CS 547: Foundation of Computer Security

S. Tripathy
IIT Patna

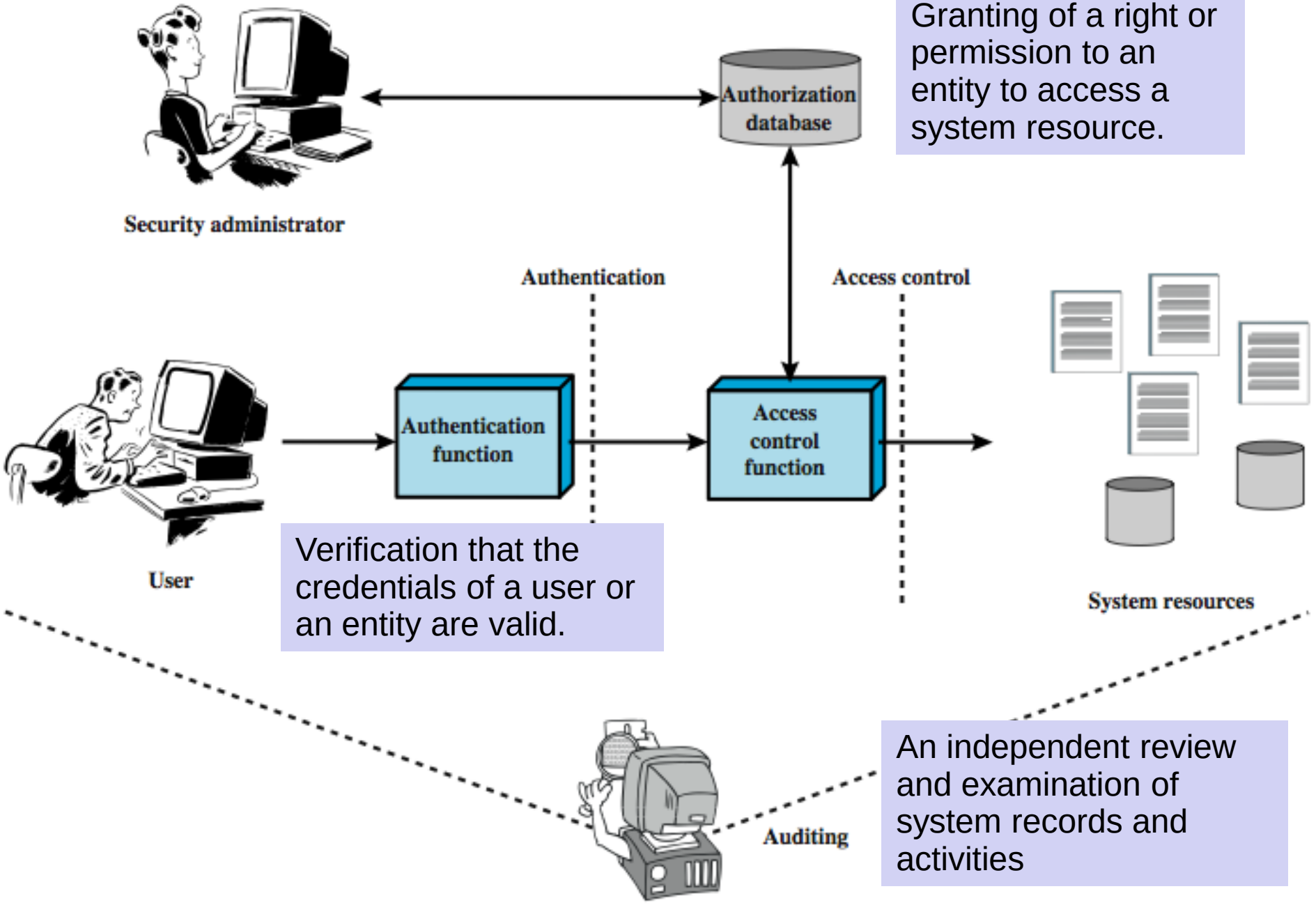
Previous Class

- Authentication
  - Password
    - Linux and Windows machines
  - Remote Authentication

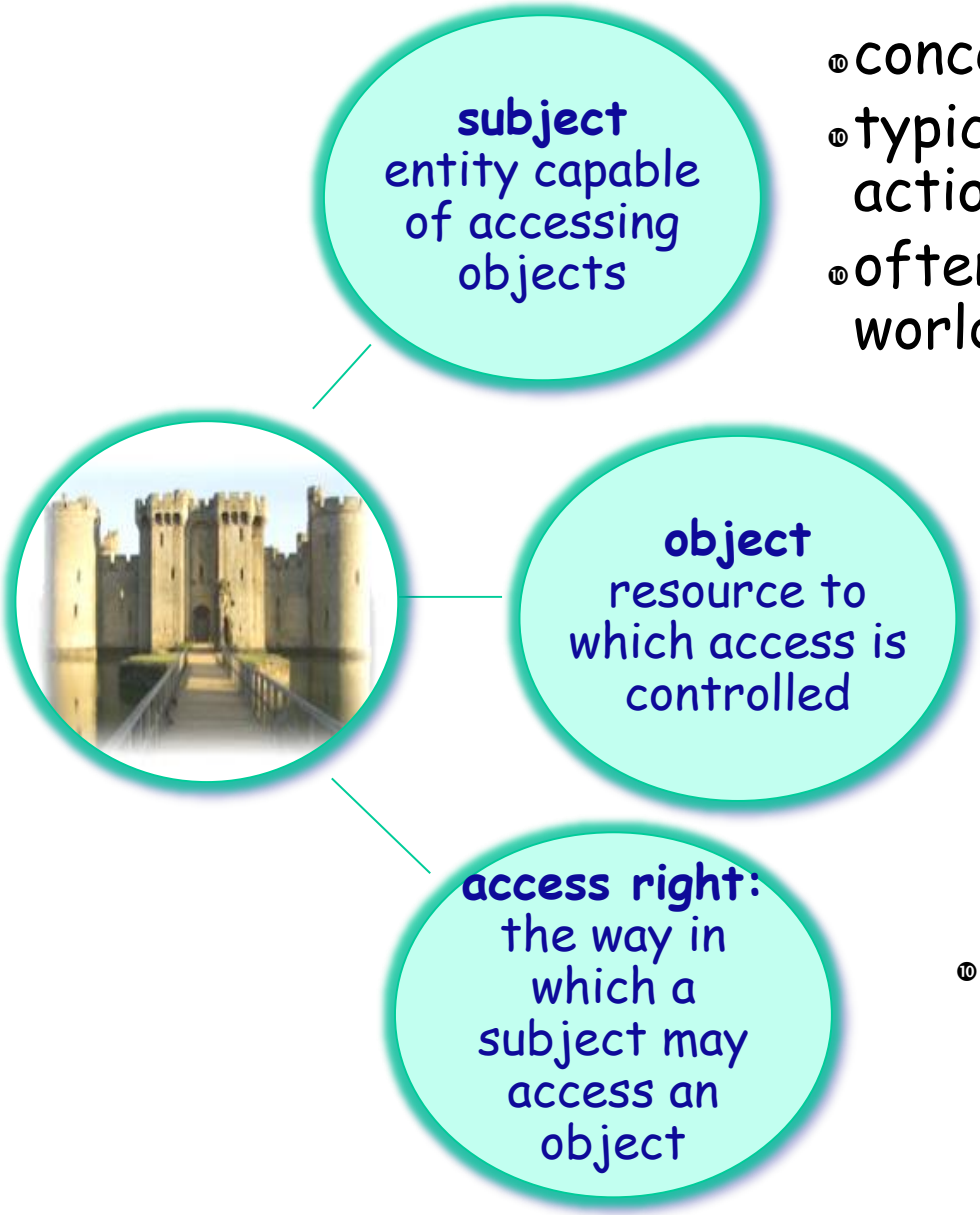
Present class

- Access Control
  - Access Control Matrix
  - Access Control Lists vs. Capabilities
  - Discretionary Access Control
    - Unix File Access Control
  - Mandatory Access Control
  - Role-Based Access Control

Access Control Principles



Access Control Basic Elements

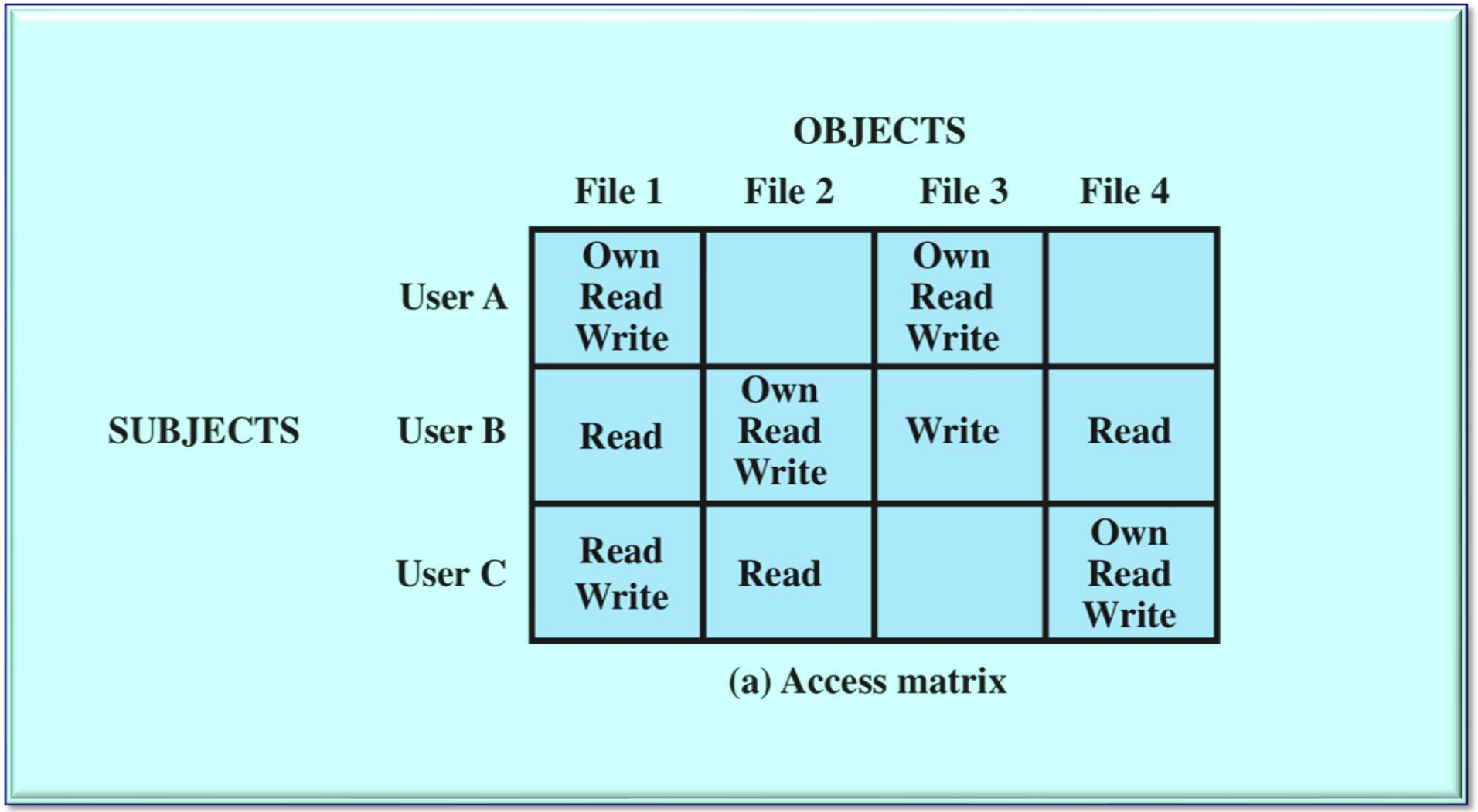


- concept equates with that of process
- typically held accountable for the actions they initiate
- often have three classes: owner, group, world

- entity used to contain and/or receive information
- protection depends on the environment in which access control operates

- e.g. read, write, execute, delete, create, search

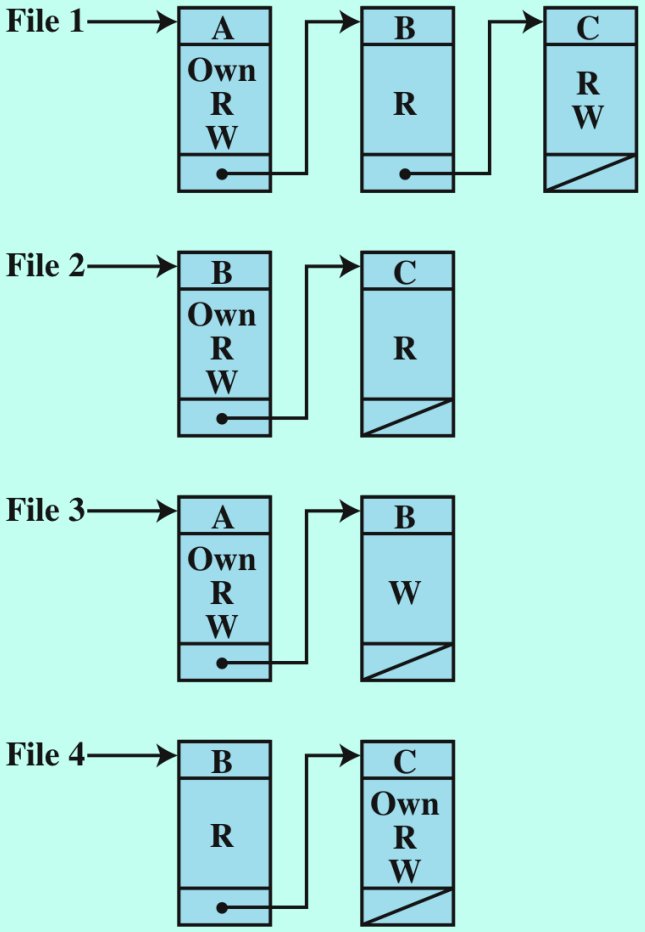
Access Matrix



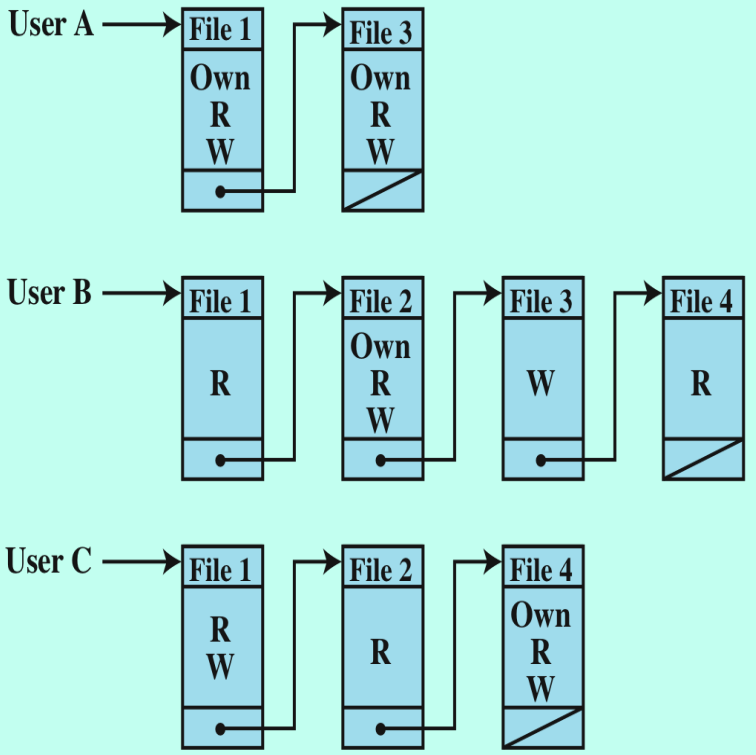
		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

(a) Access matrix

Example of Access Control Structures



(b) Access control lists for files of part (a)

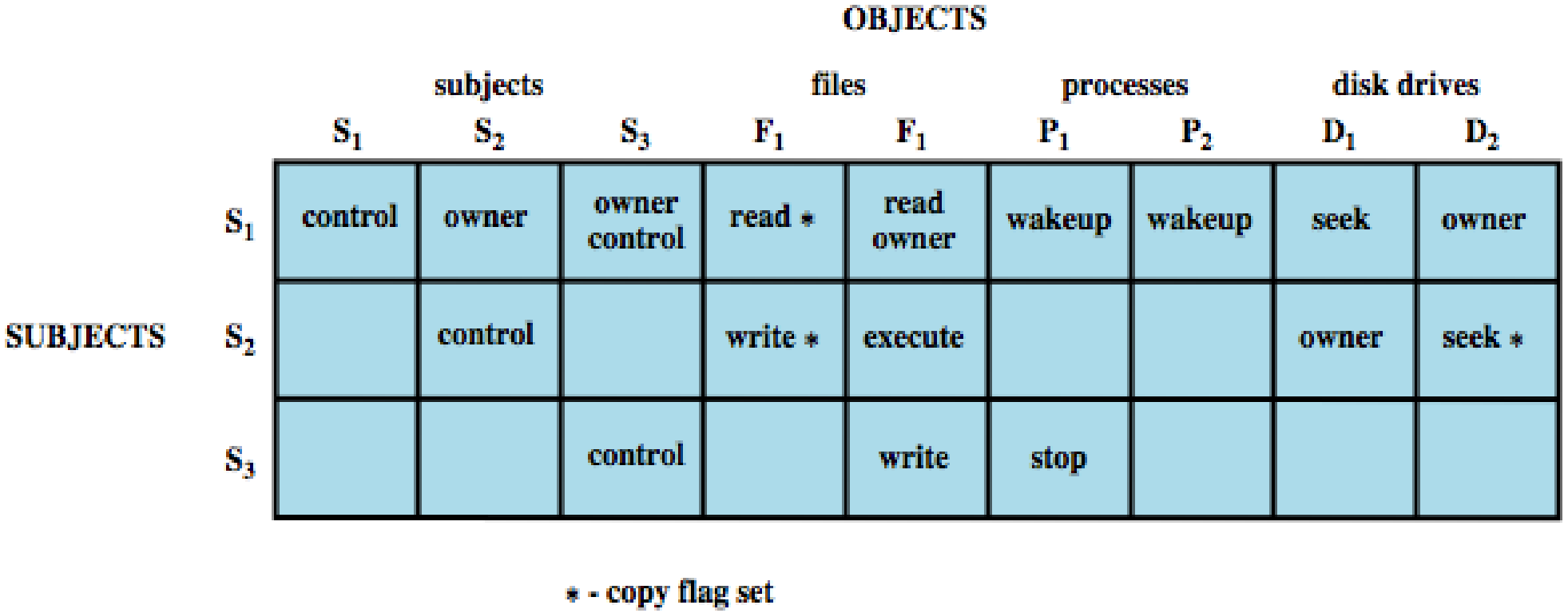


(c) Capability lists for files of part (a)

Authorization Table

Subject	Access Mode	Object
A	Own	File 1
A	Read	File 1
A	Write	File 1
A	Own	File 3
A	Read	File 3
A	Write	File 3
B	Read	File 1
B	Own	File 2
B	Read	File 2
B	Write	File 2
B	Write	File 3
B	Read	File 4
C	Read	File 1
C	Write	File 1
C	Read	File 2
C	Own	File 4
C	Read	File 4
C	Write	File 4

Extended Access Control Matrix

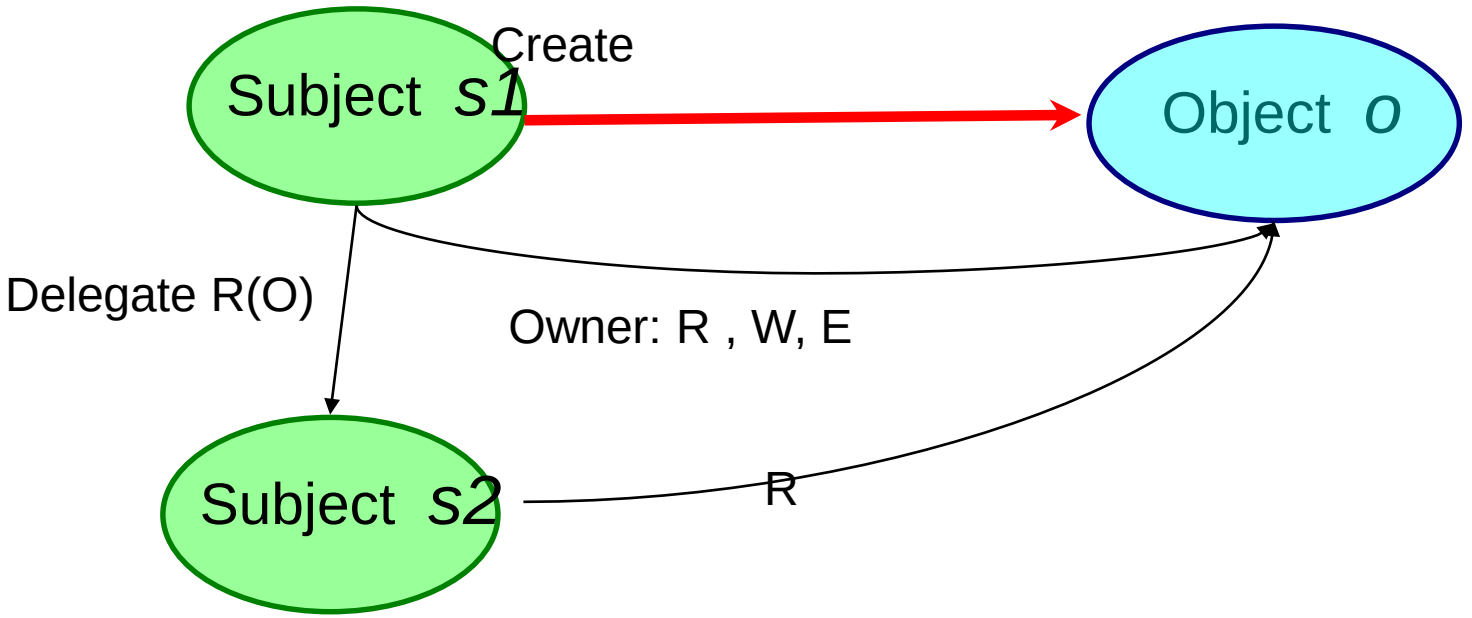


		OBJECTS								
		subjects			files		processes		disk drives	
		S_1	S_2	S_3	F_1	F_1	P_1	P_2	D_1	D_2
SUBJECTS	S_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	S_2		control		write *	execute			owner	seek *
	S_3			control		write	stop			

* - copy flag set

Access Control Models: DAC

- DAC model enforces access control based on user identities, object ownership and permission delegation. The owner of an object may delegate the permission of the object to another user.



DAC Pattern Structure

- **Subject** represents a process acting on behalf of a user in a computer-based system. It could also be another computer system, a node or a set of attributes.

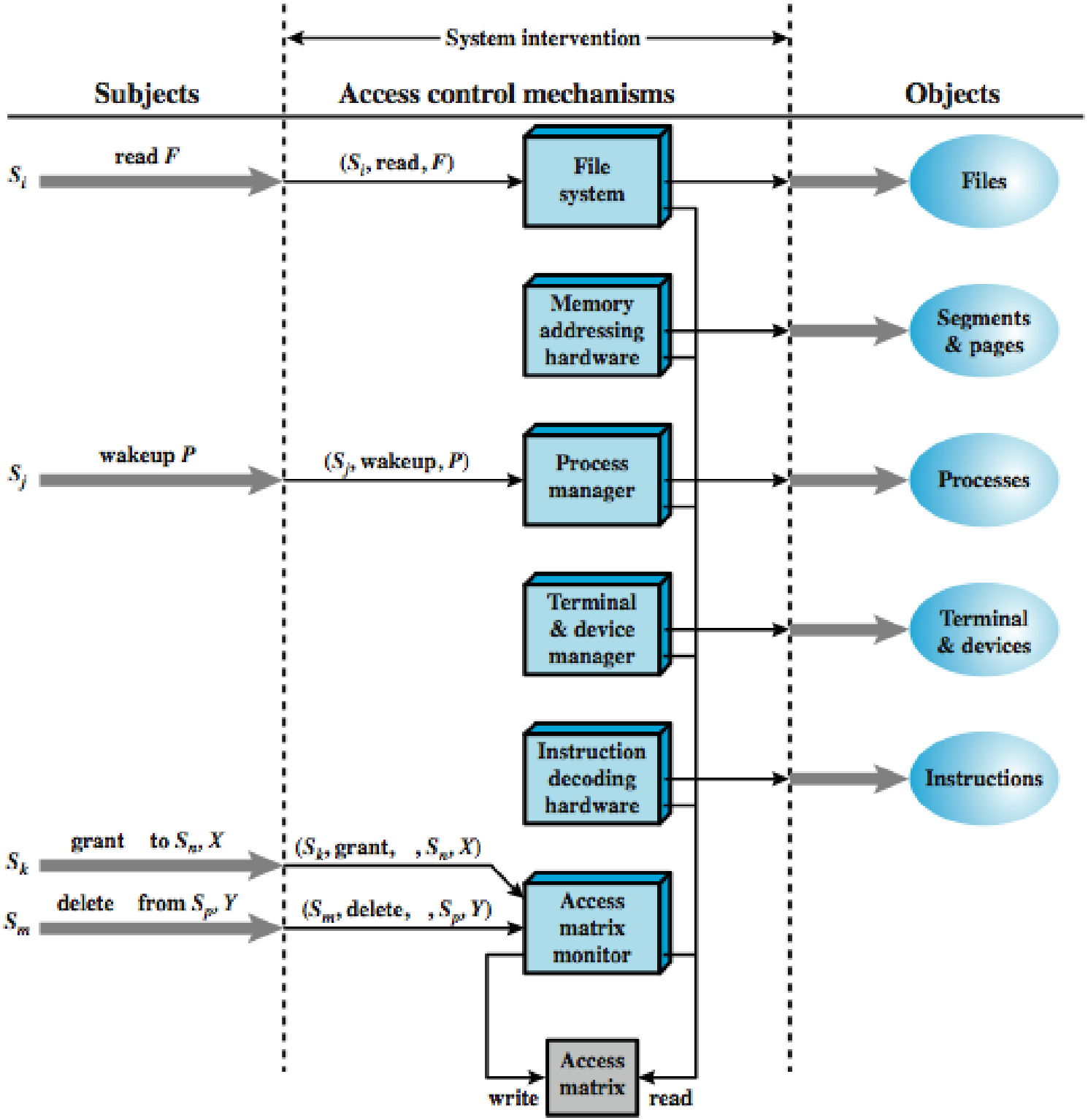
- **Permission** represents an authorization to carry out an action on the system. In general, Access Control Lists (ACLs) are used to describe DAC policies for its ease in reviewing. An ACL shows permissions in terms of objects, users and access rights.

- **ReferenceMonitor** checks permission for an access request based on DAC policies. If the user has permission to the object, the requested operation may be performed.

Access Control Entries and Lists

- An Access Control List (ACL) for a resource (e.g., a file or folder) is a sorted list of zero or more Access Control Entries (ACEs)
- An ACE refers to specifies that a certain set of accesses (e.g., read, execute and write) to the resources is allowed or denied for a user or group
- Examples of ACEs for folder "Bob's CS547 Grades"
  - Bob; Read; Allow
  - TAs; Read; Allow
  - Bob; Write; Deny
  - TAs; Write; Allow

Access Control Function



Rule	Command (by S_0)	Authorization	Operation
R1	**transfer** $\left\{\begin{matrix}\alpha* \\ \alpha\end{matrix}\right\}$ **to** S, X	'α*' in $A[S_0, X]$	store $\left\{\begin{matrix}\alpha* \\ \alpha\end{matrix}\right\}$ in $A[S, X]$
R2	**grant** $\left\{\begin{matrix}\alpha* \\ \alpha\end{matrix}\right\}$ **to** S, X	'owner' in $A[S_0, X]$	store $\left\{\begin{matrix}\alpha* \\ \alpha\end{matrix}\right\}$ in $A[S, X]$
R3	**delete** α **from** S, X	'control' in $A[S_0, S]$ or 'owner' in $A[S_0, X]$	delete α from $A[S, X]$
R4	$w \leftarrow$ **read** S, X	'control' in $A[S_0, S]$ or 'owner' in $A[S_0, X]$	copy $A[S, X]$ into w
R5	**create object** X	None	add column for X to A; store 'owner' in $A[S_0, X]$
R6	**destroy object** X	'owner' in $A[S_0, X]$	delete column for X from A
R7	**create subject** S	none	add row for S to A; execute **create object** S; store 'control' in $A[S, S]$
R8	**destroy subject** S	'owner' in $A[S_0, S]$	delete row for S from A; execute **destroy object** S

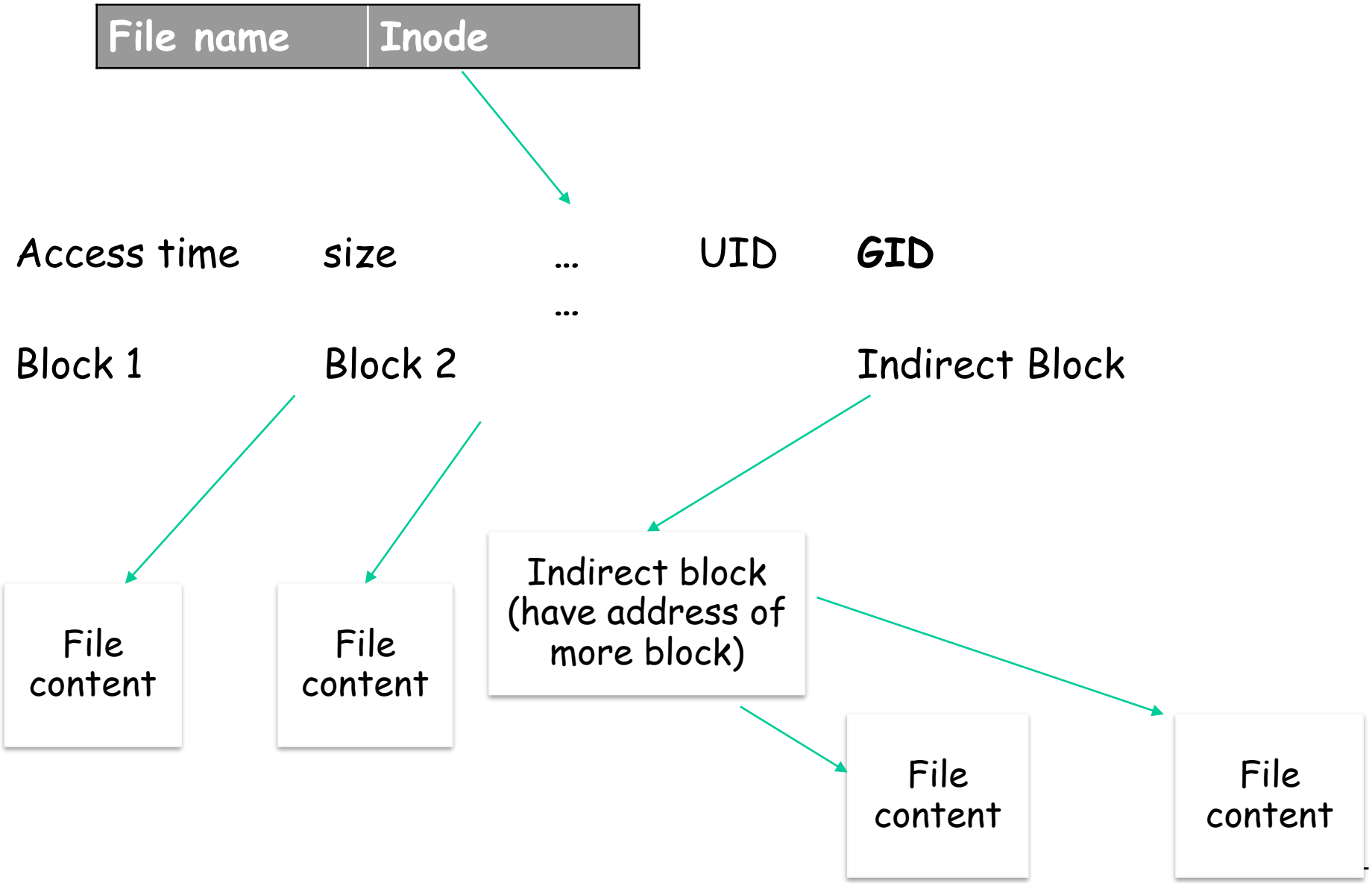
Access Control System Commands

Linux File Access Control

- File Access Control for:
  - Files
  - Directories
  - Therefore…
    - \dev\ : *devices*
    - \mnt\ : *mounted file systems*
    - What else? *Sockets, pipes, symbolic links…*

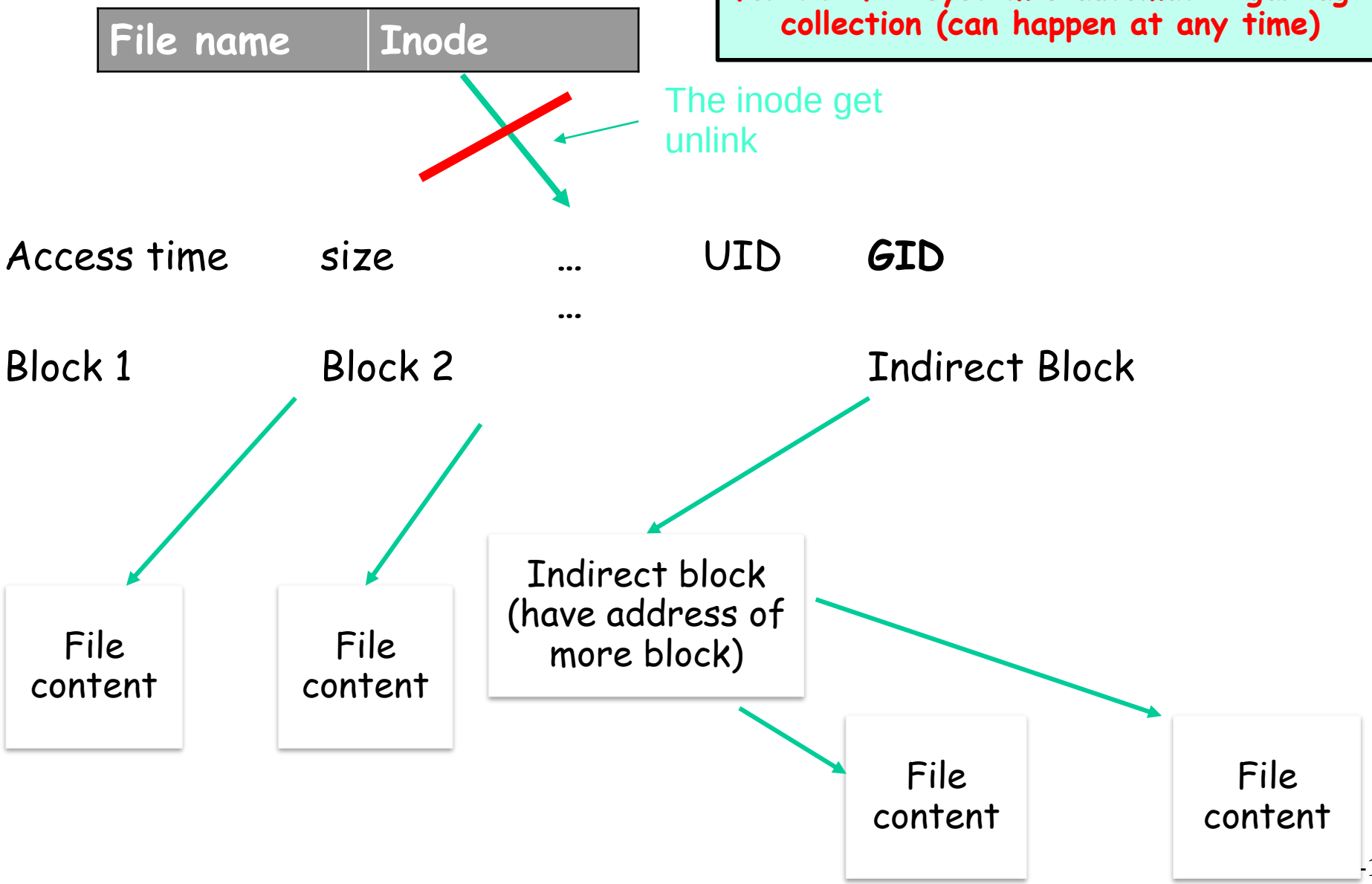
Because of the way devices and mounted file systems are represented in Linux as part of the file system, they are also covered by the same access control scheme as normal files.

Linux File system architecture



General delete behaviour

Data remains on disk, and is available for the file system's automated garbage collection (can happen at any time)



Delete behaviour

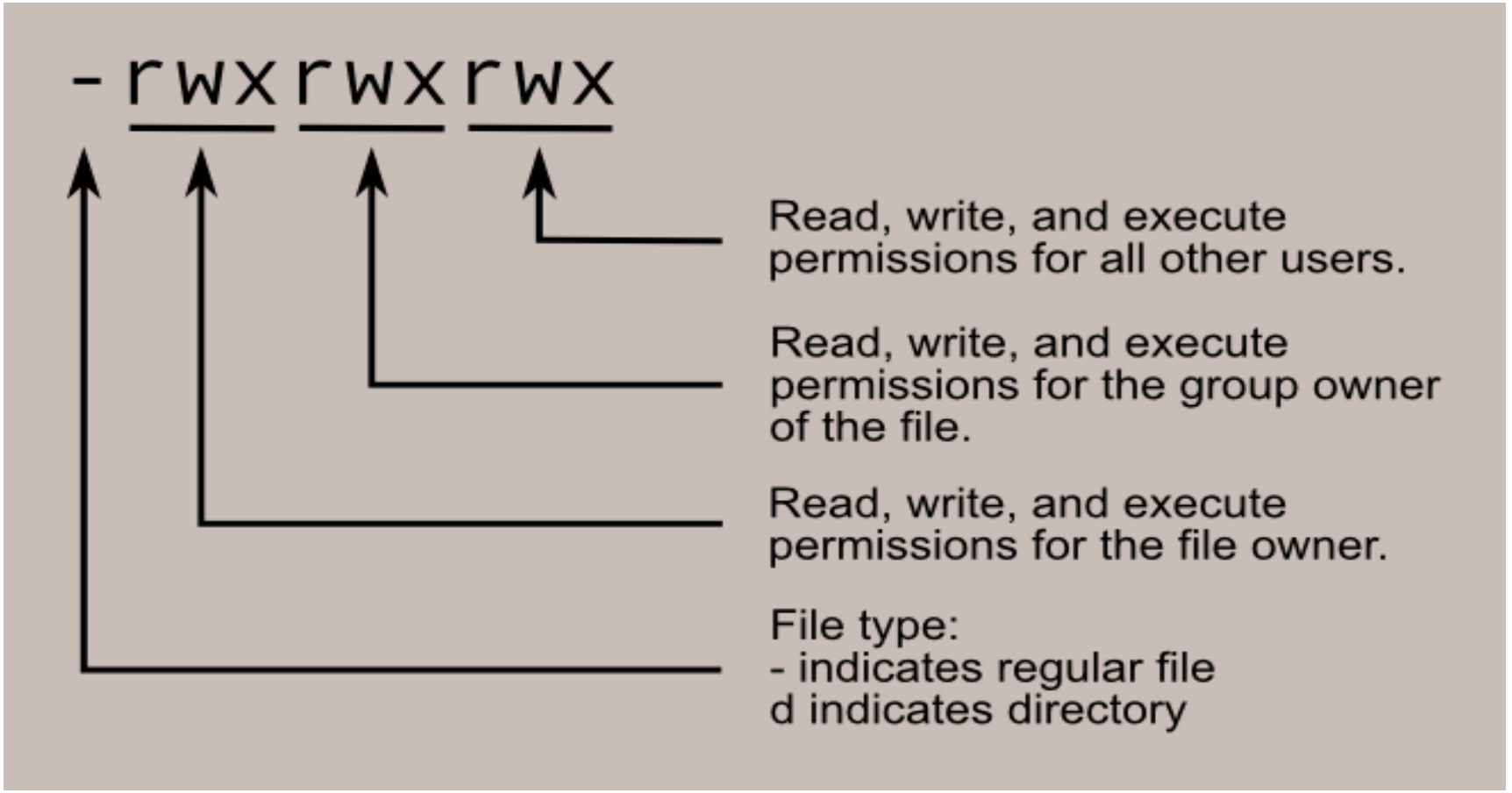
- When we delete a file the only Inode link will break i.e. during deletion Data gets left on the disk, and the inode is just unlinked.

- That's how data recovery software work, because content are not deleted only pointer's are unlinked.

Managing File and Directory Permissions

- Mode: Inode Section that stores permissions
- Three sections, based on the user(s) that receive the permission:
  - User permissions: Owner
  - Group permissions: Group owner
  - Other permissions: Everyone on system
- Three regular permissions may be assigned to each user:
  - Read
  - Write
  - Execute

Interpreting the Mode



Permissions Examples (Regular Files)

-rw-r—r--	read/write for owner, read-only for everyone else
-rw-r-----	read/write for owner, read-only for group, forbidden to others
-rwx------	read/write/execute for owner, forbidden to everyone else
-r--r--r--	read-only to everyone, including owner
-rwxrwxrwx	read/write/execute to everyone

Permissions Examples (Directories)

drwxr-xr-x	all can enter and list the directory, only owner can add/delete files
drwxrwx---	full access to owner and group, forbidden to others
drwx--x---	full access to owner, group can access known filenames in directory, forbidden to others
-rwxrwxrwx	full access to everyone

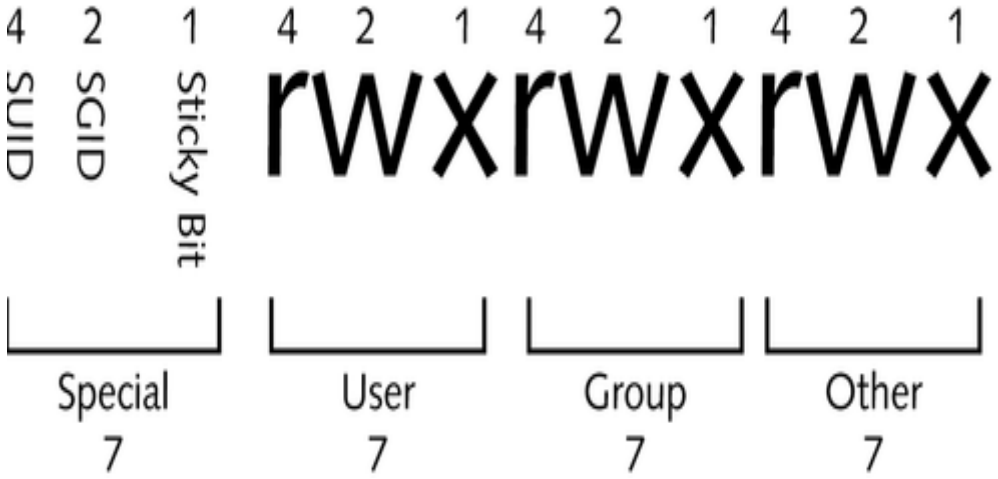
Changing permissions

- chmod (change mode) command: Change mode (permissions) of files or directories

- Permissions stored in a file's or a directory's inode.

Special permissions

- SUID (Set User ID)
- SGID(Set Group ID)
- Sticky Bit



Special with regular permission

Thanks