ECE 443/518 – Computer Cyber Security
Lecture 22 Access Control II

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

November 6, 2024

Outline

Reading Assignment

- This lecture: ICS 6,7,14
- Next lecture: Digital Forensics

Outline

Goals and Principles of Operation

- Goal: preserve integrity of data.
  - E.g. among developers and users of a system.
- Separation of duty: allow multiple parties to perform a critical function to prevent a single one to cheat.
  - Less chance of collusion when more parties are involved.
- Separation of function: partition the system functionality so each party only works on a necessary portion.
- Logging and auditing: provide recovery and accountability.

Biba Integrity Model

- A set O of objects representing data.
- A set S of subjects representing who can access data.
- A set I of integrity levels representing trust.
- A function i that assigns a subject/object an integrity level.
- Biba's model
  - $s \in S$ can read $o \in O$ if and only if $i(s) \leq i(o)$.
  - $s \in S$ can write to $o \in O$ if and only if $i(o) \leq i(s)$.
  - $s_1 \in S$ can execute $s_2 \in S$ if and only if $i(s_2) \leq i(s_1)$.
- Read up, write down.

Outline

Chinese Wall Model

- Derived from the British laws concerning conflict of interest.
  - Refer equally to confidentiality and integrity.
- Example: investment house.
  - Object (data): records provided by client companies.
  - Subject (analysts): make use of data to guide investments for client companies.
  - Conflict of interest: an analyst cannot provide guidance to two companies in competition, since potentially one may gain at others expense.

Formal Model

- The objects of the database are items of information related to a company.
- A company dataset (CD) contains objects related to a single company.
  - $CD(O)$: the company dataset that contains object O.
- A conflict of interest (COI) class contains the datasets of companies in competition.
  - $COI(O)$: the COI class that contains object O.
  - Assume each object belongs to exactly one COI class.
- An analyst cannot read data from two companies if they belong to the same COI class.
  - $PR(S)$: set of data read by the analyst S so far.

CW-Simple Security Condition

- S can read O if and only if any of the following holds.

1. There is an object O' such that S has accessed O' and $CD(O') = CD(O)$.
   - If S is reading data from a company, S is allowed to read other data from the same company.
2. For all objects $O' \in PR(S)$, $COI(O') \neq COI(O)$.
   - S can read data from a COI class S never read before.
3. O is a sanitized object.
   - S can read data that is publicly available.

CW-$*$-Property

- S may write O if and only if both of the following conditions hold.

1. The CW-simple security condition permits S to read O.
2. For all unsanitized objects O' that S can read, $CD(O') = CD(O)$.
   - S cannot propagate information between different companies.

Outline

A Naive Implementation

- Implement access control matrix as a 2-D array.
- Issues
  - Lots of subjects and objects imply huge storage requirement.
  - To create and destroy subjects and objects require to manage array storage dynamically, which is complicated and could lead to buggy implementation.
  - To search for certain information is not efficient, e.g. who is the owner of an object?
- Observations
  - No access: empty entries.
  - Default rules: similar/same entries.
  - Hierarchy for data management: same entries.

Access Control Lists (ACL)

- Store each column as a list.
  - $acl(o)$ per object o.
  - Consist of pairs (s, r): r describes how s could access o.
  - Save storage by not storing $r = \emptyset$.
- Owner of object can be stored with the list to avoid search.
- Use additional optimizations to save storage further.
  - Group subjects to reduce size of every ACL.
  - Use default values to eliminate most ACLs.

ACL 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}.

acl(file 1) = { (process 1, { read, write, own }), (process 2, { append }) }
acl(file 2) = { (process 1, { read }), (process 2, { read, own }) }
acl(process 1) = { (process 1, { read, write, execute, own }), (process 2, { read }) }
acl(process 2) = { (process 1, { write }), (process 2, { read, write, execute, own }) }

(Bishop)

ACL for Unix/Linux Systems

- Abbreviation by grouping subjects.
- Three classes of subjects: user, group, others.
  - Two subjects are associated with each object – a owner and a group owner.
  - Class user: the owner.
  - Class group: subjects in the same group as the group owner.
  - Class others: all other subjects.
- Three access rights: read, write, execute.
  - 3-bit for each group of subjects: r highest, x lowest.
  - Written as an octal number or 3 letters.
    - e.g. 7 for 'rwx' and 5 for 'r-x', where '-' stands for not allowed.
- Need 9 bits to store ACL for each object.
  - Written as 3 octal number or 9 letters for user, group, and others from left to right.
  - e.g. 755 for 'rwxr-xr-x' where user (owner) can read/write/execute, group (group member) can read/execute, others can read/execute.

ACL for Unix/Linux Systems (Cont.)

- Permission for directories
  - read: list files in directory.
  - write: create/delete file, modify file name.
  - execute: enter directory.
- setuid and setgid for executable files: one bit each
  - If you execute a file, the system shall use your id and your group id to authorize accesses to files.
  - As an exception, you may execute the file using its owner's id if setuid is set, or its group owner's id if setgid is set.
  - Useful to expose resources (not necessarily files) accessible only by the owner to other users.
- setgid for directory
  - All subdirectories created will have the same group owner.
  - Useful to share directories among a group of users.
- It is also possible to use ACLs to assign fine grained permissions to each subject.

ACL Flavors

- Which subjects can modify an object's ACL?
  - At least the owner should be able to.
  - In some systems, other subjects may be allowed to modify ACL, at the cost of additional storage and system complexity.
- Do the ACLs apply to privileged users?
  - No for Linux: but what about 'sudo rm -rf /'?
  - Yes for Windows: but how do administrators remove malicious software?
- Does the ACL support groups or wildcards?
  - Save storage and effort but be careful about new subjects/objects.
- How are contradictory permissions handled?
  - Take the more specific match.
  - Deny access if any denies.
  - Take the first match.
- What about default settings?
  - Default applies last, e.g. to deny.
  - Apply default setting at creation and allow to modify.

Summary

- Integrity policies protect data integrity by constraining who can do what in the system.
  - Biba: subjects and objects. Read up, write down.
- Hybrid policies allow to protect both integrity and confidentiality.
- Access Control Lists (ACL) is a possible implementation of access control matrix, and is optimized for practical use.