Aalto University

Network Security: Goals of authenticated key exchange

Tuomas Aura

CS-E4300 Network security
Aalto University

Purpose of key exchange

- With public keys:
  - A and B each have public-private key pairs and certificates
  - Goal: generate a symmetric shared secret session key
  - Public keys are used for the key exchange. Session keys are used for efficient protection session data (symmetric encryption and MAC or AE)
- With a shared master secret:
  - A and B share a secret master key, e.g., 128-bit random number
  - Goal: generate a shared session key for short-term use
  - Motivation: compromise of a session key is quite likely; the seldom-used master key can be better protected, e.g., SIM
- The master key and certificates (or the CA) are called roots of trust

Basic security goals

- Create a good session key:
  - Secret i.e. known only to the intended participants
  - Fresh i.e. never seen or used before
  - Separation short-term secrets and long-term security: compromise of session keys does not endanger future authentication or secrecy
- Authentication:
  - Mutual = two-directional authentication: each party knows who it shares the session key with
  - Sometimes only one-way = unidirectional authentication

Other common security properties

- Perfect forward secrecy (PFS)
  - Compromise of long-term secrets today should not compromise old session data
  - Typically achieved with empheral Diffie-Helmann
  - Can also be implemented with public-key encryption by creating a fresh key pair and then throwing it away

Other common security properties

- Entity authentication: each (or one) participant knows that the other is online and participated in the protocol
- Key confirmation: each (or one) participant knows that the other knows the session key (implies entity authentication)
  - Receives proof vs. trusts the other participant

A knows SK.

B knows SK. B knows that A knows SK.

A knows that B knows SK. A knows that B knows that A knows SK.

…

But common knowledge is not possible in a distributed system.

Correspondence properties

- Correspondence properties (or consistency): agreement between the states and beliefs of the two endpoints, or between the endpoints' initial intentions and final states
  - More precise definition of authentication and key confirmation
  - Example:
    If responder B accepts the session key K for communication with initiator A, then A has previously created the key K for communication with B

Other common security properties

- Contributory key exchange: both endpoints contribute randomness to the session key; neither can decide the key alone
  - Key distribution where one party decides the key; common in broadcast and sometimes in asynchronous communication
- Algorithm agility: support for negotiating, upgrading and deprecating algorithms
  - Downgrading protection: Endpoints negotiate the best algorithms and latest protocol version supported by both, and the attacker cannot manipulate the process (never absolute protection)

Privacy and identity issues

- Identity protection
  - Unauthenticated Diffie-Hellman first; then encrypt the identities and certificates
  - Passive sniffer cannot learn the identities of the protocol participants
  - Usually only one side can have identity protection against active attacks: one side must reveal its identity first, making its identity vulnerable to active attacks

Would you give stronger identity protection to the initiator or responder?

Privacy and identity issues

- Non-repudiation
  – Evidence preserved, so that a participant cannot later deny taking part in the protocol (usually not an explicit goal)
- Plausible deniability
  – No evidence left of taking part (usually not an explicit goal either)

DoS resistance

- Various denial-of-service resistance requirements:
  - The protocol cannot be used to exhaust memory or CPU of the participants
  - Not easy to spoof packets that prevent others from completing a key exchange (especially off-route attackers)
  - When an on-route MitM attacker stops dropping and breaking messages, the protocol recovers
  - The protocol cannot be used to flood third parties with data or to amplify DDoS attacks
- DoS protection is never absolute

Authenticated DH properties

- Signed Diffie-Hellman with nonces and key confirmation:

1. A → B: A, B, N_A, g, p, g^x, S_A(“Msg1”, A, B, N_A, g, p, g^x), $Cert_A$
2. B → A: A, B, N_B, g^y, S_B(“Msg2”, A, B, N_B, g^y), $Cert_B$,
   MAC_{SK}(A, B, “Responder done.”)
3. A → B: A, B, MAC_{SK}(A, B, “Initiator do

$SK = h(N_A, N_B, g^{xy})$

Which security properties?

- Secret, fresh session key
- Mutual or one-way authentication
- Entity authentication, key confirmation
- Perfect forward secrecy (PFS)
- Contributory key exchange
- Downgrading protection
- Identity protection
- Non-repudiation
- Plausible deniability
- DoS resistance

What is a protocol flaw?

- Poorly understood security requirements
- Limitations on the applicability of the protocol:
  - Is the protocol used for a new purpose or in a new environment?
  - Historical examples: insider attacks, multiple parallel executions
  - Timely example: distributed cloud implementation
- Unwritten expectations for implementations
  - Encryption in old specs is assumed to protect integrity
  - Authenticated messages should include type tags
- New attacks and security requirements arise over time:
  - DoS amplification, PFS, identity protection

Notes on protocol engineering

- Security is just one requirement for network protocols
  - Cost, implementation complexity, performance, deployability, code reuse, time to market etc. may override some security properties
- Security protocol engineering requires experienced experts and peer scrutiny
  - Reuse well-understood solutions like TLS; avoid designing your own
  - Only use strong security solutions (privacy and DoS protection are never strong, though)
- The most difficult part is understanding the problem
  - Must understand both security and the application domain
  - When the security requirements are well understood, potential solutions often become obvious