ECE 443/518 – Computer Cyber Security
Lecture 21 Access Control I

Professor Jia Wang
Department of Electrical and Computer Engineering
Illinois Institute of Technology

November 4, 2024

Outline

Reading Assignment

- This lecture: ICS 2,4,5
- Next lecture: ICS 6,7,14

Outline

Access Control

- How people could interact securely if secure collaborations are
  - not available, e.g. before the invention of public-key cryptography?
  - too costly, e.g. for secure multi-party computation?
- Study the relation between
  - Subject: who? active entities like human and processes.
  - Object: what? entities containing information like files.
- Access Control: who can access what?
  - A.k.a. Authorization
- Assume certain protocol/mechanism can be enforced.
  - E.g. ignore authentication – assume identities of subjects can be established.

System

- A computing system.
  - Or any system that stores and processes information.
- Modeled as a finite state machine: states and transitions.
  - registers+memory locations+secondary storage
- Protection states: only certain bits of system states matter.
  - Depending on how subjects access objects per each state.

Secure System

- Secure policy: what protection states are secure and what protection states are insecure.
- Secure system: starting from any secure state, one cannot reach any insecure state.
  - Breach of security if an insecure state is reached.
- Security mechanism: prevents transition from secure to insecure states.

Access Control Matrix

- A framework to describe access control.
- Rows: subjects
- Columns: objects
- $a[s,o]$: rights of subject s on object o.

Access Control Matrix Example

	file 1	file 2	process 1	process 2
process 1	read, write, own	read	read, write, execute, own	write
process 2	append	read, own	read	read, write, execute, own

Figure 2–1 An access control matrix. The system has two processes and two files. The set of rights is {read, write, execute, append, own}.

(Bishop)

Protection State Transitions

- Primitive operations on access control matrix.
  - As a basis to reason with transitions.
  - As a basis to implement access control matrix.

1. create subject s
2. create object o
3. enter r into $a[s, o]$
4. delete r from $a[s, o]$
5. destroy subject s
6. destroy object o

Protection State Transitions Example

- $CreateFile(p, f)$
  - p: subject
  - f: object (the file to create)

1. create object f
2. enter *own* into $a[p, f]$
3. enter *read* into $a[p, f]$
4. enter *write* into $a[p, f]$

- Can any other subject q access f?
  - Who is allowed to modify $a[q, f]$?
  - What if we would like every one to read but not write f?
  - What about a new subject?

Difficulties

- Subject: Alice and Bob
- Object: file X
- Secure states: Alice can but Bob cannot access X
- What if Alice copies X into Y and allows Bob to access Y?
  - Obviously you cannot simply forbid Alice to copy X, e.g. Alice could memorize X and at a later time append it to a file Y that Bob has access.

Outline

Security Properties

- Confidentiality: no member of a set X of entities obtain information or resources I.
  - Information flow: someone in X may obtain I indirectly via entities authorized to obtain I.
- Integrity: all members of a set X of entities trust information or resources I.
  - Trust comes from authorization on who and how to modify I.
  - Separation of duties: multiple entities should be involved.
- Availability: all members of a set X of entities can access information or resources I.
- Security policies involve one or more of such properties.

Policy vs. Mechanism

- Security policy: one cannot copy another's homework.
- If A copies B's homework file because B forgot to read protect the homework file, who breaches security?
  - Obviously A breaches security.
  - However, B doesn't since there is no security policy for B to read protect the homework file.
- There is no mentioning of read protection in the security policy.
  - Read protection is a security mechanism: something that can be enforced for a security policy.
- By enforcing file access control as a security mechanism, A can no longer copy B's homework file.
  - Still, A may find other ways to copy other's homework.

More Example on Policy vs. Mechanism

- Security policy: information regarding a particular product is proprietary and is not to leave the control of the company.
- What about backups containing such information on cloud?
- Security mechanism
  - Depend on how cloud controls access to such information in plaintext.
  - Or the company can make use of cryptography.

Types of Security Policies

- A military security policy (also called a governmental security policy) is a security policy developed primarily to provide confidentiality.
- A commercial security policy is a security policy developed primarily to provide integrity.

The Role of Trust

- To reason with security policies and security mechanisms requires certain assumptions.
- Trust: are these assumptions valid?
- Download and install patch to improve OS security.
  - Patch is authentic.
  - Patch is of good quality.
  - Patch installs correctly.
  - Patch will not interfere with existing configurations.

Types of Access Control

- Discretionary access control (DAC)
  - A.k.a. identity-based access control (IBAC).
  - An individual user can set an access control mechanism to allow or deny access to an object.
  - E.g. you use a password to control who can visit your website.
- Mandatory access control (MAC)
  - Occasionally called a rule-based access control.
  - A system mechanism controls access to an object and an individual user cannot alter that access.
  - E.g. laws may grant access to certain information without owner's permission.

Outline

Goals of Confidentiality Policies

- A.k.a information flow policy.
  - Unauthorized entities may access information indirectly.
- Prevent the unauthorized disclosure of information.
  - Integrity and availability are not of concern.

The Bell-LaPadula Model

- Military-style classifications for confidentiality.
- Goal: prevent read access to information at a security classification higher than personnel's clearance.
  - E.g. to prevent someone to read a secret and then publish it somewhere for anyone to access.
- Combining mandatory access control defined via security classifications, and discretionary access control.

Access Control Details

- Security classification: sensitivity levels of object (information).
  - The higher the levels, the greater the need to keep it confidential.
  - E.g. TOP SECRET (TS) $>$ SECRET (S) $>$ CONFIDENTIAL (C) $>$ UNCLASSIFIED (UC)
  - Written as $L(O)$ for object O.
- Security clearance: levels of subject (entities).
  - Same choice of levels as security classification.
  - Written as $L(S)$ for subject S.
- Discretionary access control.
  - A subject S has discretionary read (or write) access to an object O.

Simple Security Condition and Star Property

- Simple Security Condition: S can read O if and only if $L(O) \leq L(S)$ and S has discretionary read access to O.
- $*$-Property: S can write O if and only if if $L(O) \geq L(S)$ and S has discretionary write access to O.
- Read down, write up.
  - No reads up, no writes down.
- Basic Security Theorem: the system remains secure if transitions preserve simple security condition and $*$-property.
  - Information always flows from lower-level objects to higher-level objects.
  - Assume subjects only communicate via objects.

Bell-LaPadula Example

- Security clearance and classification
  - TS Tamara, Personnel Files
  - S Sally, Electronic Mail Files
  - C Claire, Activity Log Files
  - UC Ulaley, Telephone List Files
- Can Claire and Ulaley read Personnel Files?
- Can Tamara read Telephone List Files?
- Can Tamara read Personnel Files to obtain everyone's password and write them into Activity Log Files?

Extension: Categories

- Object may belong to multiple categories.
  - Contain sensitive information regarding all those categories.
  - Written as $C(O)$ for object O.
- Subject may access multiple categories.
  - "need to know": no subject should be able to read objects unless reading them is necessary.
  - Written as $C(S)$ for subject S.
- Simple Security Condition: S can read O if and only if $L(O) \leq L(S)$ and $C(O) \subseteq C(S)$ and S has discretionary read access to O.
- $*$-Property: S can write O if and only if if $L(O) \geq L(S)$ and $C(S) \subseteq C(O)$ and S has discretionary write access to O.
- Basic Security Theorem holds similarly.

Bell-LaPadula Example with Categories

- Subjects
  - George: $(SECRET, \{NUC, EUR\})$
  - Paul: $(SECRET, \{EUR, US, NUC\})$
- Objects
  - DocA: $(CONFIDENTIAL, \{NUC\})$
  - DocB: $(SECRET, \{EUR, US\})$
  - DocC: $(SECRET, \{EUR\})$
- What can George read?
- What can Paul read?
- What can Paul write?

The Need to Decrease Security Level

- Paul cannot write anything that can be read by George.
  - This is reasonable since Paul knows information *US* which George cannot know.
  - But this is at least not convenient.
- Current security level: a subject may (effectively) decrease its security level from the maximum in order to communicate with entities at lower security levels.
  - Paul can decrease to $(SECRET, \{EUR\})$ to write DocC that George can read.
- Essentially, decreasing security level implies the subject should "forget" any information from higher security levels.
  - Paul need to "forget" anything in $(SECRET, \{US, NUC\})$ to reach $(SECRET, \{EUR\})$.
  - The challenge is how to enforce such requirement.

Summary

- From a system perspective, security policies mostly concern of access control (a.k.a. authorization) – who can do what at when.
  - Security mechanism concerns of how to enforce them.
- The Bell-LaPadula model provides confidentiality but may prevent a personnel with more sensitive knowledge to communicate with a personnel with lower security levels.