Goals for Today

• Learning Objective:
  • Understand the foundations of securing operating systems

• Announcements, etc:
  • MP4 due April 24th
  • C4 Students: Presentations start next next Wednesday!

Reminder: Please put away devices at the start of class
CS 423
Operating System Design: Security
What security considerations exist in operating system?
What security considerations exist in operating system?

An (incomplete) list of threats to your operating system...

Denial-of-Service: Attacker prevents or degrades normal use of system.
- Binary Exploitation: Programming error (e.g., buffer overflow) in the kernel code is exploited to take control of system.
- Data Exfiltration: Attacker performs unauthorized reads on your data objects.
- Data Tampering: Attacker performs unauthorized writes on your data objects.
OS Security Goals

- Data confidentiality:
  - Preventing data exposure
- Data integrity:
  - Preventing tampering with data
- System availability:
  - Preventing denial of service
OS Security Goals
OS Security Goals

• How do we achieve security goals in operating systems?
How do we achieve security goals in operating systems?

• Cryptography?
  • Prohibitive costly
  • Limited support for crypto in kernel
How do we achieve security goals in operating systems?
- Cryptography?
  - Prohibitive costly
  - Limited support for crypto in kernel
- Access Control!
  - Low overhead
  - Kernel already uses discretionary access control
Access Control

- Determine whether a principal can perform a requested operation on a target object
  - **Principal/Subject**: user, process, etc.
  - **Operation/Action**: read, write, etc.
  - **Object**: file, tuple, etc.
• A computer system is a set of processes and objects
• A process operates within a protection domain
• A protection domain specifies the resources a process may access and the types of operations that may be invoked on the objects.

• **The Principle of Least Privilege:** The protection domain of a process should be as small as possible consistent with the need of that process to accomplish its assigned task.
Protection Domains

Example of 3 protection domains:

Domain 1

File1[R]
File2[RW]

Domain 2

File3[R]
File4[RWX]
File5[RW]

Domain 3

Printer1[W]
File6[RWX]
Plotter2[W]

What are domains based on in Linux?
Find the Access Control!
Discretionary Access Control (DAC)

• Owner of creator of resources specified which subjects have which access to a resources

• Commonly implemented in commercial products

• Access is managed by individual users, not a central security policy
Discretionary Access Control (DAC)

Access Mask defines permissions for User, Group, and Other

```
chmod u=rwx,g=rx,o=r myfile
chmod 754 myfile
```

4 stands for "read",
2 stands for "write",
1 stands for "execute", and
0 stands for "no permission."
Access Control Lists

- Each column in access matrix specifies access for one Object.

- On invocation of a method R on an object O by a process running in a domain D,

- The access control list is searched to check whether D is allowed to perform method R on object O (e.g., allowed to read the file or execute the program).

- A default (e.g., “rest of the world”) can be associated with an access list so that any Domain not specified in the list can access the Objects using default methods.

- It is easy for the owner of the Object to grant access to another Domain or revoke access.

- ACL entries can be for individual users or for a group of users.
Use of access control lists to manage file access in UNIX
Capabilities

• Capability is an unforgeable ticket
  • Managed by OS
  • Can be passed from one process to another
• Permissive
• OS Mechanism (Reference monitor) checks ticket
  • Does not need to know the identity of the user/proc
• Can be used to partition superuser privilege
• Implementation: POSIX.1e capabilities
When capabilities are used, each process has a capability list.
• What might go wrong with DAC or Capabilities?
Problems?

• What might go wrong with DAC or Capabilities?
  • Security is left to the discretion of subjects
  • Impossible to guarantee security of system
  • Security of system changes over time.
• Solution?
Problems?

• What might go wrong with DAC or Capabilities?
  • Security is left to the discretion of subjects
  • Impossible to guarantee security of system
  • Security of system changes over time.

• Solution?
  • **Mandatory Access Control**: Operating system constrains the ability of subjects (even owners) to perform operations on objects according to a system-wide security policy.
Bell-LaPadula Model

- A multi-level security model that provides strong confidentiality guarantees.
- Formalizes Classified Information
- State machine (Lattice) specifies permissible actions
• The Simple Security Property: A subject running at security level k can read only objects at its level or lower. (*no read up*)

• The * Property: A subject at security level k can write only objects at its level or higher (*no write down*)
Using Bell-Lapadula, we can reason about permissible information flows in a system.
• Bell-LaPadula provides confidentiality. What about integrity?

• Biba model provides Integrity guarantees in a manner analogous to Bell-Lapadula’s secrecy levels.

• Integrity prevents inappropriate modification of data.
• **The Simple Integrity Property:** A subject running at integrity level k must not read an object at a lower integrity level (*no read down*)

• **The * Integrity Property:** A subject at security level k can only write objects at its level or lower. (*no write up*)
All the Access Controls

- Basic Access Matrix
  - UNIX, ACL, various capability systems
- Aggregated Access Matrix
  - TE, RBAC, groups and attributes, parameterized
- Plus Domain Transitions
  - DTE, SELinux, Java
- Lattice Access Control Models
  - Bell-LaPadula, Biba, Denning
- Predicate Models
  - ASL, OASIS, domain-specific models, many others
- Safety Models
  - Take-grant, Schematic Protection Model, Typed Access Matrix