address, which is four bytes long. After that

comes the Protocol Data Unit (PDU), which has a variable length and contains the data transmitted by the advertising or data channel. At the end, there is a 24-bit cyclic redundancy check (CRC) calculated from the PDU.

Preamble (1 or 2 bytes)	Access Address (4 bytes)	PDU (2 to 257 bytes)	CRC (3 bytes)
-----------------------------------	------------------------------------	--------------------------------	-------------------------

Figure 2. BLE frame format

For interoperability between applications, Bluetooth uses profiles [16]. Profiles define the interactions between different layers or between peers on the same layer. BLE makes use of two profiles: the Generic Access Profile (GAP) and the Generic Attribute Profile (GATT). GAP defines how devices discover each other, establish a connection, and the security models used. It defines roles that are present in each device. In LE, there are four specific roles: Broadcaster, Observer, Peripheral, and Central. GATT takes over after the connection is established. It provides a framework for operations and data format, and It defines the roles for a client-server relation.

2.2 BLE Security

According to Bluetooth specification [16], there are five keys used:

1. Identity Resolving Key (IRK) used for resolving addresses.
2. Connection Signature Resolving Key (CSRK) used for signing and validating signatures.
3. Long Term Key (LTK) used to encrypt reconnections.
4. Encrypted Diversifier (EDIV) used for establishing shared LTK and starting communication.
5. Random Number (Rand) used with EDIV during reconnection.

The specification defines some native security features [5]. It implements a frequency hopping mechanism for switching between frequency channels. Additionally, it has control over discovery mode and also implements tracking prevention mechanism. This is achieved by changing

the address so that, even if the address is detected, the device cannot be identified by other devices that are not paired with it.

BLE security differs from traditional Bluetooth. BLE pairing uses a Long-Term Key (LTK) instead of a Link Key, which serves the same function but is established in a different way [15]. In BLE Secure Connections, the key is generated at each device as the result of a secure cryptographic key agreement. LE also introduced private addresses and signing of data. CSRK is used for data signing, which ensures integrity and authentication.

2.3 Use Cases

Some of the current use cases of BLE devices for location purposes include:

- Indoor navigation where GPS cannot be effectively used.
- An additional layer of identification using proximity to a specific location.
- Tracking belongings which are small, and often lost or misplaced.
- Theft detection alarm.
- Asset tracking is used for internal stock management or to deliver detailed location information for customers.
- Location awareness is used to track the presence of a user in a specific location and push relevant information to the user.

2.4 Beacons

BLE beacons are devices that use BLE technology to transmit signals that can be identified by other devices. Google has also developed an open beacon format called Eddystone [2]. Similarly, Apple has developed iBeacon [13]. iBeacon is the most widespread implementation of BLE beacons, and it is supported also by Android and other operating systems. Its aim is to create location awareness for other devices. iBeacon only sends its

identification number and does not communicate with the other devices. However, the other devices can determine their location by using the information broadcasted by the beacon, the signal strength and prior knowledge of the beacon's location. An iBeacon advertisement provides three values that serve as identifying information for the iBeacon. These values are the UUID, major and minor. This information is hierarchical, i.e., the major and minor fields allow subdividing of the identification provided by the UUID.

3 Location and Presence Tracking - Passive Attacks

Among the main security issues that enable an adversary to track the location and presence of a BLE locator are eavesdropping and associating a device with its user for identity tracking [1]. The following is a brief description of these two issues.

3.1 Passive Eavesdropping

Passive eavesdropping refers to capturing of the data exchanged between two devices during their communication. The legacy pairing (BLE 4.1 or older) tries to overcome this by encrypting the data using AES-CCM, which is considered a secure algorithm. However, the key exchange protocols have security vulnerabilities that allow the decryption of data by an attacker. This way, the pairing method, which includes the key exchange, has serious security implications [1].

Legacy pairing does not offer passive eavesdropping protection, and this vulnerability exists in all of the pairing methods [15]. Therefore, it allows an adversary to obtain the pairing messages and retrieve exchanged keys [7].

On the other hand, BLE versions 4.2 and later are considered secure against passive eavesdropping because they make use of BLE Secure Connections for key generation [16]. The master key is generated with the Elliptic Curve Diffie-Hellman (ECDH) algorithm, and it is then used to generate other keys across the encrypted channel [2].

Sniffing can be used to capture a specific beacon's identifier. This enables the adversary to track the device. It can also be used for taking advantage of an existing beacon network [19]. This kind of attack is known as free riding, and the attacker makes use of an existing beacon infras-

structure for its own services and navigation. The attacker can also use the existing infrastructure as a tool for tracking users and inferring their habits. This attack can be prevented by switching IDs of the beacons, so that only the owner would know the respective decoding to make use of the network.

Ubertooth is an open source wireless development platform that supports BLE communication sniffing, since it is capable of capturing and monitoring the transmitted BLE packets [12]. A sniffer capable of maintaining BLE connections even during their hops across channels and with a pre-established connection was implemented by [17].

3.2 Identity Tracking

Identity tracking refers to the association between the address of a BLE device and a specific user. It is possible that an adversary has placed malicious devices to receive these messages and tracks the time, location and the locator's address. The privacy of users can be at risk if the owners of the locators are known. The adversary can use this to track the BLE device and assume that the user is also present at the same location [15].

If the device is assigned a public address, the address will be visible during the discovery process. As long as the device remains discoverable, the attacker can detect the presence of the specific device and track its location.

BLE supports a privacy mechanism, known as private addresses, to prevent the tracking of users or devices [16]. This mechanism allows a device to periodically change its address [7]. An adversary is not able to determine that the addresses relate to the same physical device, and thus cannot track the user based on the address.

During the pairing of the devices, they generate various encryption keys, one of which is the IRK. BLE devices use the IRK to generate and map the Resolvable Private Address (RPA) to an Identity Address. The advertising BLE device can be tracked by devices that have the advertiser's IRK. The IRK generated at the time of pairing is stored by both devices in their memory. This enables the previously connected devices to resolve the private address and translate them to the real ones even though the RPA changes frequently [16].

Despite the existence of this mechanism, privacy-preserving MAC addresses are rarely used in practical implementations [8]. As a result, a large number of devices are still vulnerable to tracking based on their

static addresses. An adversary can deploy sniffers in strategic locations or even use compromised devices, such as the BLE Botnet, to achieve large scale, high precision tracking.

4 Location and Proximity Detection - Active Attacks

An active attack is an exploit that includes the ability of the attacker to change the data on its way between the legitimate parties. This section describes some possible active attacks possible on BLE locators. The attacks discussed here are location spoofing, Man in the Middle attack, and Denial of Service.

4.1 Location Spoofing

Spoofing refers to an attack through which the attacker masquerades as another person or device.

In some implementation, such as the iBeacon, the UUID together with major and minor values identify the beacon. In such situations, even if randomized private addresses are used, since the identification is in the payload, such measures would be pointless. Capturing the beacon's identification using sniffing tools allows an attacker to impersonate that beacon anywhere, causing confusion and breaking assumptions related to the location of the beacon [19].

Another possible attack is Beacon Silencing. This attack manipulates devices into perceiving a locator as being far away, while it actually is within its proximity. To achieve such an attack, the adversary transmits a large number of spoofed beacons. The attack sets a greater Measured Power value on these signals. This way, the estimated proximity of the locator will be biased towards the fake information, resulting in inaccurate estimation of the distance from the locator [10].

Other spoofing attacks against systems based on BLE beacons have been evaluated on [14]. The attack scenarios considered include the attacker with physical access to deployed beacons and also the attacker with remote control over them.

4.2 Man in the Middle Attack

Man in the Middle (MitM) attacks refer to attacks in which the adversary is positioned in between two devices during their communication [4]. The

malicious device controlled by the attacker impersonates the other two devices and establishes a connection with them [1]. This device now has the possibility of transferring communication between the two legitimate devices, making them believe that they are communicating with each other. This allows the adversary to intercept the communication and also to inject or drop data packets. The common MitM attack does not work for BLE due to the Bluetooth technology of being connected to only one party at a time [11]. A BLE MitM needs to use two BLE devices, and these devices should be capable of communicating together in some other way, in order to connect both ends of communication [11]. An implementation for BLE MitM is GATTacker [9]. GATTacker can scan and copy BLE communication providing information for creating fake versions of the legitimate devices [11].

In the case of BLE locators, such an attack can be used by an adversary to send false information to the locator or its associated mobile application. In such a situation, both entities would assume that the messages are coming from the legitimate devices. The attacker can drop messages that are intended to alert a thief detection. Another aim of the attacker can be causing confusion. This can be done by sending false signals to the mobile app, making it think that the locator is somewhere else. The attacker also has the ability to annoy the user by triggering false alarms.

Even though such an attack is technically possible, most of the current scenarios where it can be used to make it unfeasible. In order to protect against MitM attacks, we have to ensure the identity of the devices in the communication through means of authentication, which can be done by using the CSRK key. CSRK is used to generate the Message Authentication Code (MAC), which serves as a signature of the sending device [16]. The signature also contains a counter to protect against replay attacks. It is placed after the Data PDU and the receiver verifies it using the pre-shared CSRK. After signature verification, the Data PDU is assumed to come from the trusted source.

4.3 Denial of Service Attack

Denial of Service refers to an attack that makes the device's Bluetooth not operational [19]. One effect can be draining the battery of the device. These attacks require proximity because of the Bluetooth's operation and can be avoided by moving out of the attacker's range. Still, it can cause service loss and customer dissatisfaction. This attack can be used

against a competitor to make their beacon system unusable. If the system is used for navigation, the attack can cause orientation problems for the customers, and depending on the situation it might have greater severity. In case such a system was deployed in an airport, the delay caused by the attack might make the users miss the plane.

On the other hand, in proximity authentication scenarios, the attack can prevent accessibility of other resources that rely on the locator for authentication. In the asset tracking and theft detection scenarios, the attack can have a large impact on user satisfaction with the product.

5 Other threats

Except for the issues in BLE implementation, there are also security and privacy issues related to the mobile application and its communication with the cloud. According to research results released by security firm Rapid7 [3], many BLE locators have such vulnerabilities. Rapid7 examined products from many manufacturers, and they discovered some common vulnerabilities. Most of the mobile applications associated with the locators had a cloud API, and in many situations, the password used for cloud authentication was stored in cleartext. The device's tracking ID was easily obtainable using a BLE scanner. Figure 3 shows the reading of this ID using nRF Scanner. The access to the cloud service used for querying or sending GPS data was in many implementations without authentication. This way, an adversary can access GPS data for any device from a web browser by using the tracking ID. Some products also allowed unauthenticated Bluetooth pairing. This way applications were able to connect and write data to attributes, such as enabling the device's alarm, causing annoyance for the legitimate user and draining the device's battery.

BTLEJACKING [1] is another tool that was presented at DefCon. It allows the attacker to sniff, jam and takeover any BLE device.

Another potential attack can happen on the mobile device itself. The Bluetooth specification restricts access to BLE data via pairing and bonding. These mechanisms are intended for establishing an authenticated connection between two devices. However, mobile devices have many applications and there is potential that a malicious app abuses an established connection by an authorized application and this way leak information [18].

The main challenge for large scale tracking is the difficulty of construct-

A checklist of security measures for BLE implementations, that can help preventing most of these attacks, is provided by NIST at [15].

6 Conclusion

BLE is emerging as proximity and low energy locator technology. However, it accompanies a set of privacy risks. Bluetooth Low Energy with the release of version 4.2 has significantly improved the security [11]. The new Secure Connections pairing model including ECDH key exchange and numeric comparison method are used to ensure privacy and data security [16].

The advantages of BLE locators are obvious in everyday life and since such devices are usually used on application with non-critical nature, these advantage in most of the situations overcome their privacy and security risks. The manufacturers are recommended to make use of all the security features designed in the latest specification to provide the best protection for their product.

References

- [1] Btlejacking, 2019. <https://github.com/virtuallabs/btlejack> Accessed 7-April-2019.
- [2] Eddystone, 2019. <https://developers.google.com/beacons/eddystone> Accessed 7-April-2019.
- [3] Rapid7, 2019. <https://blog.rapid7.com/2016/10/25/multiple-bluetooth-low-energy-ble-tracker-vulnerabilities/> Accessed 7-April-2019.
- [4] The tile app, 2019. <https://www.thetileapp.com/en-us/> Accessed 7-April-2019.
- [5] Smart Card Alliance. Bluetooth Low Energy (BLE) 101: A Technology Primer with Example Use Cases. *Mobile & NFC Council, SAD*, pages 21–22, 2014.
- [6] Kassem Fawaz, Kyu-Han Kim, and Kang G Shin. Protecting privacy of BLE device users. In *25th USENIX Security Symposium (USENIX Security 16)*, pages=1205–1221, year=2016.
- [7] Carles Gomez, Joaquim Oller, and Josep Paradells. Overview and evaluation of Bluetooth Low Energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753, 2012.
- [8] Taher Issoufaly and Pierre Ugo Tournoux. BLEB: Bluetooth Low Energy Botnet for large scale individual tracking. In *2017 1st International Conference on Next Generation Computing Applications (NextComp)*, pages 115–120. IEEE, 2017.
- [9] Sławomir Jasek. Gattacking Bluetooth smart devices. In *Black Hat USA Conference*, 2016.
- [10] Constantinos Koliass, Lucas Copi, Fengwei Zhang, and Angelos Stavrou. Breaking BLE beacons for fun but mostly profit. In *Proceedings of the 10th European Workshop on Systems Security*, page 4. ACM, 2017.
- [11] Tal Melamed. An active man-in-the-middle attack on Bluetooth smart devices. *Safety and Security Studies*, page 15, 2018.
- [12] Florina Mendoza, Lucía Alonso, Andrés López, Daniel Patricia Arias Cabarcos, et al. Assessment of fitness tracker security: A case of study. In *Multi-disciplinary Digital Publishing Institute Proceedings*, volume 2, page 1235, 2018.
- [13] Nic Newman. Apple iBeacon technology briefing. *Journal of Direct, Data and Digital Marketing Practice*, 15(3):222–225, 2014.
- [14] William Oliff, Avgoustinos Filippoupolitis, and George Loukas. Evaluating the impact of malicious spoofing attacks on Bluetooth low energy based occupancy detection systems. In *2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA)*, pages 379–385. IEEE, 2017.

- [15] J Padgette, J Bahr, M Batra, M Holtmann, R Smithbey, L Chen, and K Scarfone. Guide to Bluetooth security: recommendations of the National Institute of Standards and Technology. National Institute of Standards and Technology (NIST), US Department of Commerce, Special Publication 800-121 Revision 2, May 2017, 2014.
- [16] Bluetooth SIG Proprietary. Bluetooth Core Specification v5.0. *Covered Core Package version*, 2016.
- [17] Mike Ryan et al. Bluetooth: With Low Energy Comes Low Security. *WOOT*, 13:4–4, 2013.
- [18] Pallavi Sivakumaran and Jorge Blasco. Attacks Against BLE Devices by Co-located Mobile Applications. *CoRR*, abs/1808.03778, 2018.
- [19] Hui Jun Tay, Jiaqi Tan, and Priya Narasimhan. A survey of security vulnerabilities in Bluetooth low energy beacons. *Parallel Data Lab., Carnegie Mellon Univ., Pittsburgh, PA, USA, Rep. CMU-PDL-16-109*, 2016.