CIT 380: Securing Computer Systems

Access Control

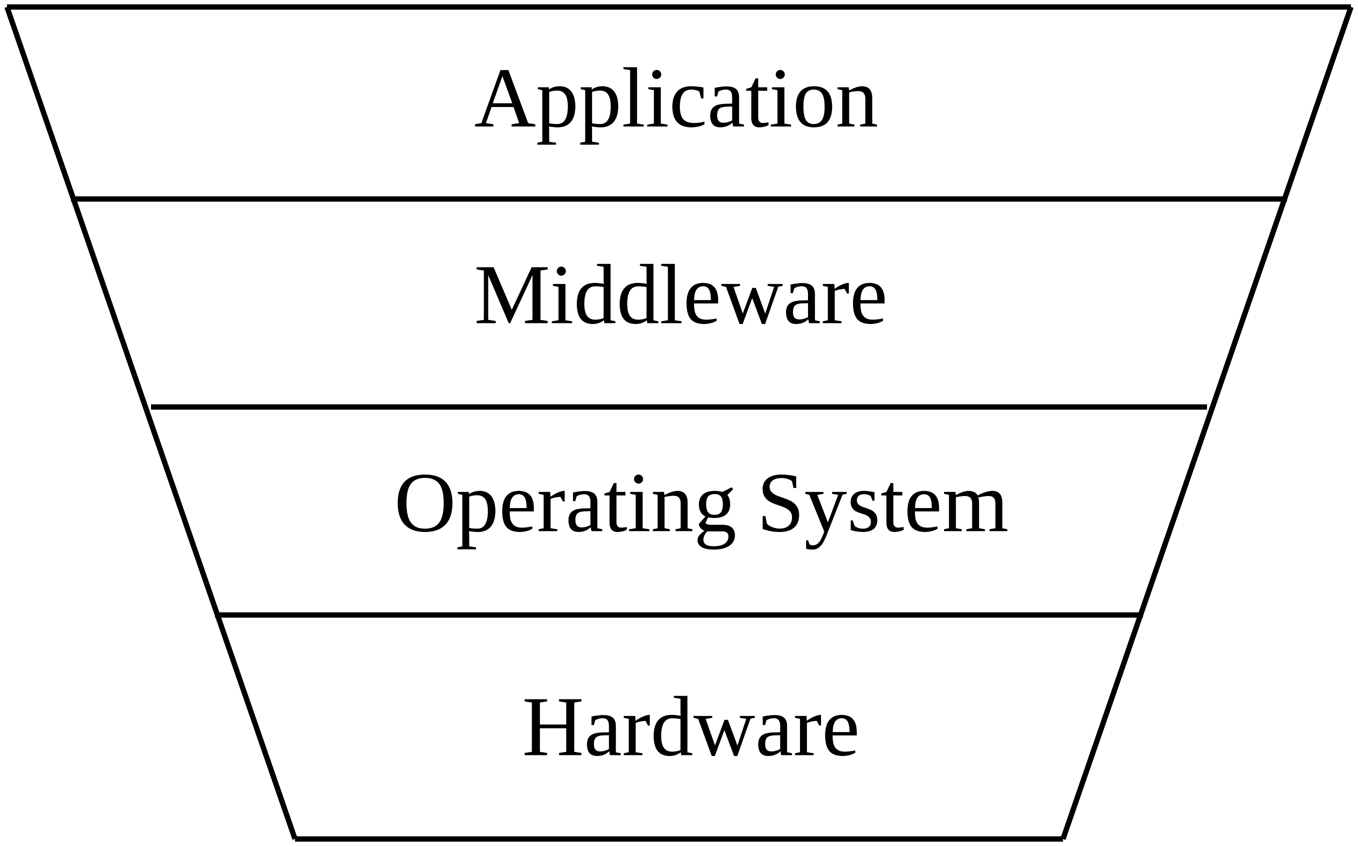
Topics

1. Access Control Matrix
2. ACLs and Capabilities
3. Role-Based Access Control (RBAC)
4. Discretionary Access Control (DAC)
5. Examples: UNIX, Windows, Android, SQL
6. Mandatory Access Control (MAC)
7. Hardware Protection

Access Control

Access control is the selective restriction of access to an information, computational, or physical resource.

Access Control is Pervasive



Access Control is Pervasive

1. Application
   - Complex, custom security policy.
   - Ex: Amazon account: wish list, reviews, CC
2. Middleware
   - Database, system libraries, 3rd party software
   - Ex: Credit card authorization center
3. Operating System
   - File ACLs, IPC, Android permissions system, SELinux
4. Hardware
   - Memory management, hardware device access.

Access Control Matrices

A table that defines permissions.

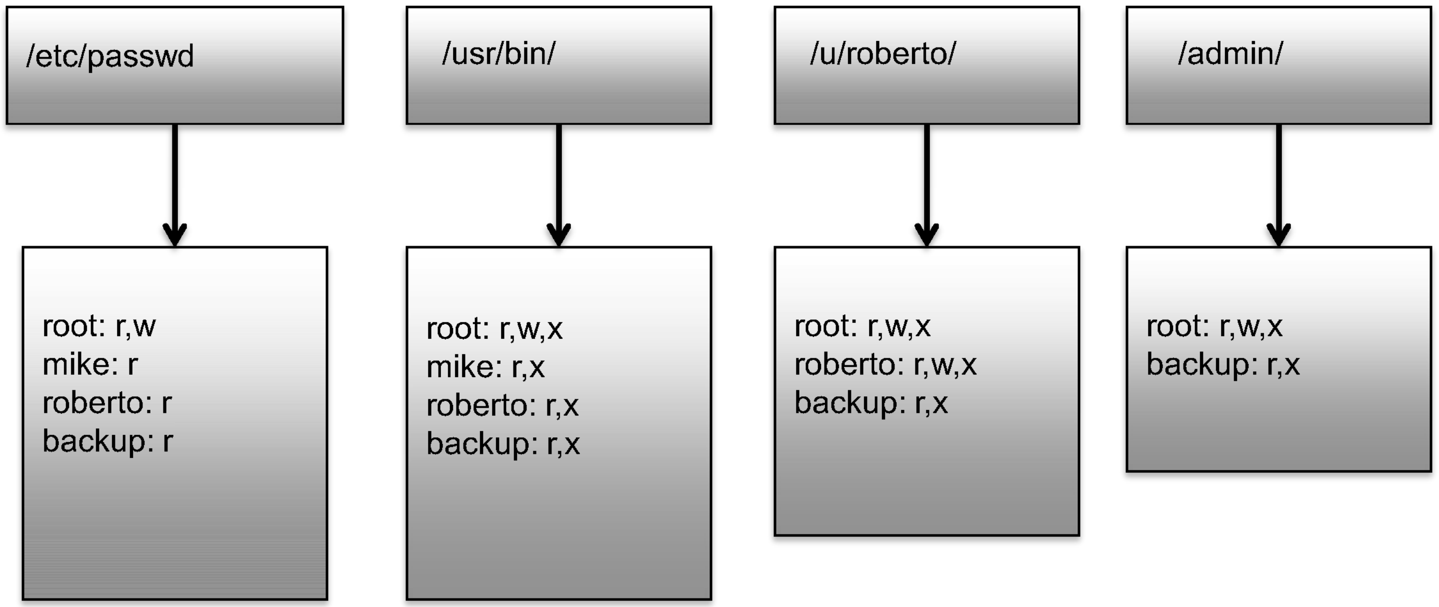
- Each row of this table is associated with a **subject,** which is a user, group, or system that can perform actions.
- Each column of the table is associated with an **object,** which is a file, directory, document, device, resource, or any other entity for which we want to define access rights.
- Each cell of the table is then filled with the access rights for the associated combination of subject and object.
- Access rights can include actions such as reading, writing, copying, executing, deleting, and annotating.
- An empty cell means that no access rights are granted.

Example Access Control Matrix

	/etc/passwd	**/usr/bin/**	**/u/roberto/**	**/admin/**
root	read, write	read, write, exec	read, write, exec	read, write, exec
mike	read	read, exec		
roberto	read	read, exec	read, write, exec	
backup	read	read, exec	read, exec	read, exec
. . .	. . .	. . .	. . .	. . .

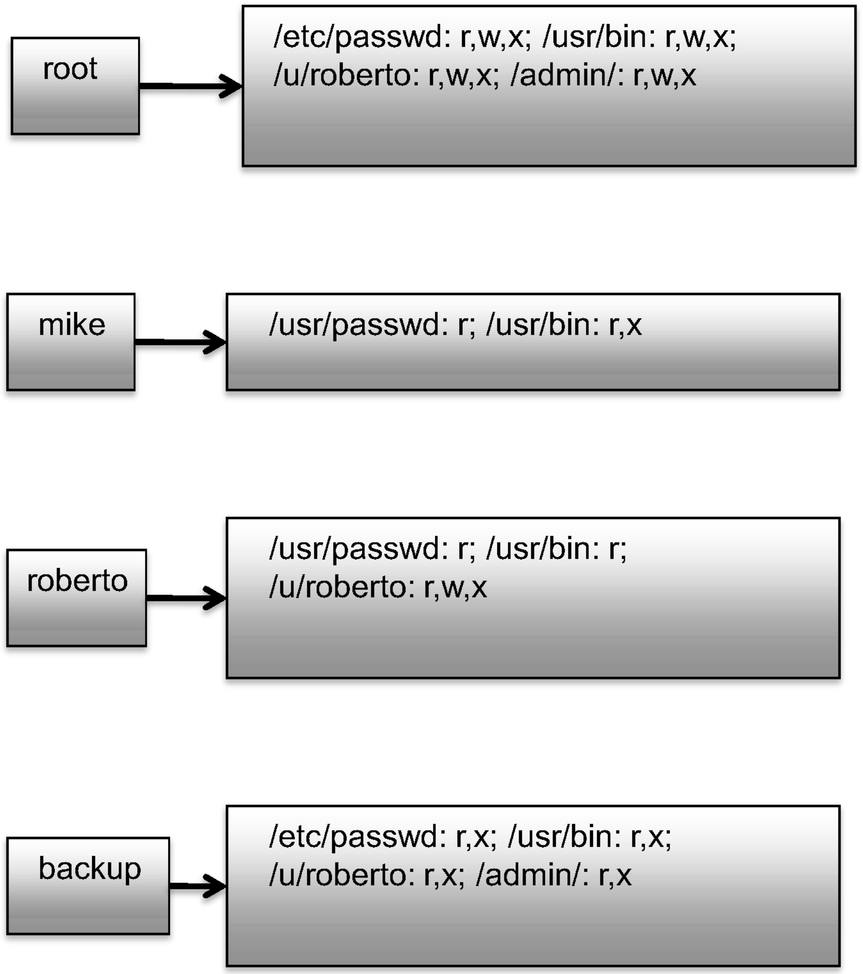
Access Control Lists (ACLs)

An **ACL** defines, for each object, o, a list, L, called o's access control list, which enumerates all the subjects that have access rights for o and, for each such subject, s, gives the access rights that s has for object o.



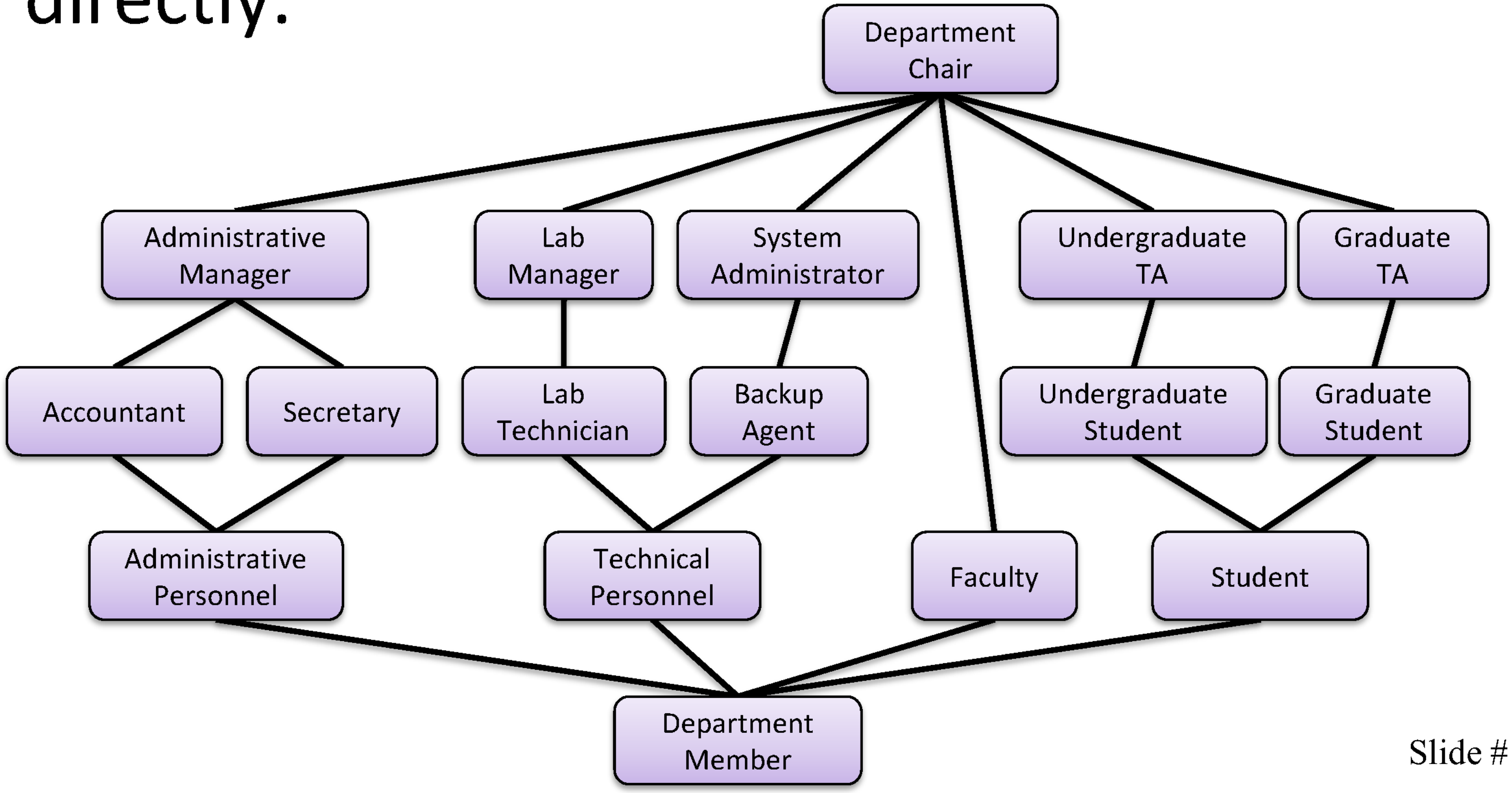
Capabilities

Capabilities take a subject-centered approach to access control. It defines, for each subject s, the list of the objects for which s has nonempty access control rights, together with the specific rights for each such object.


root
/etc/passwd: r,w,x; /usr/bin: r,w,x; /u/roberto: r,w,x; /admin/: r,w,x
mike
/usr/passwd: r; /usr/bin: r,x
roberto
/usr/passwd: r; /usr/bin: r; /u/roberto: r,w,x
backup
/etc/passwd: r,x; /usr/bin: r,x; /u/roberto: r,x; /admin/: r,x

Role-based Access Control

Define **roles** and then specify access control rights for these roles, rather than for subjects directly.



Discretionary and Mandatory

Discretionary Access Control (DAC)

- Users set ACLs on objects OR
- Sysadmins set capabilities for each user.
- Consumer OS like Windows, Linux use DAC.

Mandatory Access Control (MAC)

- Administrator configures access control matrix.
- Access cannot be altered while system is running.
- Examples: chattr, SElinux, Windows MIC

UNIX Access Control Model

- UID
  - integer user ID
  - UID=0 is ***root***
- GID
  - integer group ID
  - Users can belong to multiple groups
- Objects have both a user + group owner.
- System compares object UID with EUID.
  - EUID identical except after su or SETUID.

UNIX File Permissions

Three sets of permissions:

- User owner
- Group owner
- Other (everyone else)

Three permissions per group

- read
- write
- execute

UID 0 can access regardless of permissions.

Files: directories, devices (disks, printers), IPC

UNIX File Permissions

Best-match policy

- OS applies permission set that most closely matches.
- You can be denied access by best match even if you match another set.

Directories

- read = listing of directory
- execute = traversal of directory
- write = add or remove files from directory

Special File Permissions

Each object has set of special permission bits

sticky

- On a directory, means users can only delete files that they own

setuid

- Execute program with EUID = owner's UID

setgid

- Execute program with EGID = owner's GID
- On directories, causes default group owner to be that of directory owner's GID.

Changing Permissions: chmod

Set specifiers

- u = user
- g = group
- o = other

Permissions

- r = read
- w = write
- x = execute

```
# remove other access
chmod  o-rwx *.c

# add group r/w access
chmod g+rw *.c

# allow only you access
chmod u=rwx *
```

Octal Permission Notation

Each set (u,g,o) is represented by an octal digit.

Each permission (r,w,x) is one bit within a digit.

ex: `chmod 0644 file`

- u: rw, g: r, o: r

ex: `chmod 0711 bin`

- u: rwx, g: x, o: x

4	read	setuid
2	write	setgid
1	execute	sticky

Changing Ownership

`newgrp`

- Group owner of files is your default group.
- Changes default group to another group to which you belong.

`chgrp`

- Changes group owner of existing file.

`chown`

- Changes owner of existing file.
- Only root can use this command.

Default Permissions: umask

Determines permissions given to newly created files

Three-digit octal number

- Programs default to 0666
- Umask modifies to: 0666 & ~umask
- ex: umask=022 => file has mode 0644
- ex: umask=066 => file has mode 0600

setuid/setgid

Solution to UNIX ACLs inability to directly handle (user, program, file) triplets.

Process runs with EUID/EGID of file, not of user who spawned the process.

Follow principle of least privilege

- create special user/groups for most purposes

Follow principle of separation of privilege

- keep setuid functions/programs small
- drop privileges when unnecessary

Limitations of Classic ACLs

ACL control list only contains 3 entries

- Limited to one user.
- Limited to one group.

Root (UID 0) can do anything.

POSIX Extended ACLs

Supported by most UNIX/Linux systems.

- Slight syntax differences may exist.

`getfacl`

`setfacl`

- chmod 600 file
- setfacl -m user:gdoor:r-- file
- File unreadable by other, but ACL allows gdoor

Windows NT Access Control

Security IDs (SIDs)
- users
- groups
- hosts

Token: user SID + group SIDs for a subject

ACLs on
- files and directories
- registry keys
- many other objects: printers, IPC, etc.

Standard NT Permissions

Read: read file or contents of a directory

Write: create or write files and directories

Read & Execute: read file and directory attributes, view directory contents, and read files within directory.

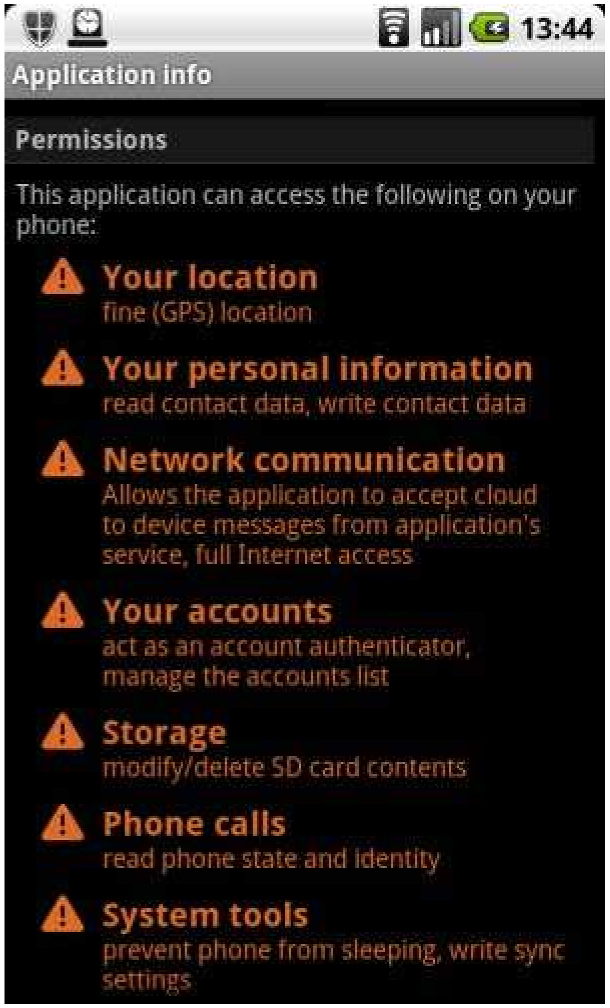
List Folder Contents: RX, but not inherited by files within a folder.

Modify: delete, write, read, and execute.

Full Control: all, including taking ownership and changing permissions

Android: App Level DAC

- Android is a version of Linux for mobile devices.
- Each app runs with its own UID and GID.
  - By default can only access files within its own directory.
  - Android apps request permissions at install time.



Application info

Permissions

This application can access the following on your phone:

Your location
fine (GPS) location

Your personal information
read contact data, write contact data

Network communication
Allows the application to accept cloud to device messages from application's service, full Internet access

Your accounts
act as an account authenticator, manage the accounts list

Storage
modify/delete SD card contents

Phone calls
read phone state and identity

System tools
prevent phone from sleeping, write sync settings

Database Access Control Models

Database access control models

- DAC provided by SQL GRANT, REVOKE stmts.
- MAC provided by db-specific extensions.

Subjects

- Users, Roles

Objects

- Databases, tables, rows, columns, views.

Rights

- Select, insert, update, delete, references, grant.

SQL Access Control Examples

The grant command gives access to a user

```
grant select on students to james
```

or a role:

```
grant select, insert, update on grades
  to faculty
```

and includes power to grant options:

```
grant insert on students to registrar
  with grant option
```

The revoke command removes access

```
remove insert on grades from faculty
```

Database App Security Models

One Big Application User

- Single user account for all accesses.
- DB sees no difference between app users.
- Vulnerability that compromises one app user can lead to exposure of data of all users.

Application users are database users

- Each app user has a database user account too.
- Restrict users to accessing their own data even if there is a vulnerability in the application.

Immutable Files

Immutable Files on FreeBSD (MAC)

- `chflags +noschg`
- Cannot be removed by root in securelevel >0

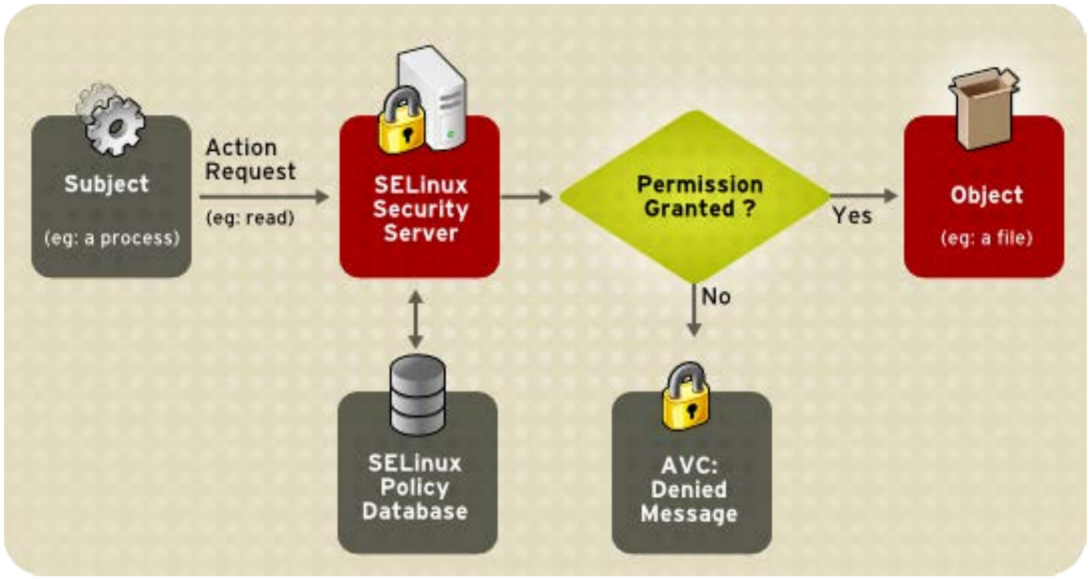
Immutable Files on Linux (DAC)

- `chattr +i`
- Cannot delete, rename, write to, link to.
- Applies to root too, unlike normal permissions.
- Only root can remove immutable flag.

SELinux

SELinux is a fine-grained MAC system for Linux.

- Operates at process level.
- Files have security contexts.
- Process can access file only if allowed by policy.
- Policy can only be changed after reboot.



SELinux Expansions

SEPostgreSQL

- Enforces MAC on Postgres database tables.
- SECURITY LABEL SQL statement to label db objs.
- Uses system SELinux configuration.

SEAndroid

- SELinux for Android platform +
- MAC for Android permissions and inter-component communication (ICC).

Hardware Protection

Confidentiality

– Processes cannot read memory space of kernel or of other processes without permission.

Integrity

– Processes cannot write to memory space of kernel or of other processes without permission.

Availability

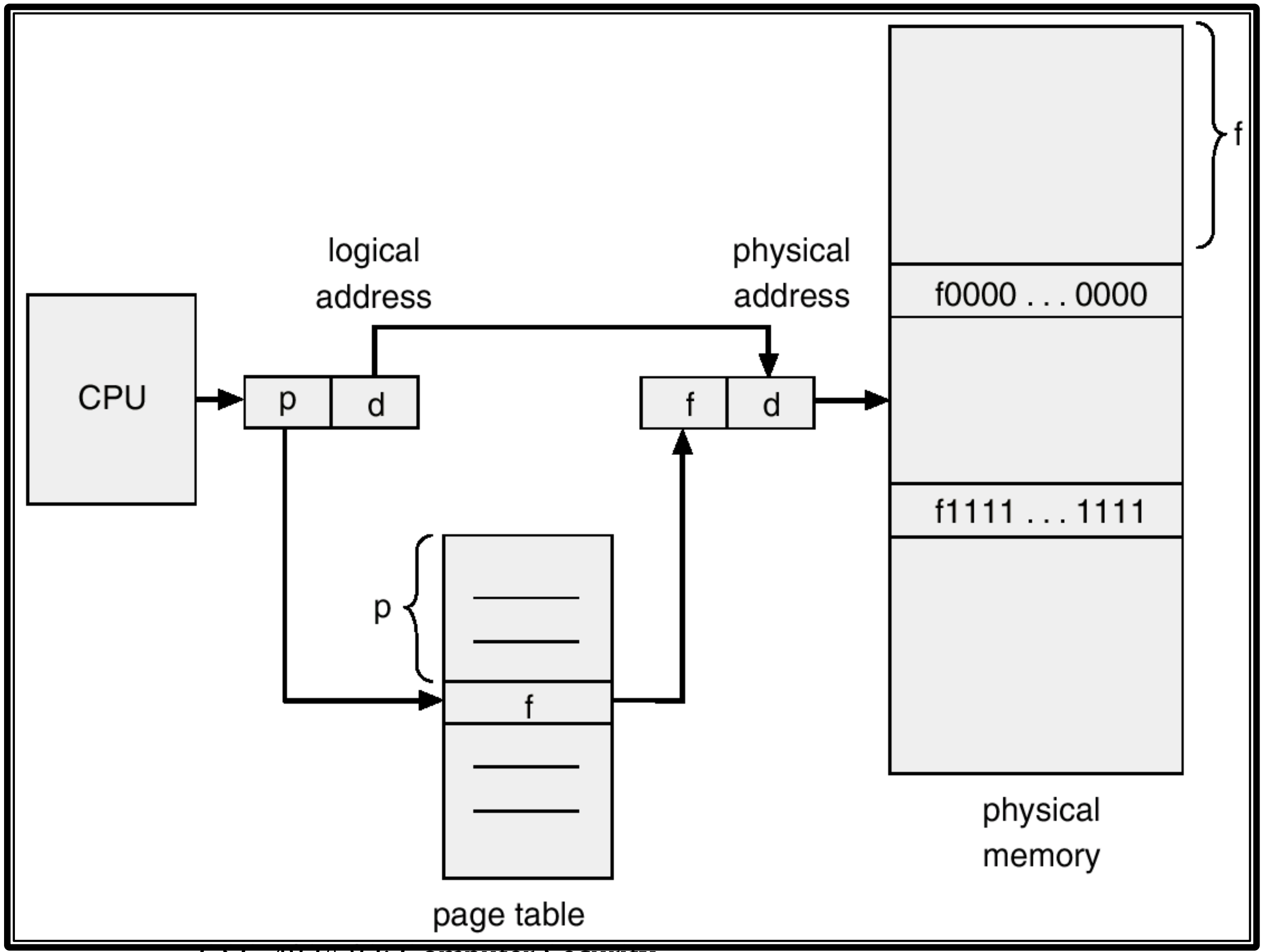
– One process cannot deny access to CPU or other resources to kernel or other processes.

Hardware Mechanisms: VM

Each process has its own address space.

- Prevents processes from accessing memory of kernel or other processes.
  - Attempted violations produce page fault exceptions.
- Implemented using a page table.
- Page table entries contain access control info.
  - Read
  - Write
  - Execute (not separate on Intel CPUs)
  - Supervisor (only accessible in supervisor mode)

VM Address Translation



Hardware Mechanisms: Rings

Protection Rings.

- Lower number rings have more rights.
- Intel CPUs have 4 rings
  - Ring 0 is supervisor mode.
  - Ring 3 is user mode.
  - Most OSes do not use other rings.
- Multics used 64 protection rings.
  - Different parts of OS ran in different rings.
  - Procedures of same program could have different access rights.

Hardware: Privileged Instructions

Only can be used in supervisor mode.

Setting address space

- MOV CR3

Enable/disable interrupts

- CLI, STI

Reading/writing to hardware

- IN, OUT

Switch from user to supervisor mode on interrupt.

Hardware: System Timer

Processes can voluntarily give up control to OS via system calls to request OS services.

- SYSENTER, INT 2e

Timer interrupt

- Programmable Interval Timer chip.
- Happens every 1-100 OS, depending on OS.
- Transfers control from process to OS.
- Ensures no process can deny availability of machine to kernel or other processes.

Key Points

1. Access Control models
    1. Access Control Matrix
    2. ACL implements ACM by column (object based)
    3. Capability implements ACM by row (subject based)
2. Types of Access Control
    1. Mandatory (MAC)
    2. Discretionary (DAC)
    3. Role Based (RBAC)
3. UNIX ACLs
    1. Chmod, umask, POSIX ACLs
4. Hardware Access Controls
    1. Rings, VM, privileged instructions, system timer

References

1. Anderson, *Security Engineering 2nd Edition*, Wiley, 2008.
2. Bishop, *Computer Security: Art and Science*, Addison-Wesley, 2002.
3. Goodrich and Tammasia, *Introduction to Computer Security*, Pearson, 2011.