Access control models and policies

Tuomas Aura
CS-C3130 Information security

Aalto University 2023

Outline

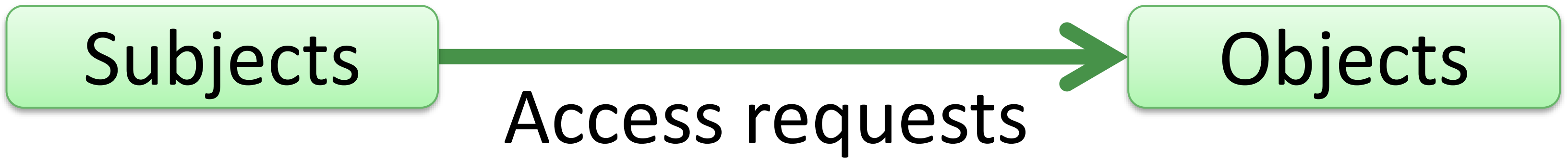
1. Access control
2. Discretionary AC
3. Mandatory AC
4. Role-based AC
5. Other AC models and variations

Models and terminology for thinking about security policies – in my experience, very useful abstractions

ACCESS CONTROL (AC)

Access control concepts

- Subjects request actions on objects



  - Alice wants to read file.txt
  - Bob wants to update account balance
  - Process wants to open a socket

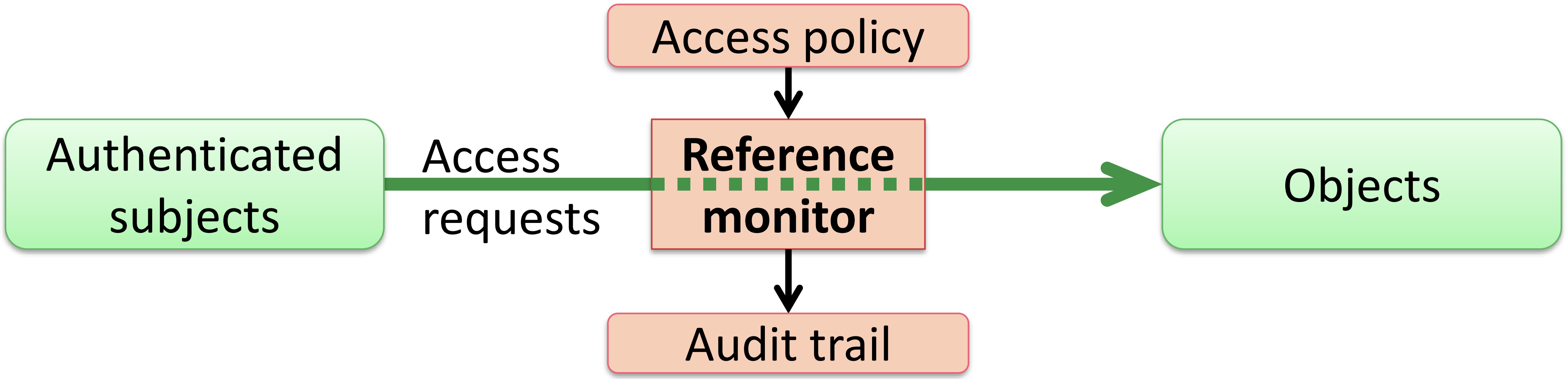
Access control (AC)

- Traditional definition: AC = authentication + authorization

- Authentication = verifying subject identity
  (implies identification of the subject)
- Authorization = checking that the subject has the right to request the action on the subject

- Authorization without identification and authentication?

Reference monitor



- Reference monitor controls access by subjects to objects
  – Grants i.e. allows or denies access requests
  – Follows the policy (i.e. rules) set by administrators
  – Logs events to audit trail

The focus for the rest of this lecture is the AC policy, and the abstract models for defining AC policies

Trusted computing base

- Trusted computing base (TCB) = all system components that need to be trusted to implement access control
  - Includes hardware, firmware, OS, other software
  - Security kernel = implementation of reference monitor in an OS
  - Isolation of subjects from each other is needed to prevent them from circumventing the AC policy

Access control matrix

- Access control matrix is the simplest, most general AC model

  $M : Subjects \times Objects \rightarrow P(Actions)$

- Subject S is allowed to request action A on object O iff $A \in M(S,O)$

	file1.txt	file2.txt	file3.txt
Alice	read, write	write	-
Bob	read	read	-
Carol	read, write	-	-
David	append		open, read, write, close

Why is this not practical for implementations?

- AC matrix represents the protection state of a system

Access control list (ACL)

- ACL = list of the access rights associated with an object
- ACLs are a practical way to represent the AC matrix: one column of the matrix is stored for each object
  - file1.txt ACL:
    Alice: { read, write }; Bob: { read };
    Carol: { read, write }; David: { append }.
  - file2.txt ACL:
    Alice: { write }; Bob: { read }.
  - file3.txt ACL:
    David: { open, read, write, close }.

ACL examples

ACL = list of the access rights associated with an object

- ACL examples:
  - Windows and Mac file systems, Confluence wiki, Github
  - Key-card locks, receptionist checking bookings

ACLs have many well-known applications

Capabilities

- Capability = an access right associated with the subject
- Capabilities are another way to represent the AC matrix: one row is stored for each subject
  - Alice’s capabilities:
    file1.txt: { read, write }; file2.txt: { write }.
  - Bob’s capabilities:
    file1.txt: { read }; file2.txt: { read }.
  - Carol’s capabilities:
    file1.txt: { read, write }.
  - David’s capabilities:
    file1.txt: { append }; file3.txt: { open, read, write, close }.

Capabilities

Capability = an access right associated with the subject

- Examples of capabilities:
  - Metal keys, driver's license, parking permit
  - Mobile app permission, Dropbox link, OAuth 2.0 token

Recently, real applications of capabilities have appeared in computing!

What is the point of defining such abstract concepts and models?

Why abstractions?

- What is the point of defining such abstract concepts and models?
  - Language for discussing AC models
  - Tool for understanding fundamental similarities in systems that first look quite different
  - Helps to compare the expressiveness of AC systems
  - Helps to understand fundamental limitations of AC systems

Principals

- Subjects are often ephemeral, e.g. processes; something more persistent is needed to define the access policy
- Solution: access rights are assigned to principals
- Principal is an authenticated identity, e.g. user account
- Subjects act on behalf of principals
  - Process (subject) runs as as a specific user account (principal)
  - User’s process running as admin or as normal user act on behalf of different principals and, thus, have different access rights

Dynamic protection systems

Extra material

Useful background information if you want to read classic computer security literature

- AC matrix represents the static protection state
- Dynamic protection systems are more interesting
- Subjects and AC matrix cells themselves can be objects
  - Access to them is also controlled by the matrix
- Protection state transitions:
  - Subjects may grant and remove access rights
  - Subjects may create and destroy subjects and objects
- Researchers have proposed some theoretical dynamic protection models
  - Safety question: given an initial state and defined state transitions, can subject *s* get the access right *r* to object *o*?
  - Examples: HRU model (safety undecidable), take-grant model (safety decidable)
- Dynamic protection models are not widely used or studied – even though most real systems are dynamic

DISCRETIONARY ACCESS CONTROL

Discretionary access control (DAC)

- Data owner, usually a user, sets the access rights
- Subjects can share their access rights to others
  - File owner or creator decides who can access it
  - User or process that can read a secret file can also copy and share it
- Typical in commercial and consumer systems
- Commonly implemented with ACLs or capabilities

- Note: There may be a policy against sharing, and access may be audited to detect violations, but the policy is not enforced technically

DAC outside computers

- DAC matches our everyday experience:
  - Paper documents can be copied and shared
  - Person with a key can open the door to others; door keys can be shared and copied
  - Telling your friends a secret, hoping that they do not tell it to anyone else, does not prevent them

MANDATORY ACCESS CONTROL

Mandatory access control (MAC)

- Access control is based on rules (policy) set by administrator
- AC policy is enforced by the reference monitor and cannot be changed by users
- Subjects cannot leak access rights to others, e.g.
  - User can read a secret file but cannot copy, print or email it
  - Process can download data from the Internet or write to the file system, but not both → users cannot download malware
- MAC is also called rule-based AC

Origins of MAC

- MAC originates from military policies
  - Officer can read a secret document but not make a copy or remove it from the premises
  - Intelligence officer may not be allowed to read his own reports after submitting them
  - Officer who had contact with foreign agents may lose access to classified information

MAC in commercial systems

- Isolation policies: Apps in a phone cannot communicate with each other unless permitted by a policy
- DRM: User can play the music she purchased, but cannot share
- Secure document viewers: Protected PDF files do not allow editing, printing or cut-and-paste; Snapchat supports sharing of photos for a limited time

Enforcement of MAC in distributed systems is always problematic

MAC outside computers

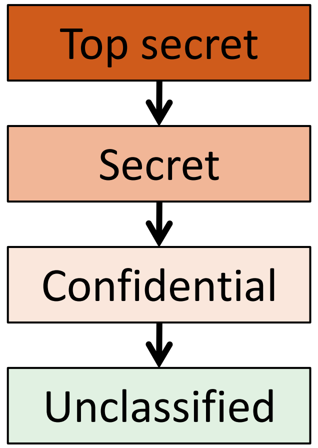
- Examples of MAC-like policies outside computers:
  - Biometric authentication prevents sharing of capabilities, e.g. photo on driver’s license, face recognition in ATMs
  - Admit-one event ticket; amusement park wristband
  - In UK, jurors must not read newspapers or watch TV about the court case to avoid external influence

Clearance and classification !

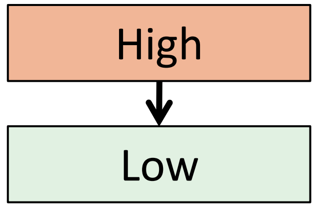
- Mandatory access control policy is often based on security labels
  - Subject clearance
  - Object classification

    ℓ : (Subjects ∪ Objects) → Labels
- Simple security property:
  S can read O iff $\ell(S) \geq \ell(O)$
- MAC based on clearance and classification levels is called multi-level security (MLS)

Real labels:



Theoretical:



Multi-level security

Extra material

- Labels depend on the organization but should form a finite lattice $\langle$ Labels, $\geq \rangle$

  i.e. a partial order with join $\vee$ and meet $\wedge$ operations

- Example: military security labels

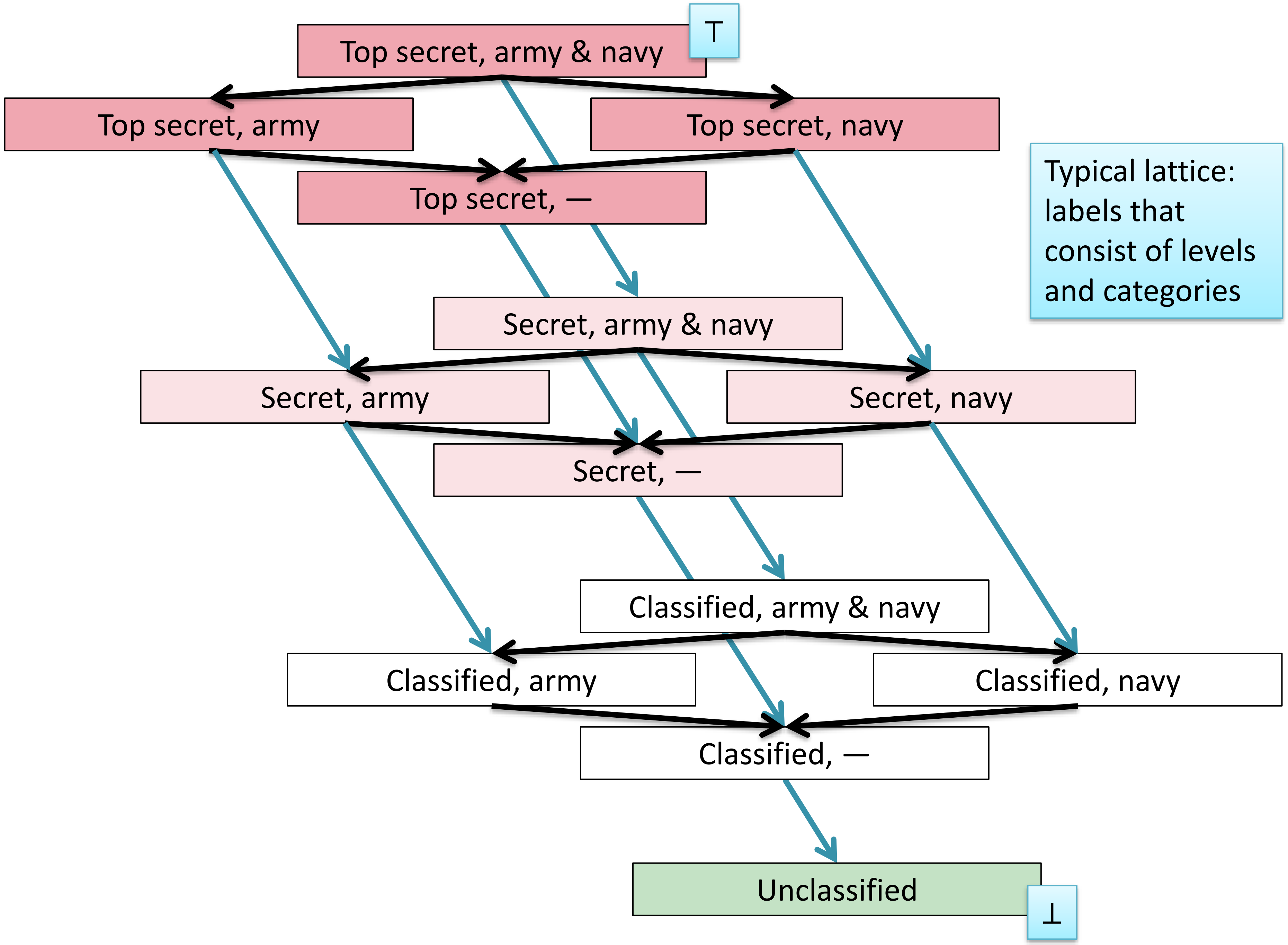
  Levels: top secret > secret > confidential > unclassified

  Categories = { army, navy, air force }

  Labels = Levels × P(Categories)

  Domination relation:
  $\langle l1,c1\rangle \geq \langle l2,c2\rangle$ iff $l1 \geq l2$ and $c1 \supseteq c2$

- How to define labels for commercial systems?



Extra material

Typical lattice: labels that consist of levels and categories

Notes on algebra

Extra material

- Basic algebraic concepts like set, matrix, partial order make it easy to define security policies precisely and to compute access-control decisions correctly
- Lattice ⟨ L, ≥ ⟩ is an algebraic structure: set with a partial order ≥ and join ∨ and meet ∧ operations
- Partial order is a relation that is:
  - reflexive: $a \geq a$
  - anti-symmetric: $a \geq b$ and $b \geq a$ imply $a = b$ and
  - transitive: $a \geq b$ and $b \geq c$ imply $a \geq c$
- Join $c = a \vee b$ is
  - an upper bound for a and b: $c \geq a$, $c \geq b$ and
  - the least upper bound: $c' \geq a$ and $c' \geq b$ imply $c' \geq c$
- Meet $c = a \wedge b$ is
  - a lower bound for a and b: $a \geq c$, $b \geq c$ and
  - the greatest lower bound: $a \geq c'$ and $b \geq c'$ imply $c \geq c'$
- When merging secret documents or making access control decisions about sets of documents, compute the join of their secrecy labels
  - e.g. secret ∨ top secret = top secret
- When merging information with integrity labels, compute the meet:
  - e.g. low integrity ∧ high integrity = low integrity

Labels in Finnish government

- Laki julkisen hallinnon tiedonhallinnasta 18 §
- Asetus asiakirjojen turvallisuusluokittelusta valtionhallinnossa,
- Julkisuuslaki 24-25 § (*freedom of information act*)

Suojaustasoluokiteltu tieto (ERSAL, SAL, LUOT, RAJ) :

KÄYTTÖ RAJOITETTU
TL IV
JulkL (621/1999) 24.1 §:n ___k
L (___/ ___) ___ §:n ___k

LUOTTAMUKSELLINEN
TL III
JulkL (621/1999) 24.1 §:n ___k
L (___/ ___) ___ §:n ___k

SALAINEN
TL II
JulkL (621/1999) 24.1 §:n ___k
L (___/ ___) ___ §:n ___k

ERITTÄIN SALAINEN
TL I
JulkL (621/1999) 24.1 §:n ___k
L (___/ ___) ___ §:n ___k

Classified information: restricted, confidential, secret, top secret

Muu julkisuuslain mukaan salassa pidettävä tieto:

SALASSA PIDETTÄVÄ
JulkL (621/1999) 24.1 §:n ________k
Muun lain (___/_____) ____ §:n ___

Other information that is confidential by law

Bell-LaPadula model !

- Bell-LaPadula (BLP) is a MAC policy for protecting secrets
  – Military security model for multi-user computers that process secrets
- Bell-LaPadula rules:

  S can read O iff $\ell(S) \geq \ell(O)$ simple security property

  S can write O iff $\ell(O) \geq \ell(S)$ *-property
- Also called: no read up, no write down

Important observation: simple security property alone does not prevent leaks by unreliable subjects (users or software)

Biba model

!

- In computer systems, integrity of data and the system is often more important than confidentiality
  - Which is more important in a bank IT system or in Sisu?
- Biba is a MAC policy for protecting integrity of data

- Biba rules:

  S can write O iff $\ell(S) \geq \ell(O)$

  S can read O iff $\ell(O) \geq \ell(S)$
- Also called: no write up, no read down

No read down is rarely implemented in real systems

Biba examples

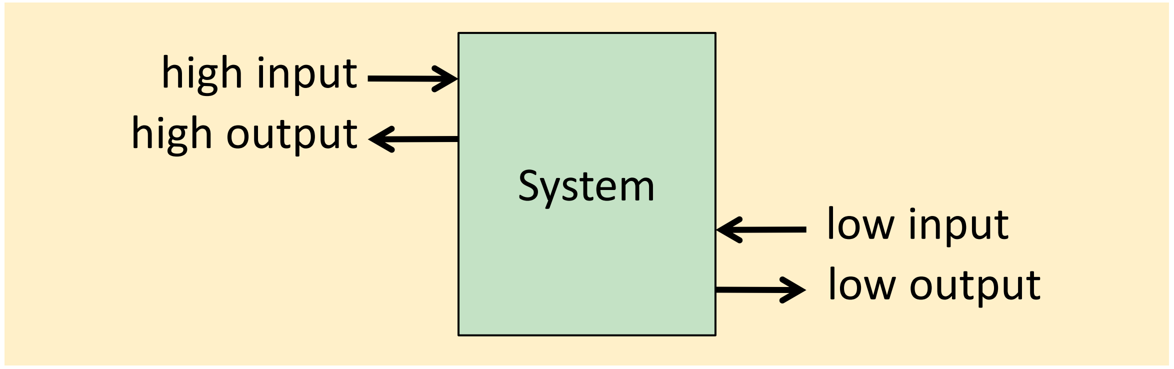
- Multi-level integrity policies in commercial systems:
  - VM monitor can control and modify VMs but not the other way
  - Web applications in the browser cannot modify the host

- Mutual isolation is a related requirement:
  - Type-safe Java or .Net apps running in the same runtime environment
  - Web pages in different tabs or iframes
  - Apps running in the same phone
- Theoretically, mutual isolation equals multiple Biba policies

Extra material

Information flow security

- BLP and Biba are information flow policies
  - BLP prevents flow of information from high to low
  - Biba prevents flow of information from low to high
- Information flow policies are the basis for many security proofs. Typical proofs show non-interference:
  - view of one subject is not affected by the data of the other
  - low output does not depend on high input (secrets), or high output (integrity) does not depend on low input



- How to combine BLP and Biba in the same system?

High water mark, low water mark

- How to classify an object that combines low and high inputs?

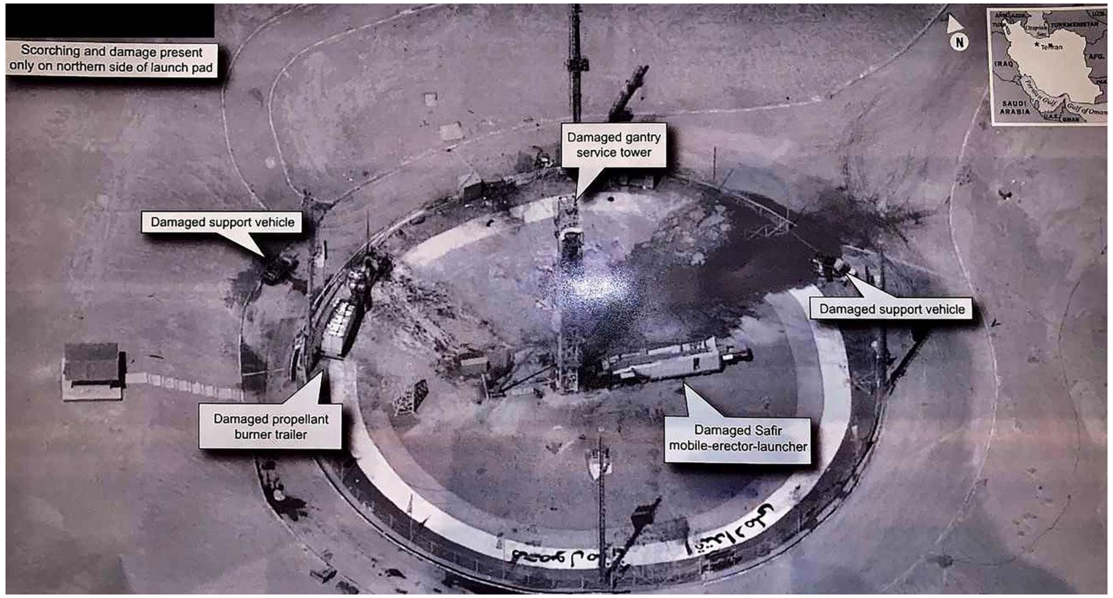
  → High water mark policy for secrecy: highest input level

  → Low water mark policy for integrity: lowest input level

- Low water mark can also be applied to subjects
- Problem: over time, all documents will become top secret with the lowest integrity level

Upgrading and downgrading

- In practice, security levels need to be changed by humans
  - E.g. downgrading intelligence data to enable broader access
  - E.g. upgrading intelligence reports based on open data
- Declassification = downgrading to unclassified level


[redacted]
Scorching and damage present only on northern side of launch pad
N
Damaged gantry service tower
Damaged support vehicle
Damaged support vehicle
Damaged propellant burner trailer
Damaged Safir mobile-erector-launcher

Declassified in a tweet in 2019

Data sanitization

- Documents may need to be sanitized i.e. redacted before downgrading or declassification
  - E.g. removing names of personnel before publication

- Sanitization of electronic documents is technically difficult
  - Painting black box over text in PDF is not enough:
    e.g. http://download.repubblica.it/pdf/rapportousacalipari.pdf

- High subjects can leak data intentionally via covert channels !
  - E.g. character spacing on printed documents; time between network messages; variation in processor load

Covert channels

Extra material

- Covert channels
  - Uncontrolled channels used for circumventing access control
  - Used by a high-level subject to intentionally leak data from high to low level in violation of the no-write-down policy
  - Examples: modulating processor, disk or network load to communicate bits; leaking secret bits in data fields that should be random numbers
  - Often refers to low-bitrate channels within multi-level operating systems
- Steganography
  - Hiding the existence of covert communication by embedding it into another (overt) communication
  - Examples: hiding data in photo files, character spacing in text documents
- Side channels
  - Channels that leak information from high to low without the intention of high-level subjects
  - Examples: processor power consumption variation may leak a secret key; increase in pizza deliveries to Pentagon predicts war

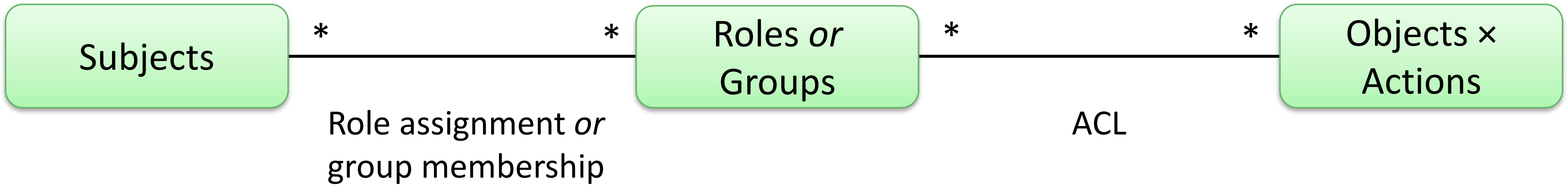
- Notes about terminology: Side channels are not covert channels because the communication is not intentional. Some academics make a distinction between the use of uncontrolled channels (covert channels) and misuse of legitimate, access-controlled channels (steganography), but the boundary between them is not always clear.

ROLE-BASED ACCESS CONTROL

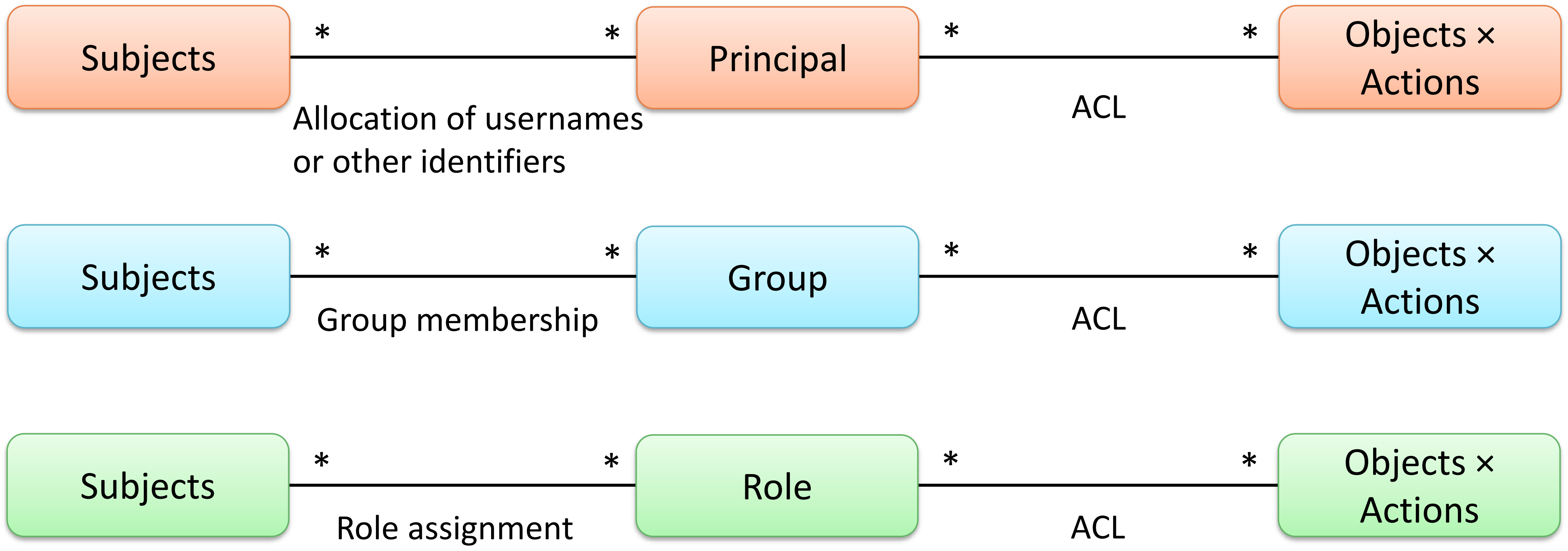
Groups and roles

Adding structure to AC model: a layer of indirection

- Group = set of subjects
  - E.g. Administrators, CS-C3130-students
  - Object’s ACL can list groups in addition to individual users
  - Both group membership and ACLs change over time
- Role = set of permissions (=set of actions on objects)
  - E.g. Administrator, CS-C3130-teacher, SCI-professor
  - Roles are usually static; only assignment to users



Indirection in access control systems



- Principals, groups and roles are conceptually quite different from each other, but all add a layer of indirection to the AC model
- They can be used for the same purposes – or to implement each other
- Could have multiple layers of indirection, e.g., principals could be mapped to groups or roles

Roles in some systems

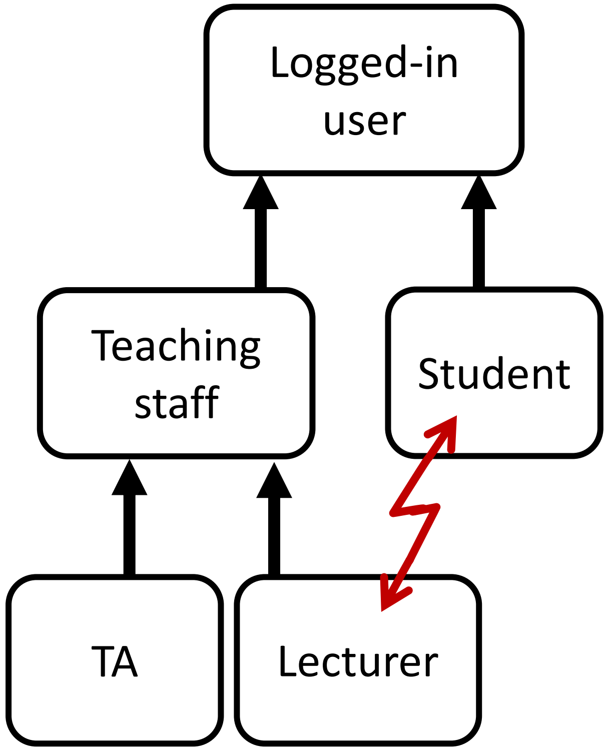
- MyCourses: Teacher, Non-editing teacher, Student, Category assistant, Workspace assistant of workspace; Aalto / HAKA logged-in user, Guest
- Moodle default roles: Site administrator, Manager, Course creator; Teacher, Non-editing teacher, Student of a course; Authenticated user, Guest
- WordPress: Administrator, Editor, Author, Contributor, Subscriber
- Joomla: Super Administrator, Administrator, Manager, Published, Editor, Registered

Role-based access control (RBAC) (1)

- NIST standard – but rarely fully implemented
- Modeling static high-level roles in an organization
  - – Doctor, Nurse, Student, Lecturer, Teaching assistant
- Roles are often parameterized
  - – Treating-doctor of patient S. Smith,
    Manager or employee 784499,
    Lecturer of CS-C3130,
    Student of CS-C3130

Role-based access control (RBAC) (2)

- Roles may form a hierarchy with rights inheritance
  - e.g. Lecturer and Teaching-assistant are Teaching-staff
- Constraints on role assignment and simultaneous activation can implement separation of duty (defined later)
- Roles are assigned to users for longer term but activated on demand for each session
- Groups are often used like roles, but some features are missing
  - e.g. Administrators, Domain Administrators in Windows



OTHER ACCESS CONTROL MODELS AND VARIATIONS

Attribute-based access control (ABAC)

- Access control is based in subject attributes (properties) instead of subject identity

    ABAC = attribute verification + authorization

- E.g. need to be 18 to buy tobacco;
  need to be an Aalto student to access course material
- Enables anonymous access

Separation of duty

- Separation-of-duty policy: two or more persons are required to complete a task
- Common in business and public administration:
  - Employees cannot approve their own expense claims
  - Financial auditors of large companies must be from outside the company
  - Safe may have two locks with keys held by two different persons
  - Lecturers issue grades to students, but only admin staff can enter them into the study register

- Unlike BLP and Biba, separation-of-duty policies are often stateful, i.e., they must have a memory of past events !

Chinese Wall model

- Chinese Wall model for accounting and consulting companies to avoid conflicts of interest
  - The same law office cannot represent both sides of a dispute
  - Consulting company can have mutually competing customers; they are assigned to different employees who do not speak to each other
- Generalized Chinese Wall model:
  - If subject S has previously accessed an object O1 and the objects O1 and O2 are in a conflict of interest, then S may not access O2
  - Idea: subject can fall to either side of the wall but cannot change sides later
- It is a separation-of-duty policy

Why "Chinese Wall"? After you fall to one side, can never get to the other side of the wall.

Clark-Wilson model

Extra material

- Data integrity cannot always be expressed in terms of ACL or MLS, i.e. who has access to what data
  - E.g. transfers between bank accounts must not change the total balance
- Integrity in commercial systems depends on following the correct procedures
- Clark-Wilson model defines rules for commercial systems for maintaining data integrity:
  - Transactions must transform data items from a consistent state to another consistent state
  - System-specific auditing and procedural controls enforce this
  - Result: If the initial state is consistent, the system will always be consistent
- Clark-Wilson model has not really been implemented; it is important because of the idea of accounting rules as a model for security policy

Other access control models

- Originator-controlled AC (ORCON)
  - Creator of data retains control over access to it
- Need-to-know principle
- Location or context-based AC

AC models are easier to define than to enforce

CONCLUSION

Why study these AC models?

- The abstract models help to recognize common patterns between different products and implementations
  - E.g. how many different user interfaces have you seen that implement an ACL?
- They also help to understand the expressiveness and limitations of the technologies
  - E.g. a stateless AC system cannot implement separation of duty, while one separation-of-duty policy can often be used to implement others
- Some models presented in this lecture are unrealistic! Nevertheless, they can be useful as tools for thinking about security policies

List of key concepts

- Access control, subject, object, action, access request, reference monitor, policy, auditing, authentication, authorization
- TCB, security kernel, isolation
- AC matrix, protection state, discretionary AC, identity-based AC, ACL, capability
- Principal, process, user
- Mandatory AC, rule-based AC, multi-level security, label, clearance, classification, Bell-LaPadula, no read down, no write up, Biba, high and low water mark, upgrading, downgrading, sanitization, redaction, covert channel
- Role-based AC, group, role, parameterized role, role hierarchy, inheritance
- Separation of duty , stateful policy, Chinese Wall model
- Attribute-based AC, location-based AC, context-based AC

Reading material

- Dieter Gollmann: Computer Security, 2nd ed., chapters 4, 8, 9; 3rd ed. chapters 5–6
- Edward Amoroso: Fundamentals of Computer Security Technology, chapters 6-13
- Ross Anderson: Security Engineering, 2nd ed., chapter 8
- Stallings, Brown: Computer Security: Principles and Practice, 4th ed., chapter 4

Problems to think about

- What are the subjects, object and actions in MyCourses / Sisu?
- Are there architectural or performance reasons for choosing between ACLs and capabilities? Consider distributed and scalable systems.
- Can you think of security mechanisms outside computers which would need MAC but actually implement DAC?
- What security labels and MAC policy would be suitable for MyCourses / Sisu?
- Give examples of systems that require confidentiality or integrity but not both.
- Which AC model and what kind of security labels could be used to describe VM or process isolation?
- If Bell-LaPadula and Biba policies are in the same system, should they use the same labels? Give an example.
- Define RBAC roles that could be used in the implementation some IT system that you know well. To what extent can your RBAC policy be implemented with groups?
- What kind security policy is there in double-blinded medical experiments?
- To what extent can access control prevent spam (email or other)? Can you think of other *open systems* where it is difficult to implement access control?
- When services and content are mainly in the cloud rather than on the user’s own computer, how does that change the types of policies that can or need to be enforced? Consider, for example, a multi-player game or social network.