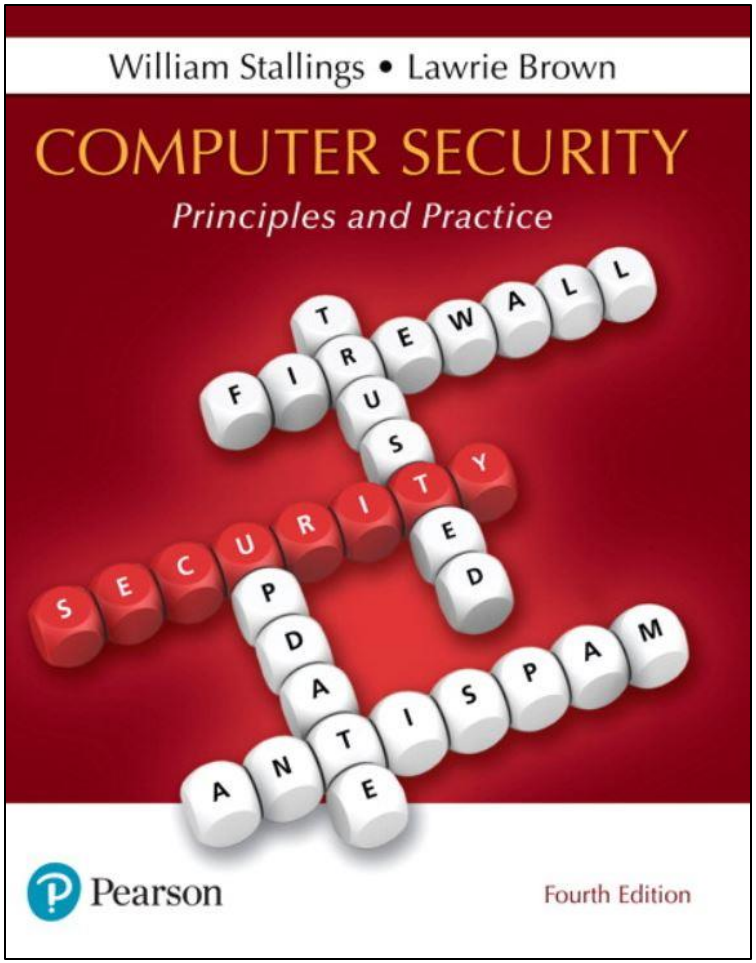
Computer Security: Principles and Practice

Fourth Edition



Chapter 4

Access Control

Access Control Definitions (1 of 2)

NISTIR 7298 defines access control as:

"the process of granting or denying specific requests to: (1) obtain and use information and related information processing services; and (2) enter specific physical facilities"

Access Control Definitions (2 of 2)

RFC 4949 defines access control as:

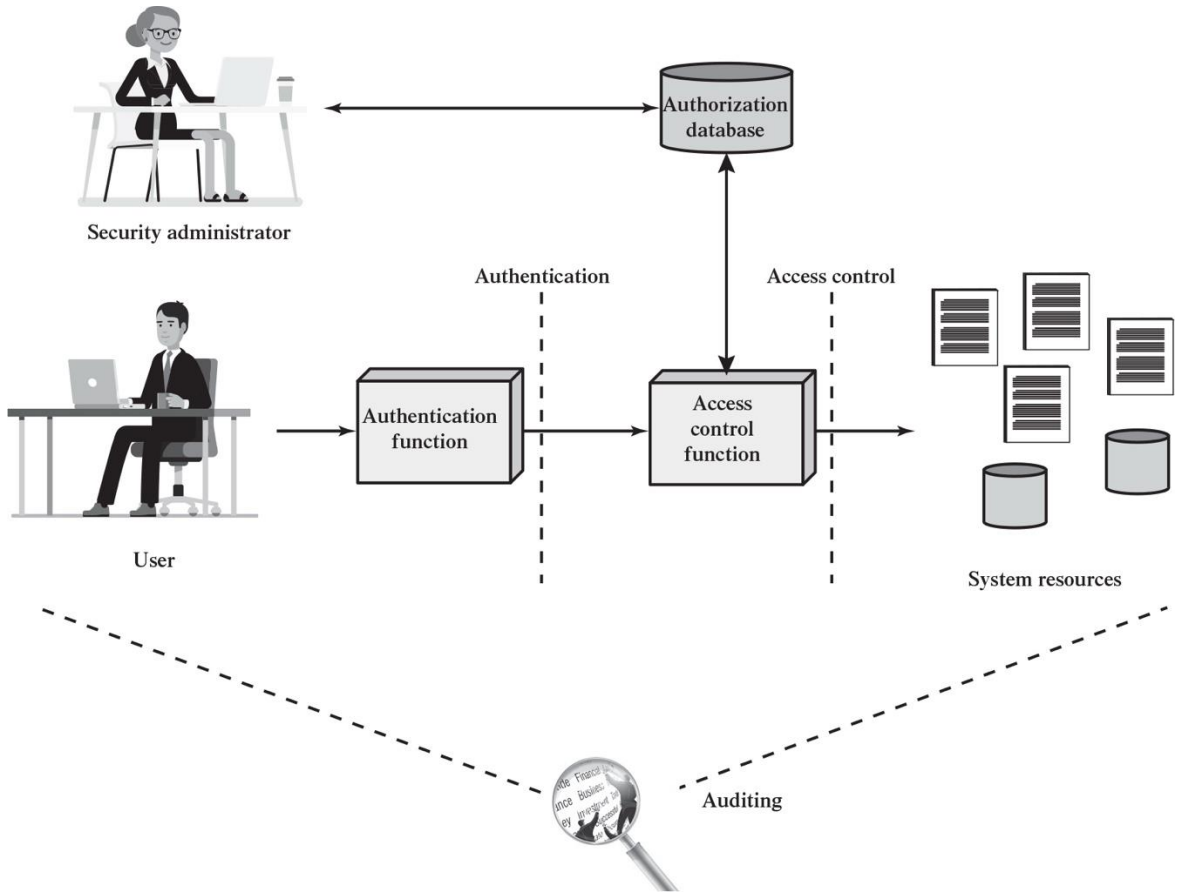
"a process by which use of system resources is regulated according to a security policy and is permitted only by authorized entities (users, programs, processes, or other systems) according to that policy"

Access Control Principles

- In a broad sense, all of computer security is concerned with access control
- RFC 4949 defines computer security as:

"measures that implement and assure security services in a computer system, particularly those that assure access control service"

Figure 4.1 Relationship Among Access Control and Other Security Functions



Source: Based on [SAND94].

Access Control Policies

- Discretionary access control (DAC)
  - Controls access based on the identity of the requestor and on access rules (authorizations) stating what requestors are (or are not) allowed to do
- Mandatory access control (MAC)
  - Controls access based on comparing security labels with security clearances
- Role-based access control (RBAC)
  - Controls access based on the roles that users have within the system and on rules stating what accesses are allowed to users in given roles
- Attribute-based access control (ABAC)
  - Controls access based on attributes of the user, the resource to be accessed, and current environmental conditions

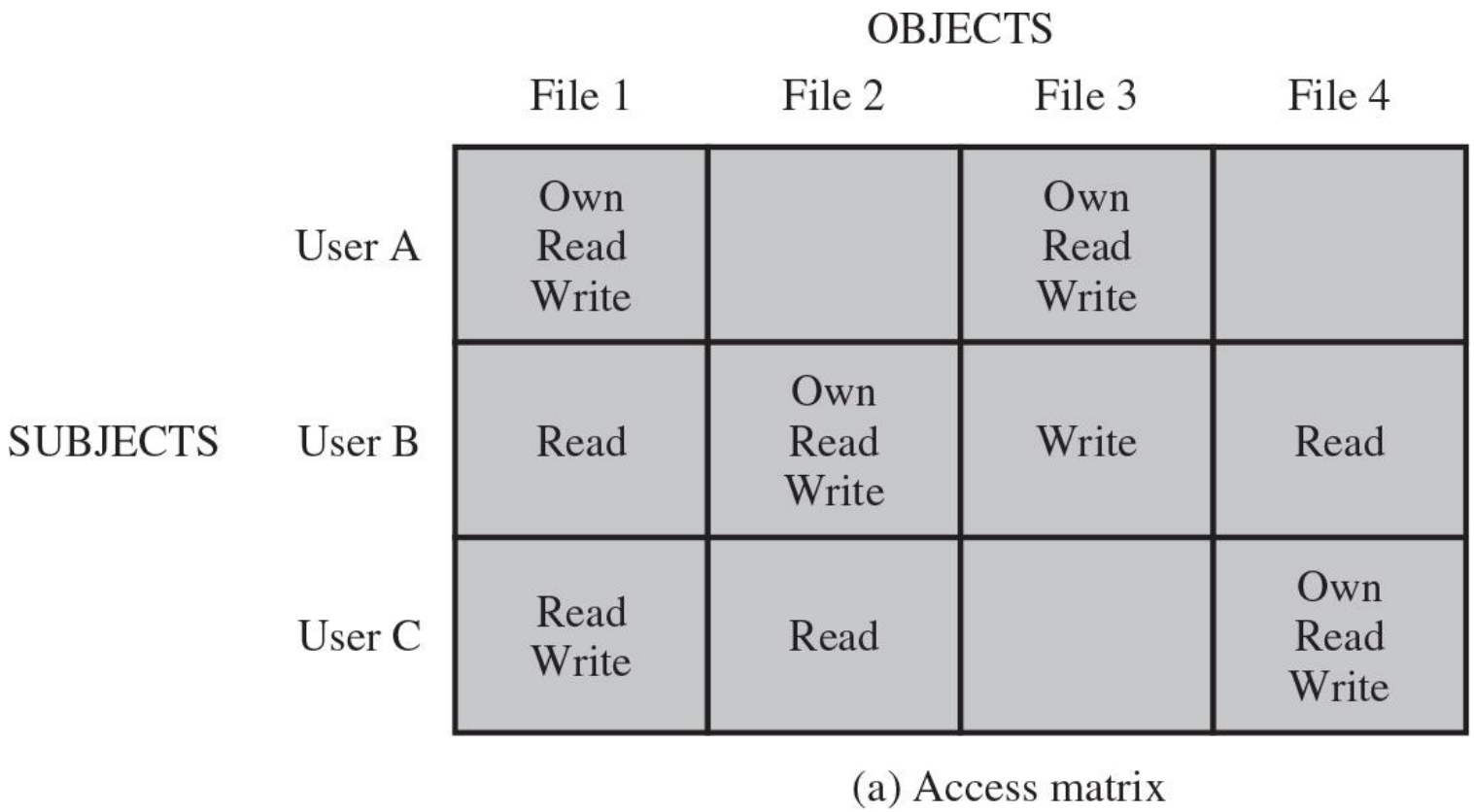
Subjects, Objects, and Access Rights

- Subject
  - An entity capable of accessing objects
  - Three classes
    - Owner
    - Group
    - World
- Object
  - A resource to which access is controlled
  - Entity used to contain and/or receive information
- Access right
  - Describes the way in which a subject may access an object
  - Could include:
    - Read
    - Write
    - Execute
    - Delete
    - Create
    - Search

Discretionary Access Control (DAC)

- Scheme in which an entity may be granted access rights that permit the entity, by its own violation, to enable another entity to access some resource
- Often provided using an access matrix
  - One dimension consists of identified subjects that may attempt data access to the resources
  - The other dimension lists the objects that may be accessed
- Each entry in the matrix indicates the access rights of a particular subject for a particular object

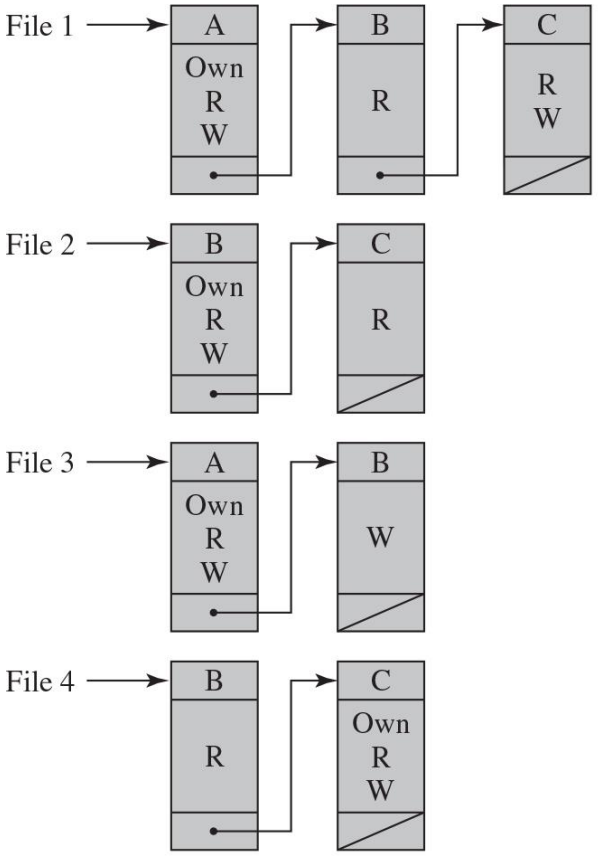
Figure 4.2 Example of Access Control Structures (1 of 2)



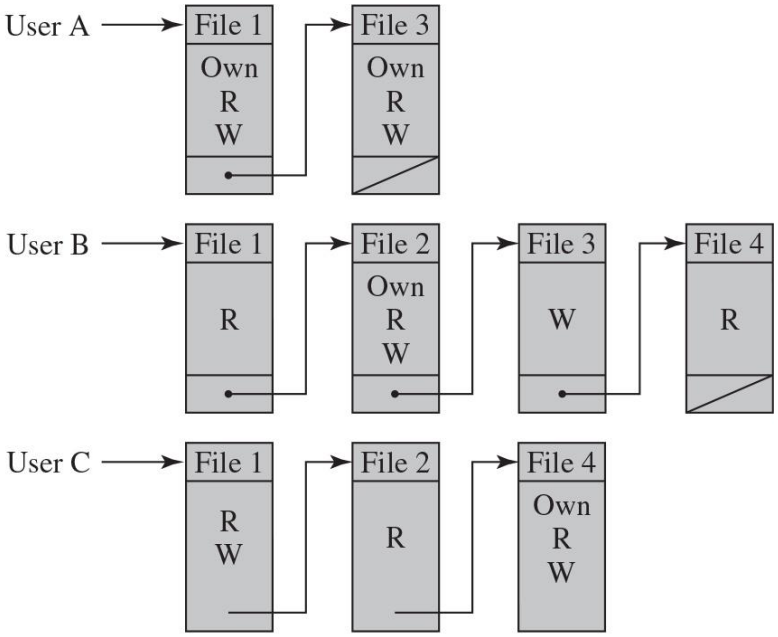
		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

(a) Access matrix

Figure 4.2 Example of Access Control Structures (2 of 2)



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



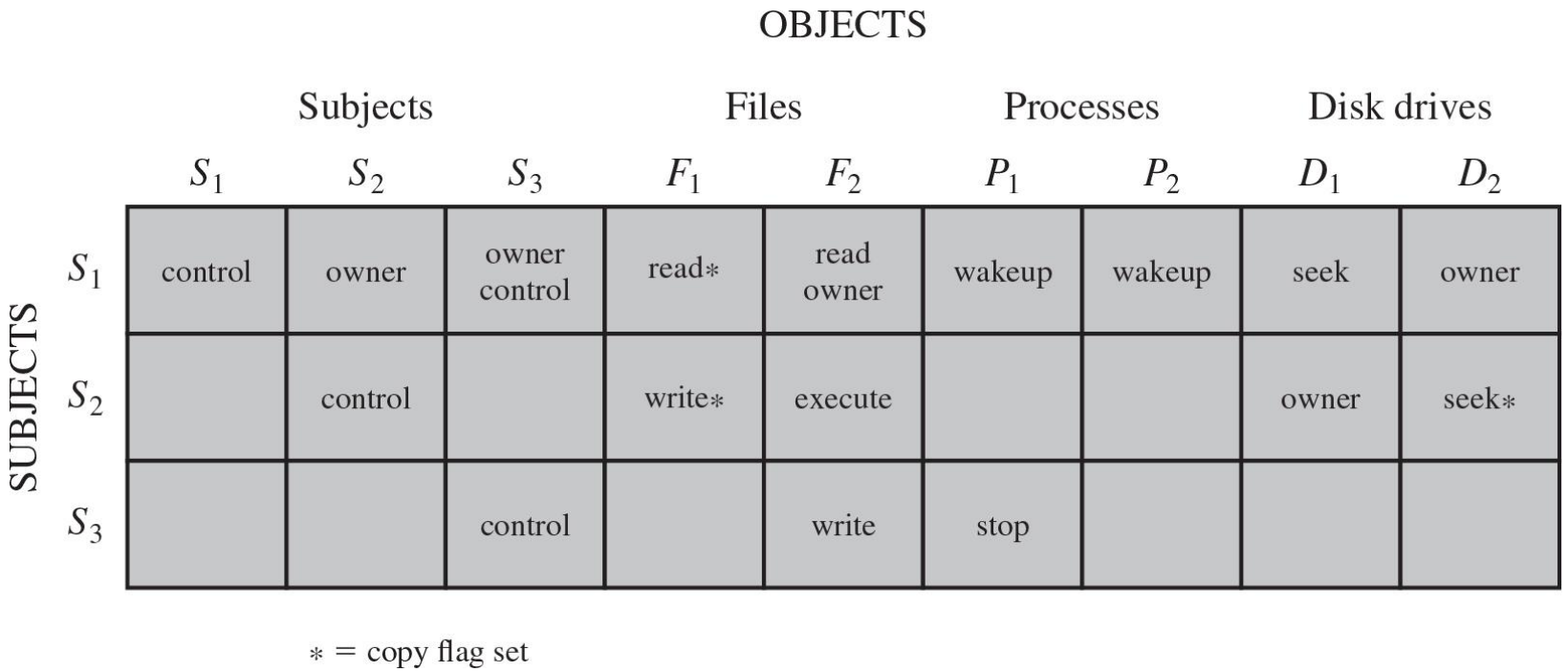
(c) Capability lists for files of part (a)

Table 4.2 Authorization Table for Files in Figure 4.2

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

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

Figure 4.3 Extended Access Control Matrix



OBJECTS

SUBJECTS	Subjects			Files		Processes		Disk drives	
	S_1	S_2	S_3	F_1	F_2	P_1	P_2	D_1	D_2
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

Figure 4.4 An Organization of the Access Control Function

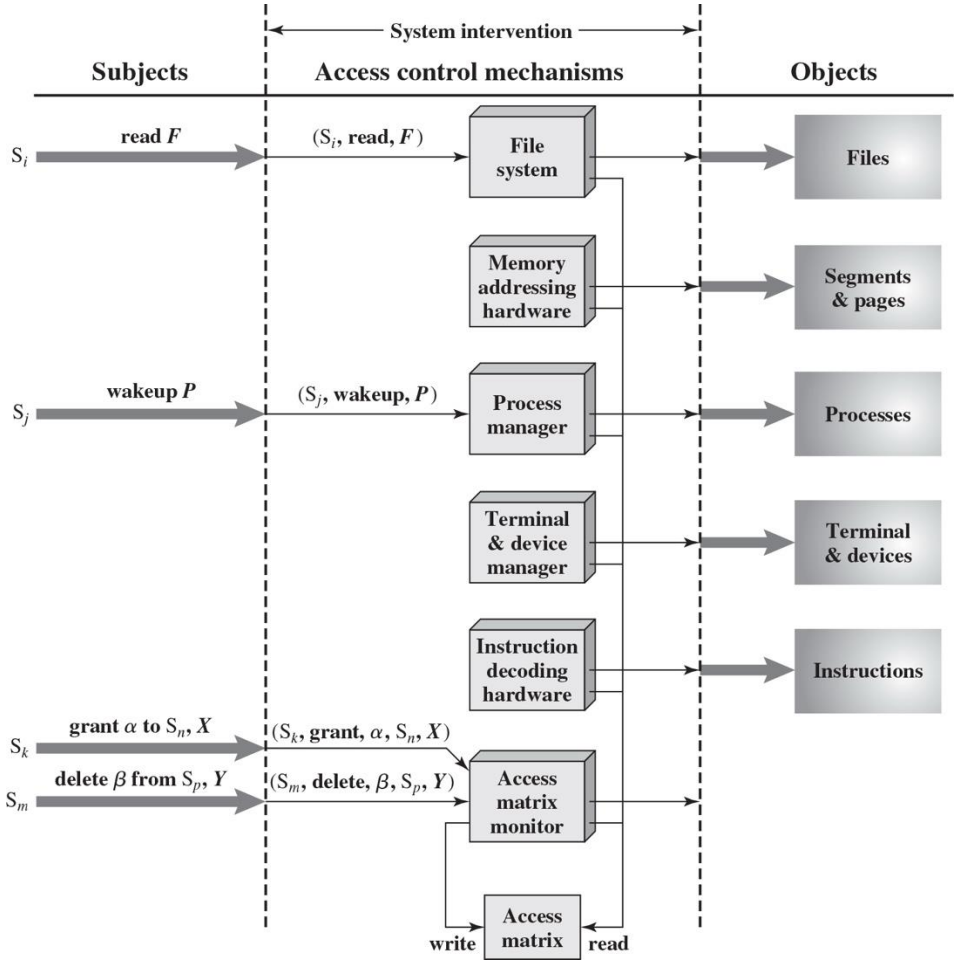


Table 4.3 Access Control System Commands (1 of 2)

Rule	Command (by S_0)	Authorization	Operation
R1	**transfer** $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ **to** S, X	"α*" in $A[S_0, X]$	store $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ in $A[S, X]$
R2	**grant** $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ **to** S, X	'owner' in $A[S_0, X]$å	store $\begin{Bmatrix}\alpha* \\ \alpha\end{Bmatrix}$ 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 $A[S_0, X]$

Table 4.3 Access Control System Commands (2 of 2)

Rule	Command (by S_0)	Authorization	Operation
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*

Protection Domains

- Set of objects together with access rights to those objects
- More flexibility when associating capabilities with protection domains
- In terms of the access matrix, a row defines a protection domain
- User can spawn processes with a subset of the access rights of the user
- Association between a process and a domain can be static or dynamic
- In user mode certain areas of memory are protected from use and certain instructions may not be executed
- In kernel mode privileged instructions may be executed and protected areas of memory may be accessed

UNIX File Access Control (1 of 3)

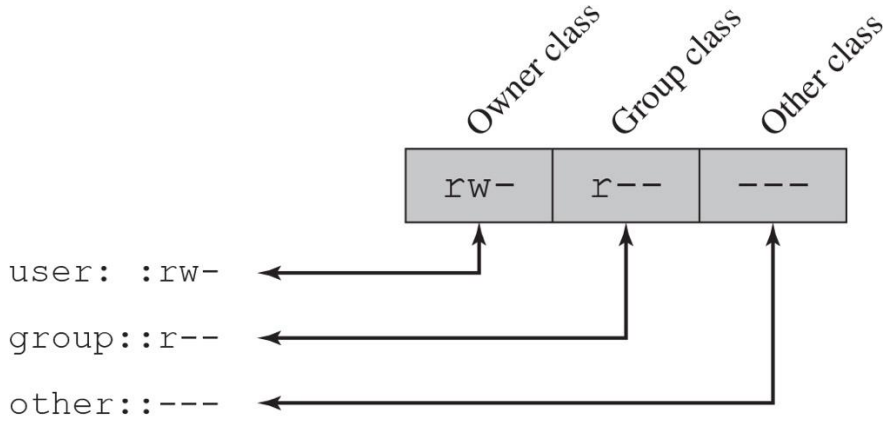
- UNIX files are administered using inodes (index nodes)
  - Control structures with key information needed for a particular file
  - Several file names may be associated with a single inode
  - An active inode is associated with exactly one file
  - File attributes, permissions and control information are sorted in the inode
  - On the disk there is an inode table, or inode list, that contains the inodes of all the files in the file system
  - When a file is opened its inode is brought into main memory and stored in a memory resident inode table

UNIX File Access Control (2 of 3)

- Directories are structured in a hierarchical tree
  - May contain files and/or other directories
  - Contains file names plus pointers to associated inodes

UNIX File Access Control (3 of 3)

- Unique user identification number (user ID)
- Member of a primary group identified by a group ID
- Belongs to a specific group
- 12 protection bits
  - Specify read, write, and execute permission for the owner of the file, members of the group and all other users
- The owner ID, group ID, and protection bits are part of the file's inode



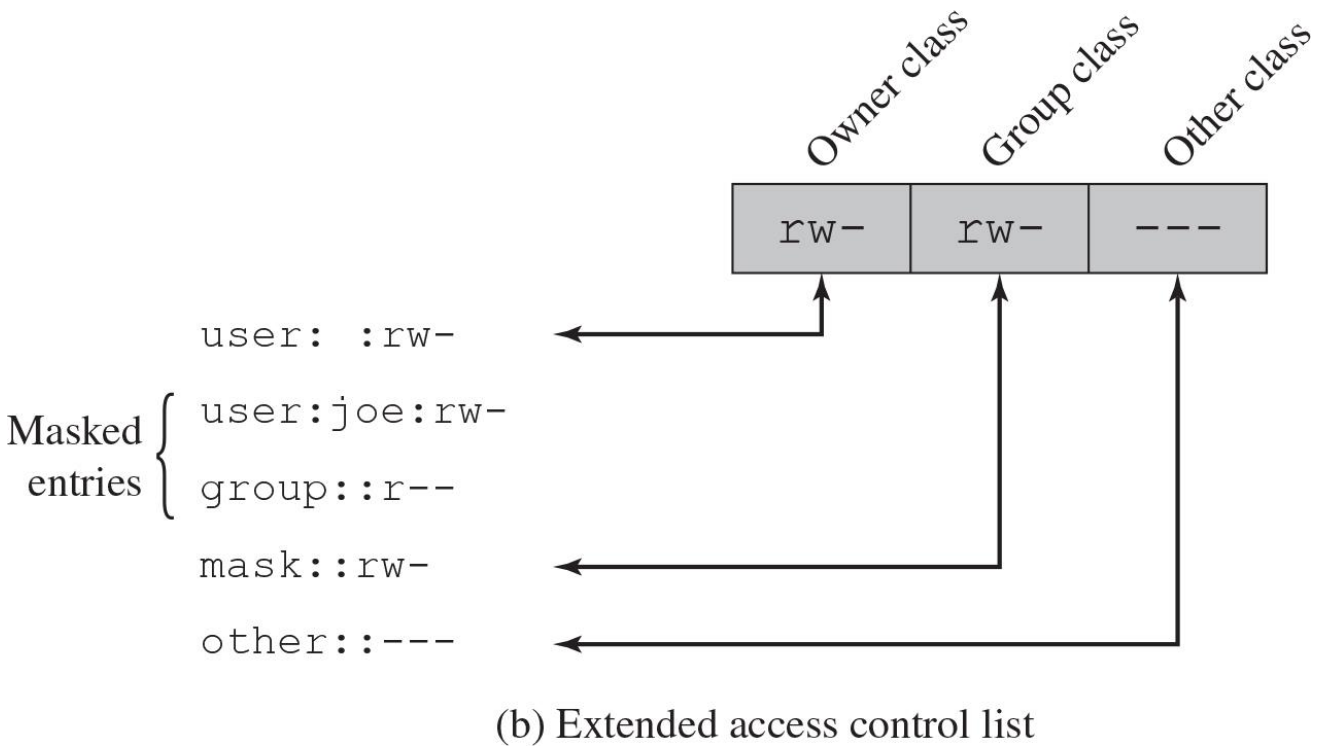
(a) Traditional UNIX approach (minimal access control list)

Figure 4.5 UNIX File Access Control

Traditional UNIX File Access Control

- "Set user ID"(SetUID)
- "Set group ID"(SetGID)
  - System temporarily uses rights of the file owner/group in addition to the real user's rights when making access control decisions
  - Enables privileged programs to access files/resources not generally accessible
- Sticky bit
  - When applied to a directory it specifies that only the owner of any file in the directory can rename, move, or delete that file
- Superuser
  - Is exempt from usual access control restrictions
  - Has system-wide access

Figure 4.5 UNIX File Access Control



(b) Extended access control list

Figure 4.6 Users, Roles, and Resources

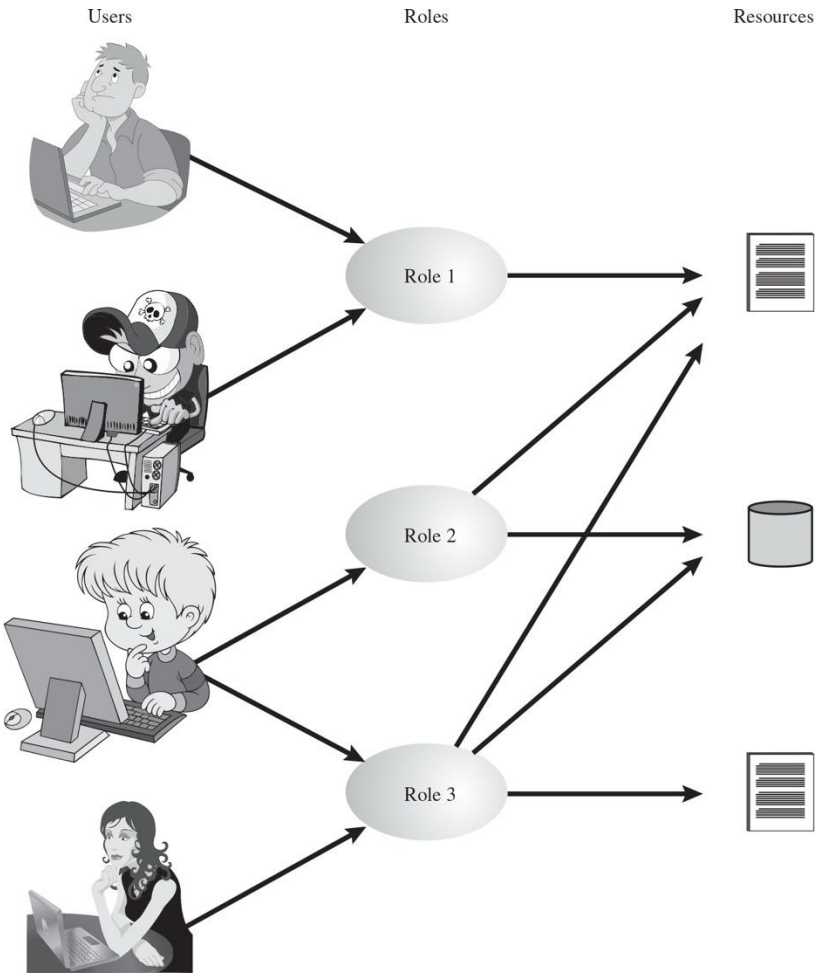
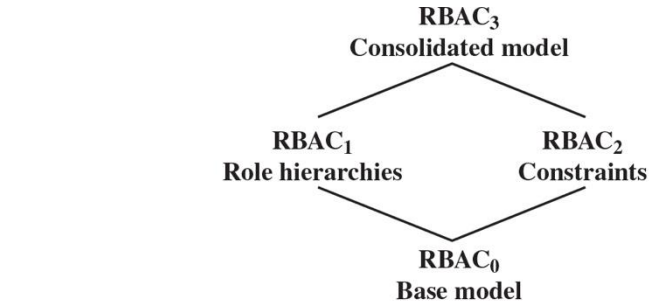


Figure 4.7 Access Control Matrix Representation of RBAC

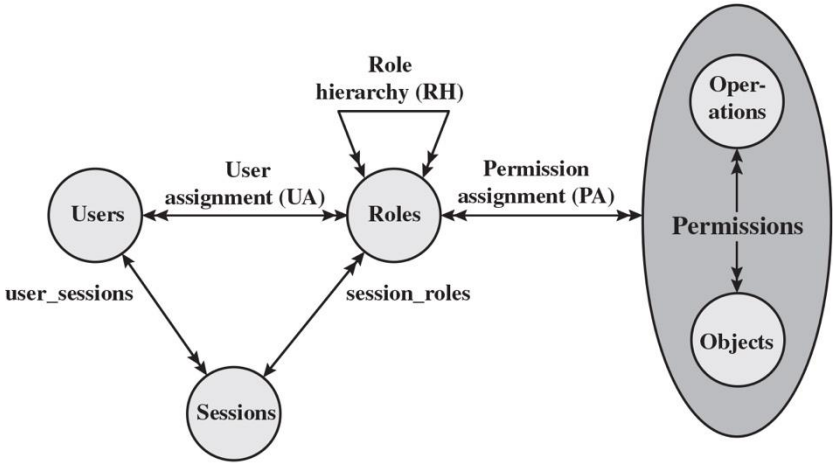
	R_1	R_2	• • •	R_n
U_1	✕			
U_2	✕			
U_3		✕		✕
U_4				✕
U_5				✕
U_6				✕
• • •				
U_m	✕			

		OBJECTS							
		R_1	R_2	R_n	F_1	F_2	P_1	P_2	D_1	D_2
ROLES	R_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
	R_2		control		write *	execute			owner	seek *
	• • •									
	R_n			control		write	stop			

Figure 4.8 A Family of Role-Based Access Control Models



(a) Relationship among RBAC models

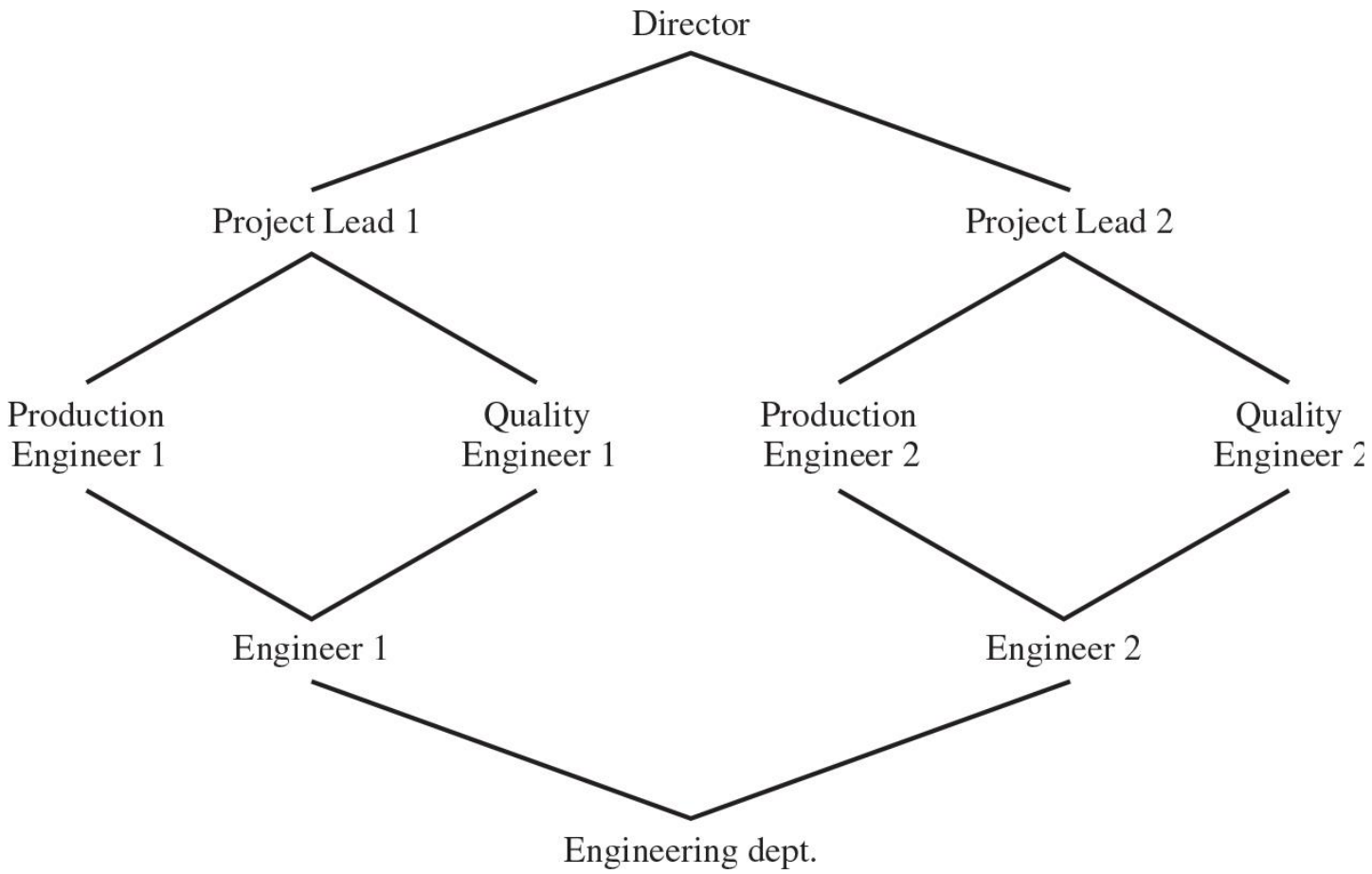


(b) RBAC models

Table 4.4 Scope RBAC Models

Models	Hierarchies	Constraints
$RBAC_0$	No	No
$RBAC_1$	Yes	No
$RBAC_2$	No	Yes
$RBAC_3$	Yes	Yes

Figure 4.9 Example of Role Hierarchy



Constraints - RBAC

- Provide a means of adapting RBAC to the specifics of administrative and security policies of an organization
- A defined relationship among roles or a condition related to roles
- Types:
  - Mutually exclusive roles
    - A user can only be assigned to one role in the set (either during a session or statically)
    - Any permission (access right) can be granted to only one role in the set
  - Cardinality
    - Setting a maximum number with respect to roles
  - Prerequisite roles
    - Dictates that a user can only be assigned to a particular role if it is already assigned to some other specified role

Summary

- Access control principles
  - Access control context
  - Access control policies
- Subjects, objects, and access rights
- Discretionary access control
  - Access control model
  - Protection domains
- UNIX file access control
  - Traditional UNIX file access control
  - Access control lists in UNIX
- Role-based access control
  - RBAC reference models

Copyright