Mandatory Access Control

1 Why need MAC

- DAC: Discretionary Access Control
  - Definition: An individual user can set an access control mechanism to allow or deny access to an object.
  - Relies on the object owner to control access.
  - DAC is widely implemented in most operating systems, and we are quite familiar with it.
  - Strength of DAC: Flexibility: a key reason why it is widely known and implemented in mainstream operating systems.
- Limitation of DAC:
  - Global policy: DAC let users to decide the access control policies on their data, regardless of whether those policies are consistent with the global policies. Therefore, if there is a global policy, DAC has trouble to ensure consistency.
  - Information flow: information can be copied from one object to another, so access to a copy is possible even if the owner of the original does not provide access to the original copy. This has been a major concern for military.
  - Malicious software: DAC policies can be easily changed by owner, so a malicious program (e.g., a downloaded untrustworthy program) running by the owner can change DAC policies on behalf of the owner.
  - Flawed software: Similarly to the previous item, flawed software can be "instructed" by attackers to change its DAC policies.
- MAC: Mandatory Access Control
  - Definition: A system-wide policy decrees who is allowed to have access; individual user cannot alter that access.
  - Relies on the system to control access.
  - Examples:
    - The law allows a court to access driving records without the owners' permission.
  - Traditional MAC mechanisms have been tightly coupled to a few security models.
  - Recently, systems supporting flexible security models start to appear (e.g., SELinux, Trusted Solaris, TrustedBSD, etc.)
- Security Policy Model
  - A security policy model is a succinct statement of the protection properties that a system, or generic type of system, must have.

2 Multilevel Security

2.1 Security Levels

- People and information are classified into different levels of trust and sensitivity.
- These levels represent the well-known security classifications:

  Unclassified $\Longrightarrow$ **Confidential** $\Longrightarrow$ **Secret** $\Longrightarrow$ **Top Secret**.

- *Clearance level* indicates the level of trust given to a person with a security clearance, or a computer that processes classified information, or an area that has been physically secured for storing classified information. The level indicates the highest level of classified information to be stored or handled by the person, device, or location.
- *Classification level* indicates the level of sensitivity associated with some information, like that in a document or a computer file. The level is supposed to indicate the degree of damage the country could suffer if the information is disclosed to an enemy.
- *Security level* is a generic term for either a clearance level or a classification level.

2.2 The Bell-LaPadula Security Policy Model

- Proposed by David Bell and Len Lapadula in 1973, in response to U.S. Air Force concerns over the security of time-sharing mainframe systems.
- This model is the most widely recognized MLS model.
- The model deal with confidentiality only.
- Two properties: No read up and No write down.
  - *Simple security property:* **Subject A is allowed to read object O only if class(O)** $\leq$ **class(A)**.
  - **-property:* **Subject A is allowed to write object O only if class(A)** $\leq$ **class(O)**.
- The *-property was Bell and LaPadula's critical innovation. It was driven by the fear that a user with "Secret" clearance might be "tricked" by attackers (e.g., through Trojan horse programs or software vulnerabilities) to copy down the information to a "Unclassified" area where the attackers can read.

2.3 The Biba Model

- Due to Ken Biba.
- Deal with integrity alone and ignores confidentiality entirely.
- Biba model covers integrity levels, which are analogous to sensitivity levels in Bell-LaPadula
- Integrity levels cover inappropriate modification of data
- Prevents unauthorized users from making modifications (1st goal of integrity)
- Two properties:

- *Simple Integrity Property:* A low integrity subject will not write or modify high integrity data.
- **-Property:* The high integrity subject will not read low integrity data.
- *Read Up, Write Down* - Subjects cannot read objects of lesser integrity, subjects cannot write to objects of higher integrity

3 Multilateral Security

- Instead of the information flow-control boundaries being horizontal, as in the MLS model, we instead need the boundaries to be the mostly vertical.
- Examples:
  - In a consultant company, a person who consult for BankOne should not have access to the data of JPMC-Chase.
  - An intelligence organization wants to keep the names of agents working in one foreign country secret from the department responsible for spying on another.
- Also known as compartmentation.
- Multilateral security models:
  - The Chinese Wall Model
  - The BMA Model (British Medical Association)
- The Chinese Wall Model
  - Proposed by Brewer and Nash to model access rules in a consultancy business where analysts have to make sure that no conflicts of interest arise when they are dealing with different clients. Informally, conflicts arise because clients are direct competitors in the same market or because of the ownership of companies. Analysts have to adhere to the following security policy:

    Rule: There must be no information flow that causes a conflict of interest.
  - Conflict of Interest (CoI) classes: indicate which companies are in competition.
  - **Read Rule:** A subject S can read an object O if:
    - O is in the same Dataset as an object already accessed by S, *or*
    - O belongs to a CoI class from which S has not yet accessed any information.
  - **Write Rule:** A subject S can write an object O if:
    - S can read O according to the Read Rule, *and*
    - No object in a different company dataset (i.e., not O's company dataset) can be read.
  - In the write rule, the flow of information is comfined to its own company dataset. Without this rule, a person who can access both A and B can read the information from A and write to B; this way, another person who can access B can also access the information in A indirectly. If this person can also access C, which is in the same CoI class as A, we have a violation.
  - The access restriction for both read and write can be lifted for sanitized information.

4 Reference Monitor

- A good design of security system should separate the policy enforcement from the rest of the system.
- Since its 1972 introduction in the "Anderson Report", the Reference Monitor (RM) concept has proved itself to be a useful tool for computer security practitioners. It has been the only effective tool we know of for describing the abstract requirements of secure system design and implementation.
- A Reference Monitor should have the following properties
  - It must be always invoked, i.e., every access is mediated.
  - It must be tamperproof. It is impossible for a penetrator to attack the access mediation mechanism such that the required access checks are not performed and authorizations not enforced.
  - It must be small enough to be subject to analysis and test, the completeness of which can be assured.

5 MAC Implementation in Windows Vista

- It is called *Mandatory Integrity Control (MIC)* in Windows Vista. MIC implements a form of the Biba model, which ensures integrity by controlling writes and deletions.
- *Label on Subjects:* When a user logs on, Windows Vista assigns an integrity SID to the users access token. Included in the SID is an integrity label that determines the level of access the token (and thus the user) can achieve.
- *Label on Objects:* Objects, such as files, pipes, processes, threads, registry keys, services, printers, etc., are also assigned an integrity SID, which is stored in the system access control list (SACL) of the objects security descriptor. The label in the SID specifies the integrity level of the object.
- *Access Control Policy:* To write to or delete an object, the integrity level of subject must be equal to or greater than the object's level.
- *Relationship to DAC:* Vista checks MAC first, if passed, it then checks DAC (e.g. access control list). Therefore, MAC provides a layer of access control in addition to DAC; it does not overwrite DAC.
- *Integrity levels:* Windows Vista defines six integrity levels (IL): *Low, Medium, High, and System*.
  - Untrusted.
  - Low: everyone (i.e. world).
  - Medium: standard users, authenticated users.
  - High: local service, network serice, elevated users.
  - System: system services.
  - Trusted Installer.

  Usually, child processes inherit the integrity level of their parents, unless the executable program running in the child process has a lower integrity level. For example, all the downloaded executables will run with Low integrity level because the labels of the executable programs are marked as Low when they are downloaded from the Internet.

The integrity level can also be customized on a per-process basis. For example, Internet Explorer 7 runs has an Low integrity level only. This means that IE has limited opportunities to be able to alter files on the machine without triggering an elevation prompt that the user must agree to.

Objects created by processes inherit the IL of the process. So files downloaded by IE still have an IL of Low – this explains why downloaded executables will only run with Low integrity level.

- *Default levels:* Objects that lack an integrity label are treated as medium by the operating system. This prevents low integrity code from modifying unlabeled objects
- *Viewing and Modifying ILs of Objects:* use `icacls`.

6 SELinux in Linux

Primarily developed by the US National Security Agency, SELinux has been integrated into version 2.6 series of the Linux kernel. SELinux can enforce mandatory access controls (MAC).

- Type Enforcement (TE)
  - Implements Domain Type Enforcement.
  - Labeling subjects and objects: Domains for processes and Types for files and other objects
  - Rules enforce how domains can access types.
- Key concepts:
  - *Domains:* Classification of a subject.
  - *Types:* Classification of an object (really the same thing as a domain but applied to objects).
  - *Users:* Identifier for a single user or an equivalence class of users.
  - *Class:* Type of an object, e.g., file or process.
- Two basic security enforcement decisions
  - *Access control:* Can subject access object?
  - *Labeling:* What label should a new object have?
- Access rules:
  - Syntax: `(allow | auditallow | dontaudit) src_type target_type:classes permissions;`
  - Example: `allow sshd_t shell_exec_t:file execute;`
  - Meaning: when a subject of `sshd_t` accesses an object of `shell_exec_t`, it has the `execute` permissions if the object is the `file` class.
- Rules for the type of a new object:
  - Example: `type_transition sshd_t tmp_t:  devfile_class_set cardmsg_dev_t;`
  - Meaning: When sshd daemon creates a device file in the tmp directory, the new file is labeled with `cardmsg_dev_t`.

7 AppArmor in Linux

AppArmor is a Linux Security Module implementation of name-based access controls. AppArmor confines individual programs to a set of listed files, posix 1003.1e draft capabilities, and network accesses.

- Rules in AppArmor policies.
  - Policy on file/directory access: AppArmor give an application explicit read, write, or execute access to files and directories. The following rule allows the read of `/etc/ntp.conf`.

    ```
    /etc/ntp.conf      r,
    ```

  - AppArmor enables the mediation of POSIX capabilities. For example, the following rule allows the use of `sys_chroot` capability.

    ```
    capability sys_chroot
    ```

  - AppArmor allows mediation of network access based on the address type and family. For example, rule A in the following allows the general use of IPv4 network; rule B allows the use of IPv4 TCP networking.

    ```
    A:  network inet
    B:  network inet tcp
    ```

- An example of AppArmor policy:

```
/usr/sbin/ntpd {
   capability sys_chroot,
   capability setuid,
   /etc/ntp.conf      r,
   /var/log/ntp       w,
   /tmp/ntp*          rwl,
}
```

- Comparing with SELinux: SELinux and AppArmor are competing products, and they both implement Mandatory Access Control.
  - AppArmor's design is name-based and does not require re-labeling of the file system.
  - The AppArmor profiles refer directly to programs and files by name, while the SELinux policies refer only to labels, and the user must label their files and programs in a separate step.