MONASH University
Information Technology

FIT2093 INTRODUCTION TO CYBER SECURITY

MONASH University
Information Technology

FIT2093 INTRODUCTION TO CYBER SECURITY

Lecture 3:
Access Control and Security Models

Unit Structure

- Introduction to security of
- Authentication
- Access Control
- Fundamental concepts of cryptography
- Symmetric encryption techniques
- Introduction to number theory
- Public key cryptography
- Integrity management
- Practical aspects of cyber security
- Hacking and countermeasures
- Database security
- IT risk management & Ethics and privacy

Previous Lecture

- **introduced user authentication**
  - using passwords
  - using tokens
  - using biometrics
- **examined user authentication issues**
- **How password is managed in unix**
- **example application and case study**

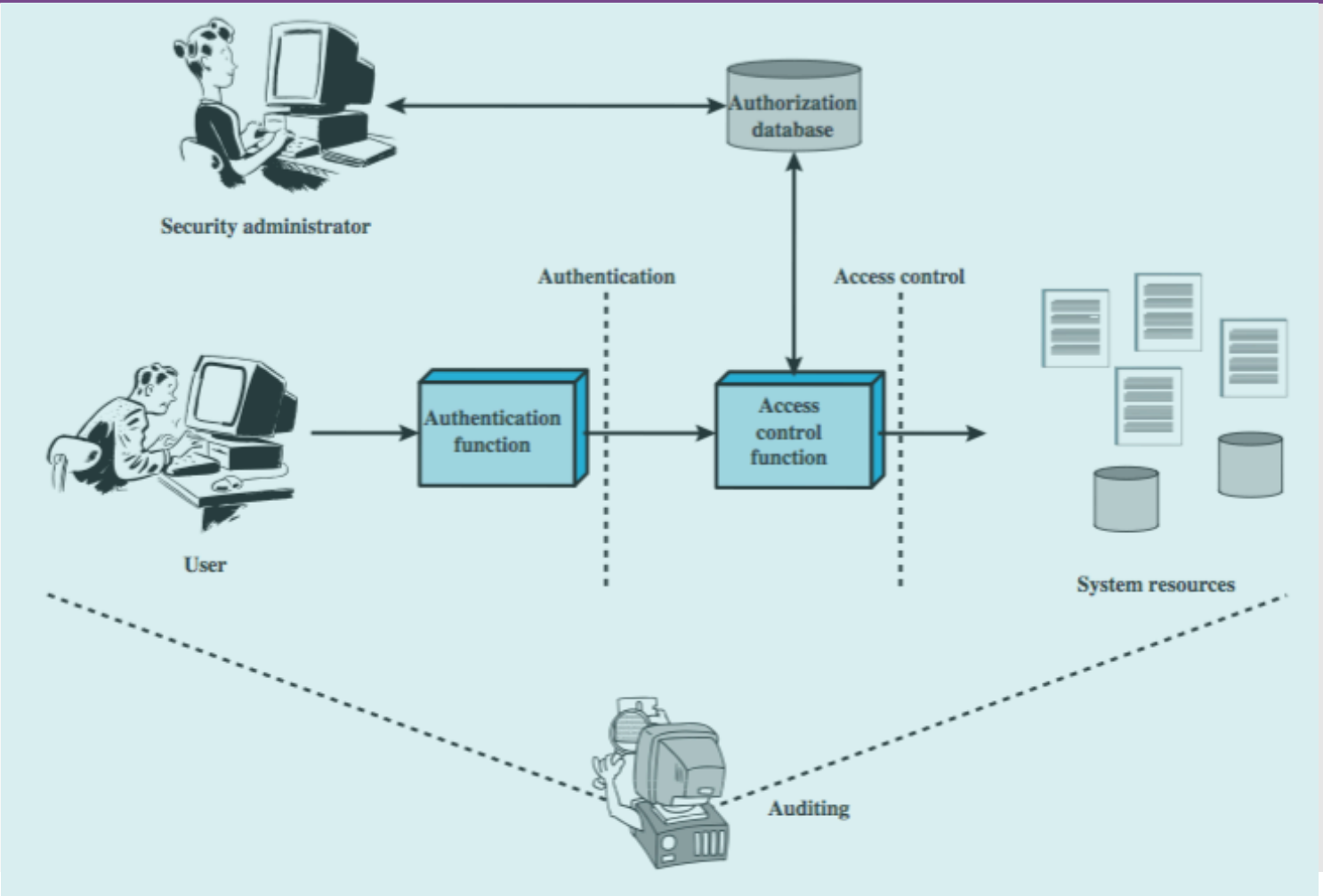
Objectives of this lecture are

- **Understand the role of access control mechanism in security.**
- **Appreciate the complexity of applying access control in a large organisation.**
- **Understand the role of the fundamental security models in relation to access control of information in organisations.**

Access Control

- **"The prevention of unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner"**
- **central element of computer security**
- **assume have users and groups**
  - authenticate to system
  - assigned access rights to certain resources on system

Access Control Principles



Access Control Elements

- **subject - entity that can access objects**
  - a process representing user/application
  - often have 3 classes: owner, group, world
- **object - access controlled resource**
  - e.g. files, directories, records, programs etc
  - number/type depend on environment
- **access right - way in which subject accesses an object**
  - e.g. read, write, execute, append, delete, create, search

What you mean by access control?

Objects – usually data objects

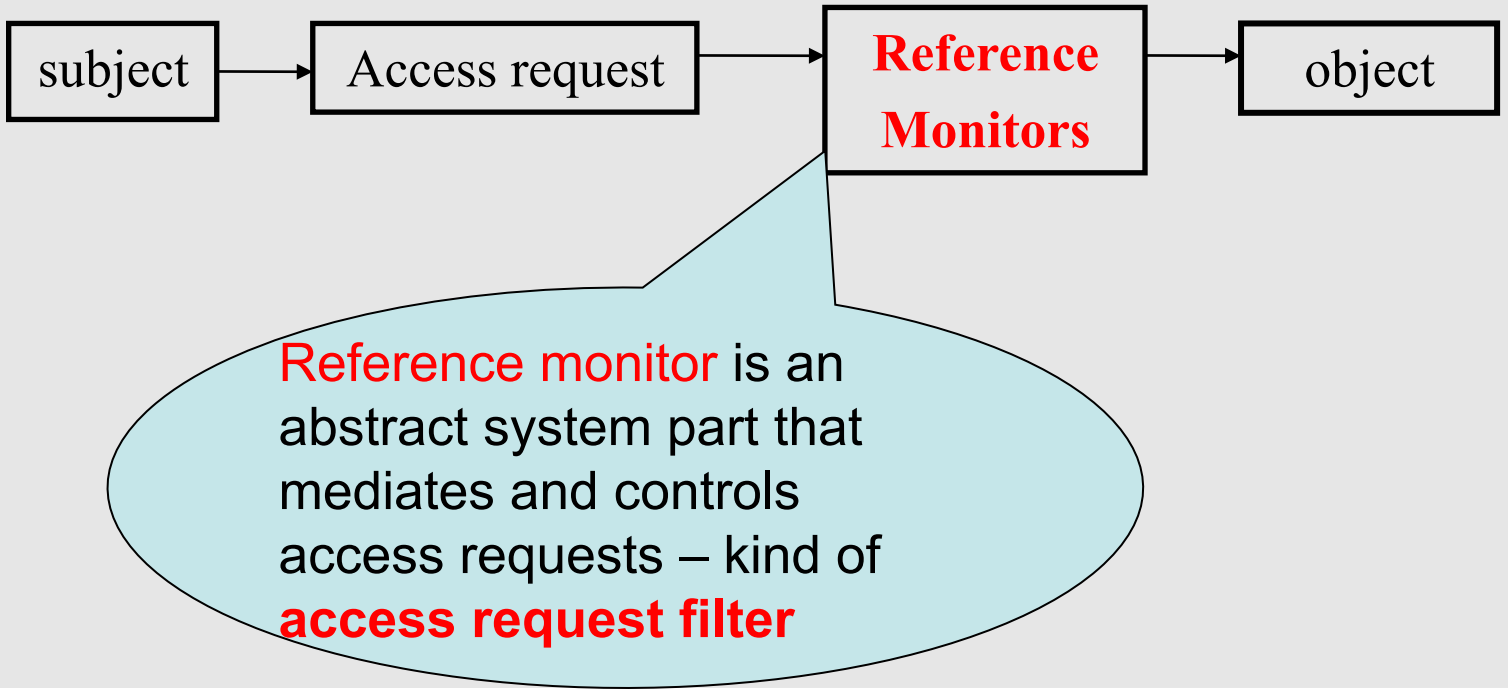
- **Rules that define who can access what?**
- **What you mean by the term ACCESS with respect to information resources?**
  - normally read, write, execute (Unix system uses these operations).
  - The file owner can control the permissions to these operations.
  - Windows has, in addition to these, permissions for *delete, change ownership and change permissions*
  - *Some systems define write that includes reading rights. These systems often have one more operation → **append** (blind write)*

Access Control

- **Another level of defense, after gaining access to the system.**
- **Any multi-users OS system needs to have a policy of controlling "who can access what" in the system.**
- **Who => Subject**
  - Active entity
  - Entity that performs 'action'
  - Example: user, program or process
- **What => object**
  - Passive entity
  - Entity where the action is performed by the subject.
  - Example: process, file, program, data, printer, memory, …

Access rights describes the way in which a subject may access an object

Access Control – General Model



Access Control

Access control

- **Limits users to only IT resources they need to perform their tasks**
- **Access control can be specified based on:**
  - Discretionary Access Control (**DAC**)
  - Mandatory access control (**MAC**)
  - Role Based Access Control (**RBAC**)

Controlled by who owns the object

Apply control based on what their job (role) is

Apply standard control to every subject (users)

Access Control: Types

- **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. This policy is termed *discretionary because an* entity might have access rights that permit the entity, by its own volition, to enable another entity to access some resource.
- **Mandatory access control (MAC)**
  - Controls access based on comparing security labels (sensitivity of resources) with security clearances (eligibility of entities to access certain resources).
- **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.

Discretionary access control (DAC)

All OS implement this method

- **Access to information is controlled by the owner of the object**
- **It can also provide for centralised or distributed security management**
  - Centralised security — an administrator provides and controls access
  - Distributed security — managers or team leaders control access

Mandatory Access Control (MAC)

- **Imposes universal security conditions for all users, IT systems and information**
- **Commonly used in military systems**
- **Information is classified based on attributes such as sensitivity, secrecy and confidentiality**
- **A subject is said to have a security clearance of a given level;**
- **an object is said to ha… classification of a gi…**

Also known as
Multilevel Model
(based on categorisation of objects)

Role-based Access Control (RBAC)

- **Access to information and its resources is based on a user's role.**
- **It can cater for hierarchies and business constraints**
  - Hierarchies define responsibilities and roles
  - Constraints provide the boundary of the job

Also known as Multilevel Model (based on categorisation of users)

Implementation of Access Control

Data objects (files), Processes → Reference Monitors as part of OS

- **Who monitors that the access control requirements are not violated?**
- **Access control can also be implemented in different layers of technology.**
  - Application, services/middleware, operating systems or hardware.
- **We will restrict ourself to the operating system level.**

Access Control-Security

- **Access control matrix (ACM)**
  - Defines the *subjects* (users), *objects* (information or resources) and *type of access*
  - A combination of these three defines an authorisation rule (also known as access rule)

Access Control Matrix

- **Represent the allowed access operation by a subject on an object as an element of a matrix**

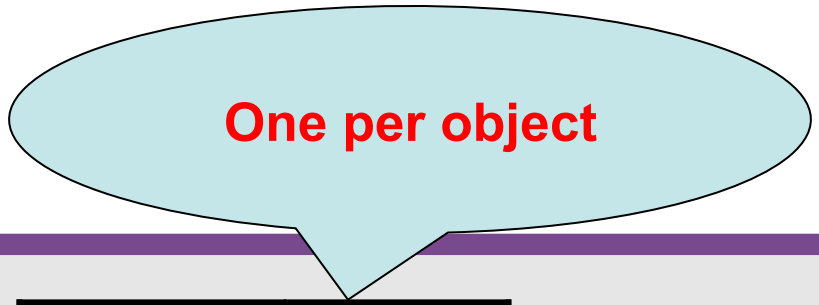
		Objects			
		File A	Program A	Directory X	File B
Subjects	Sam	rw	rwx	rwx	r
	Alice	r	x	rw	-
	Bob	w	r	-	r

Since we access objects, the possible operations are read (r), write (w) and execute (x)
[Unix specific] **deal with data**

Access Control Matrix

- **In real systems, however, access control matrices are not very practical to implement because:**
  - **the matrix is usually sparse and there is a lot of wastage and redundancy**
  - **Although new subjects and/or objects can be added or removed easily, yet the centralized matrix could be a bottleneck for access enforcement**
- **Possible solutions:**
  - Access Control List
  - Capabilities List

Access Control List



- **Focus on the object.**
- **For each object specifies the subjects and their access operations.**
  - ACL is a column of the access control matrix
- **The list is usually kept with the object.**
- **Simple to implement but difficult to manage.**
  - e.g. Removing a user.
- **Each list can be of variable length**

Subject	File A
Sam	rw
Alice	r
Bob	W
…	

Capabilities List

One per Subject

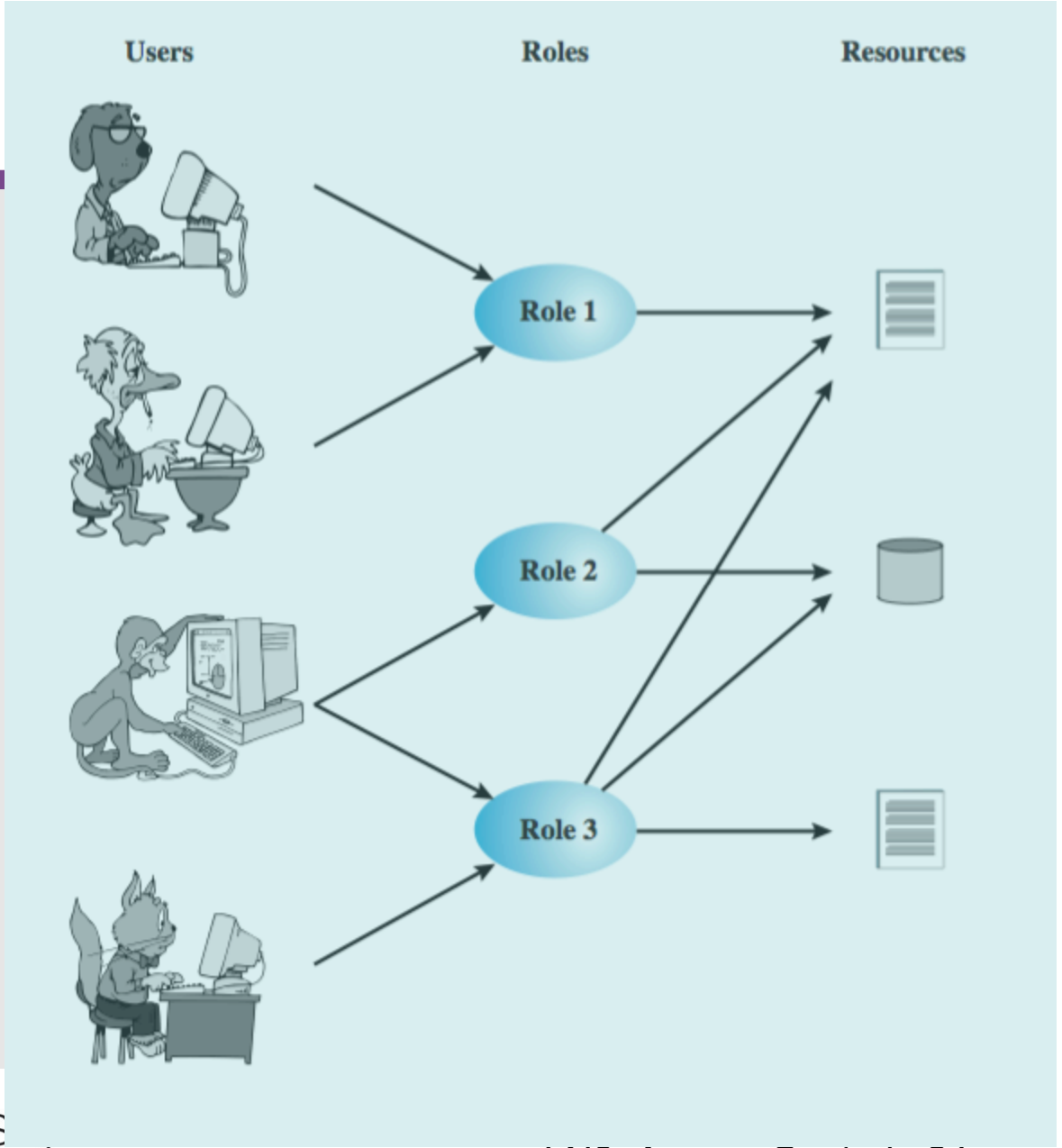
- **Focus on the subject.**
- **For each subject specifies the objects and what kind of access operation that can be applied to the objects.**
  - The rows of the access control matrix.
- **Weakness:**
  - It is difficult to determine the list of subjects with certain access right to an object.
  - Security risk if the list is forged
- **Each list can be of variable length.**

Object	Sam
File A	rw
Program A	rwx
Directory X	rwx
Program B	r

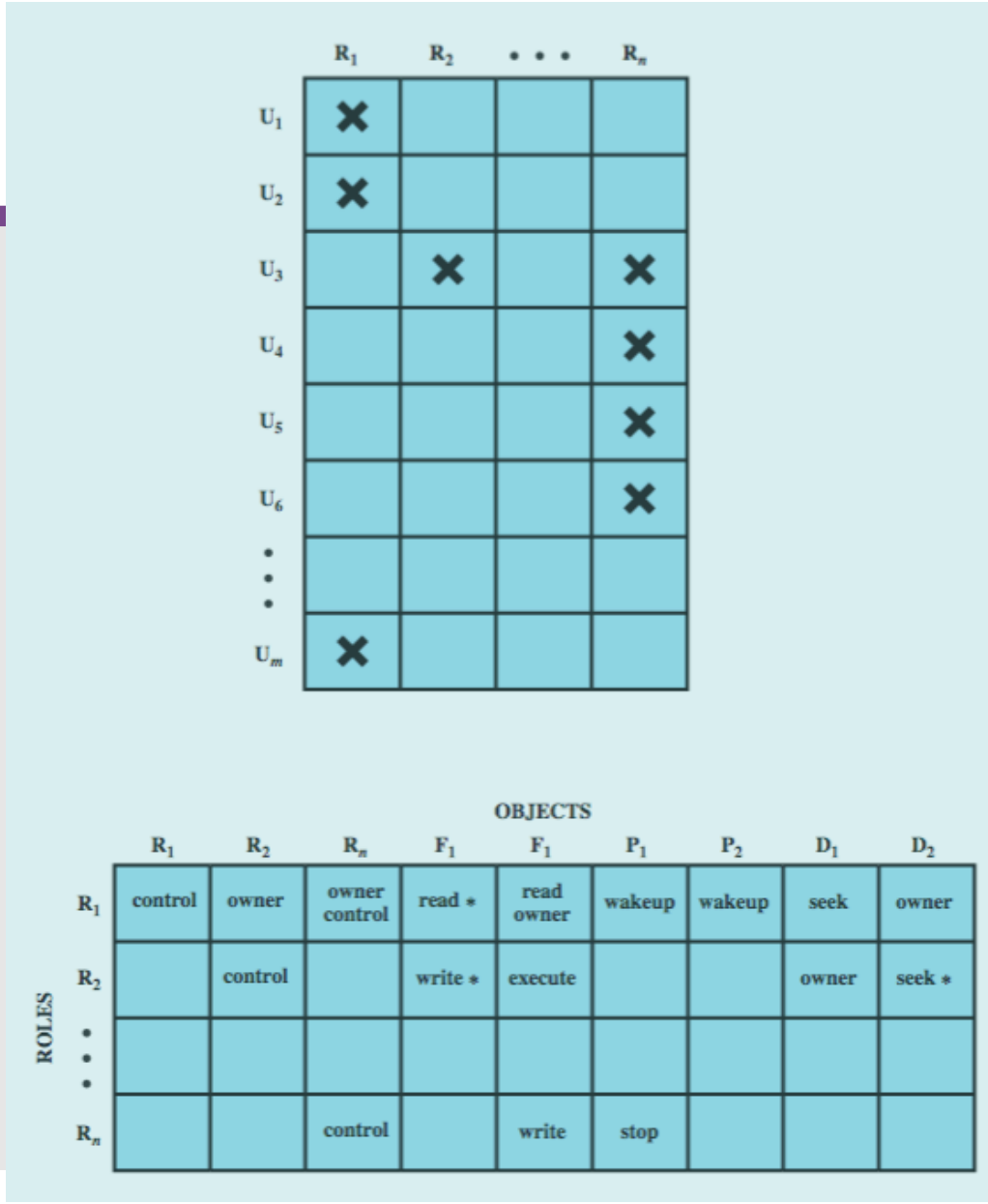
To simplify the above control structures is to

- **Groups subjects into a role and assign access to a set of objects based on the role.**
- **A group is a collection of subjects with similar (or identical) access right.**
- **A role is a collection of access permissions that a user or a group of users have on a set of objects**
  - defines the notion of context in which the access control is applied.

Role-Based Access Control



Role-Based Access Control(2)



	R_1	R_2	...	R_n
U_1	✖			
U_2	✖			
U_3		✖		✖
U_4				✖
U_5				✖
U_6				✖
...				
U_m	✖			

	OBJECTS								
ROLES	R_1	R_2	R_n	F_1	F_1	P_1	P_2	D_1	D_2
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			

Recap

- **Access Control (access right) describes the way a subject may access an object.**
- **Access Control Matrix represents the allowed access operation by a subject on an object as an element of a matrix**
- **An Access Control List lists users (subject) and their permitted access rights to the objects.**
- **Capability List specifies the objects and what kind of access operation that can be applied to the objects by the subject**

Recap:Who decides the access rights to an object?

- **DAC (Discretionary Access Control)**
  - The owner of an object decides on the access rights to the object.
- **MAC (Mandatory Access Control)**
  - The administrator decides on the access rights to an object according to a certain policy.
- **Most systems are based on DAC.**

Recap:Who decides the access rights to an object?

- **DAC (Discretionary Access Control)**
  - The owner of an object decides on the access rights to the object.
  - This policy is termed as discretionary because the subject might have access rights to the object, but by his own volition, can enable other subjects access this object.

Recap:Who decides the access rights to an object?

- **MAC (Mandatory Access Control)**
  - The administrator decides on the access rights to an object according to a certain policy.
  - Access based on comparing security labels (which indicate the sensitivity of the objects) with security clearance of subjects.
  - This policy is termed mandatory because a subject that has a clearance to access an object may not, by his own volition, enable another subject to access that object.

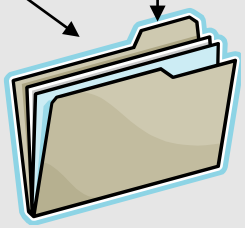
Problem with DAC model

Tom

Word Processor

Alice

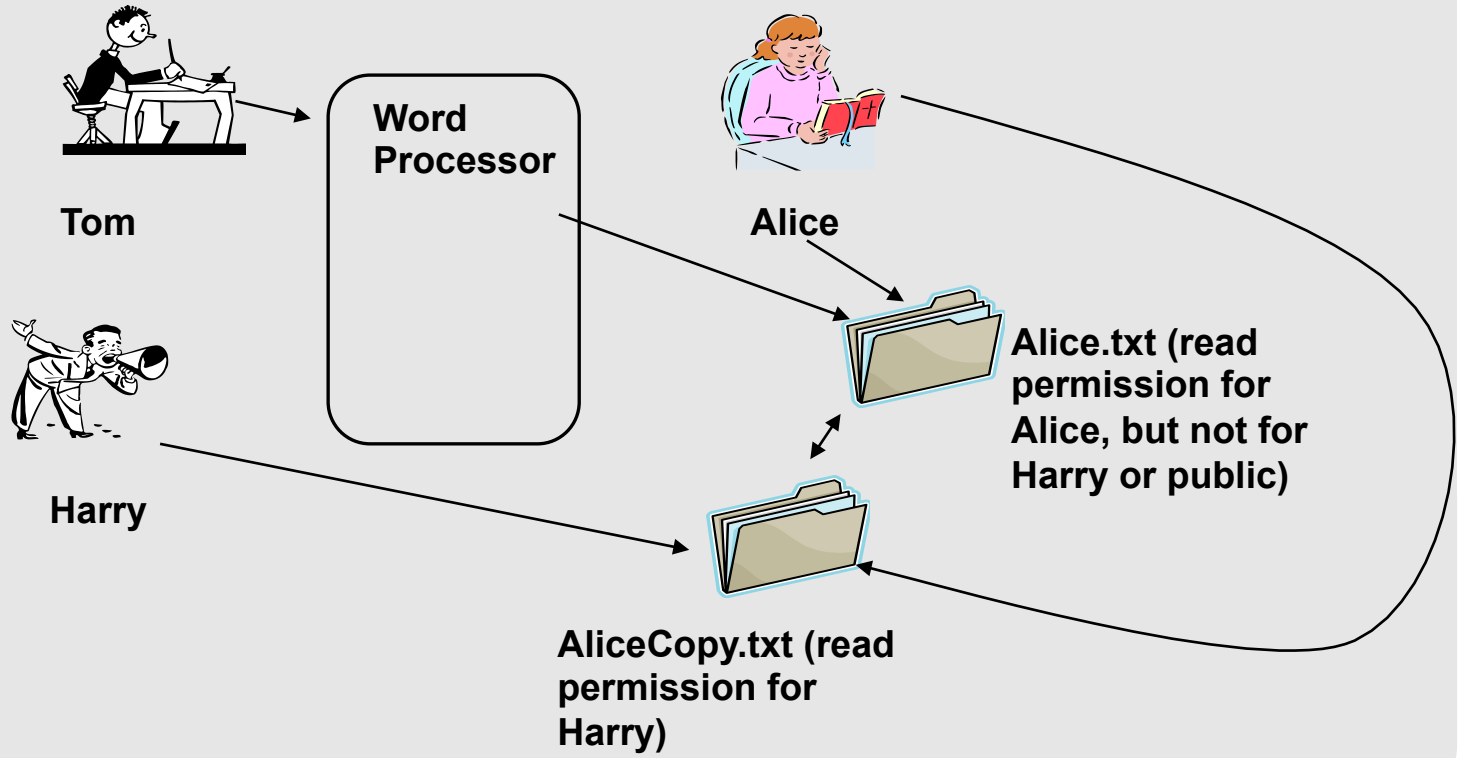
Harry (wants to read Alice.txt)



Alice.txt (read permission for Alice, but not for Harry or public)

Problem with DAC (1)

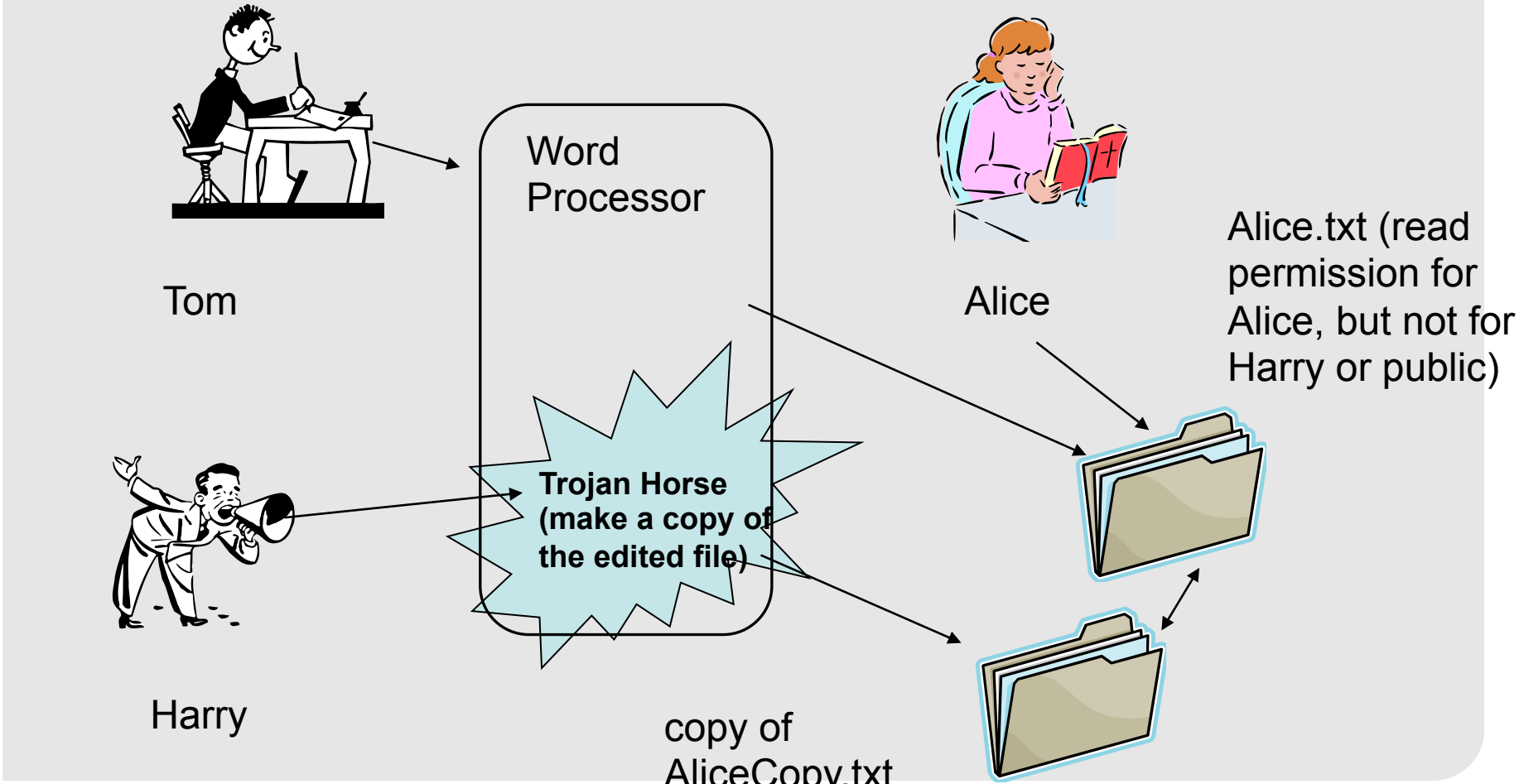
- **Alice can make a copy of Alice.txt and give read permission to AliceCopy.txt for Harry**



Trojan Horses

- **Trojan horses are useful or apparently useful programs or commands**
  - but contain malicious hidden code
  - do something undocumented which the programmer intended, but that the user would not approve of
  - act as a delivery vehicle
  - which are usually superficially attractive, eg game, s/w upgrade, etc.
- **Can cause disastrous consequences**
  - sending data or password to attackers
- **Most anti-virus programs can't detect new Trojans**
  - but once their circulations are reported they can be detected and removed
- **Strategy:**
  - important to know and trust the source of any program before running it

Problem with DAC (2)



Problem with DAC (3)

- **Without the knowledge of the owner, the other users who have access to the object may violate the original intended access restrictions by the owner.**
- **In the above example, Alice may allow Harry to read the document without the knowledge of Tom!**
- **Another example: Unix directory and files access control**
  - Q? A directory is also an object to which access should be controlled. Assignment of wrong access control can lead to confidentiality/Integrity problem (refer to Unix examples later).

Solutions

- **Combine DAC with MAC.**
- **MAC based security model:**
  - Bell-La-Padula (BLP)
- **Before we look at the model, let us look at the 'typical organisations information access policies'**

In Organisations

- **Access is controlled at different levels of security to groups of people or objects**
  - Generally known as security levels
  - Typical security levels:
    - groups: CEO, Managers, Heads, Employee
    - objects: secret, confidential, current, open, unclassified.
- **Access rules under this (multilevel) classification is expressed as the organisation's (access) security policy.**

Security Policy

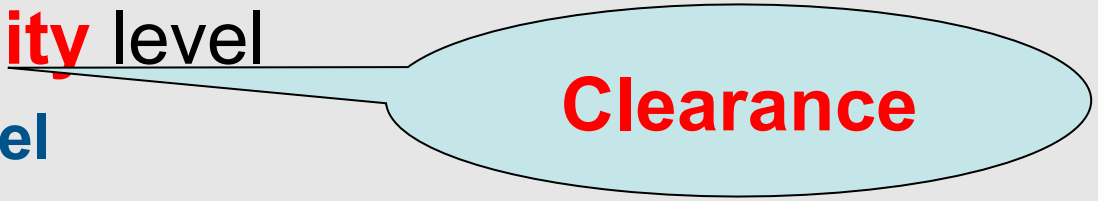
- Here *security policy* refers to a computer system's internal security policy, not organisational-level security
- These are access control policy models
- Idea of security models is to leave irrelevant details out and concentrate on some interesting property (usually confidentiality and integrity of information)

Recollect

- **MAC (Mandatory Access Control)**
  - Imposes universal security conditions for all users, IT systems and information Access is controlled at different levels of security to groups of people or objects
- **Hence security policy models can be thought of a realisation of MAC**

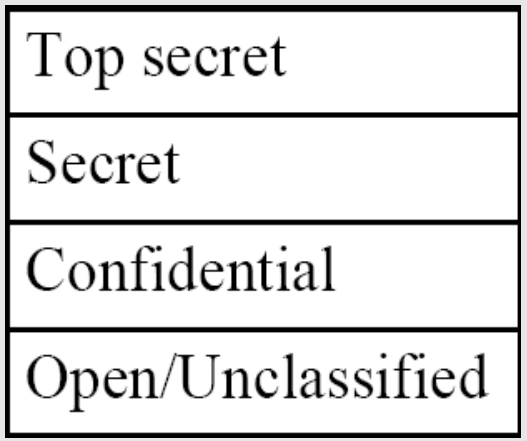
Access Control-Security Models

- **Multilevel model**
  - Categorises information by **sensitivity** and user access is based on their **responsibility** level
- **Multilateral model**
  - Information is categorised into classes based on **usage** patterns

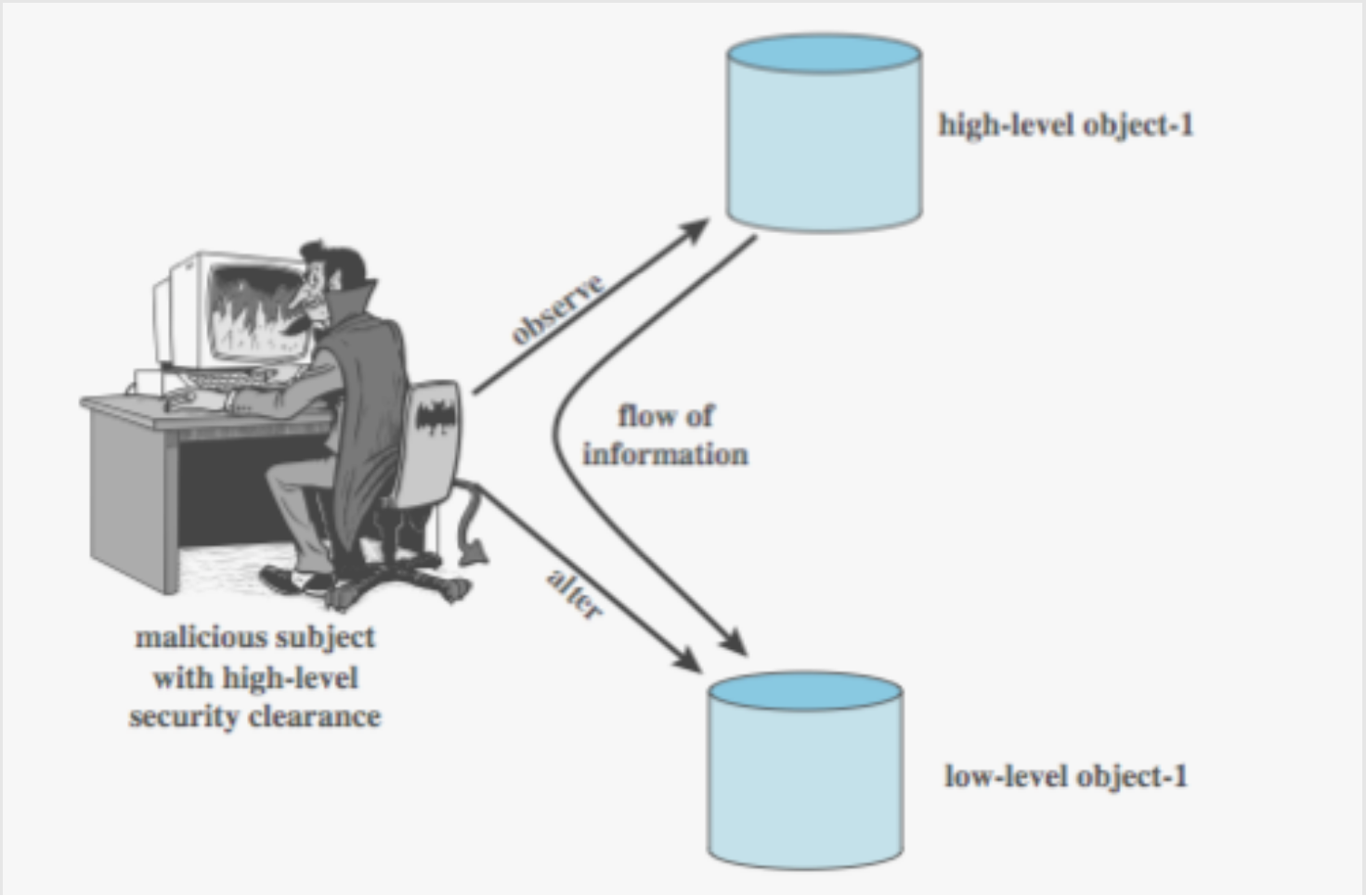


Multilevel Security (MLS)

- **Security levels**
  - A hierarchy of sensitivity attributes (ordering of levels)
  - Typical military-style hierarchy
- **An object's sensitivity attribute is called *classification***
- **Subjects have *clearances* to access objects in the hierarchy**
- ***Dominates*-relation: we say that x dominates y iff level(x) ≥ level (y)**
- ***Allows*: subjects x access to objects y accesstype → binary value**



Multi-Level Security

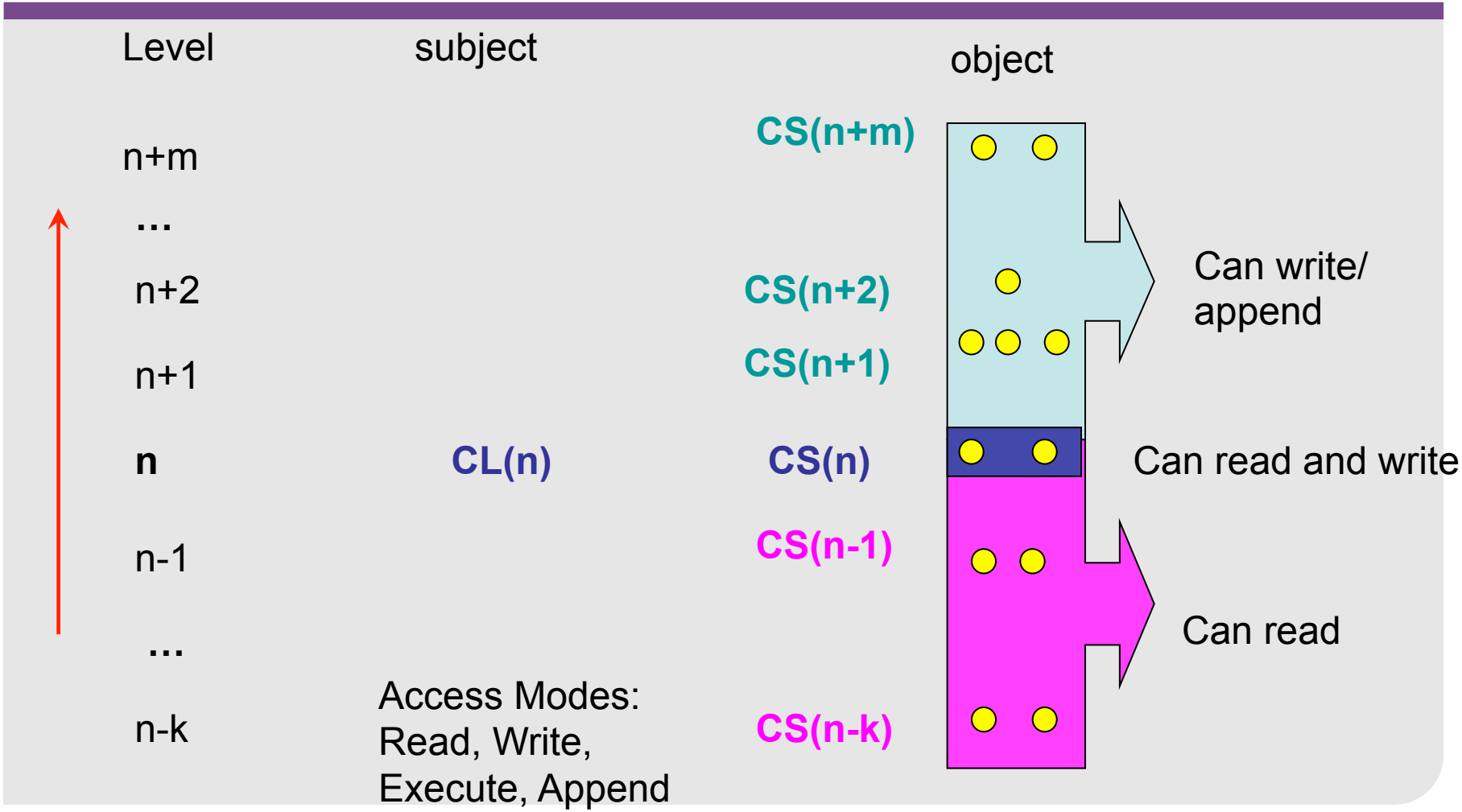


Bell-Lapadula Model (BLP)

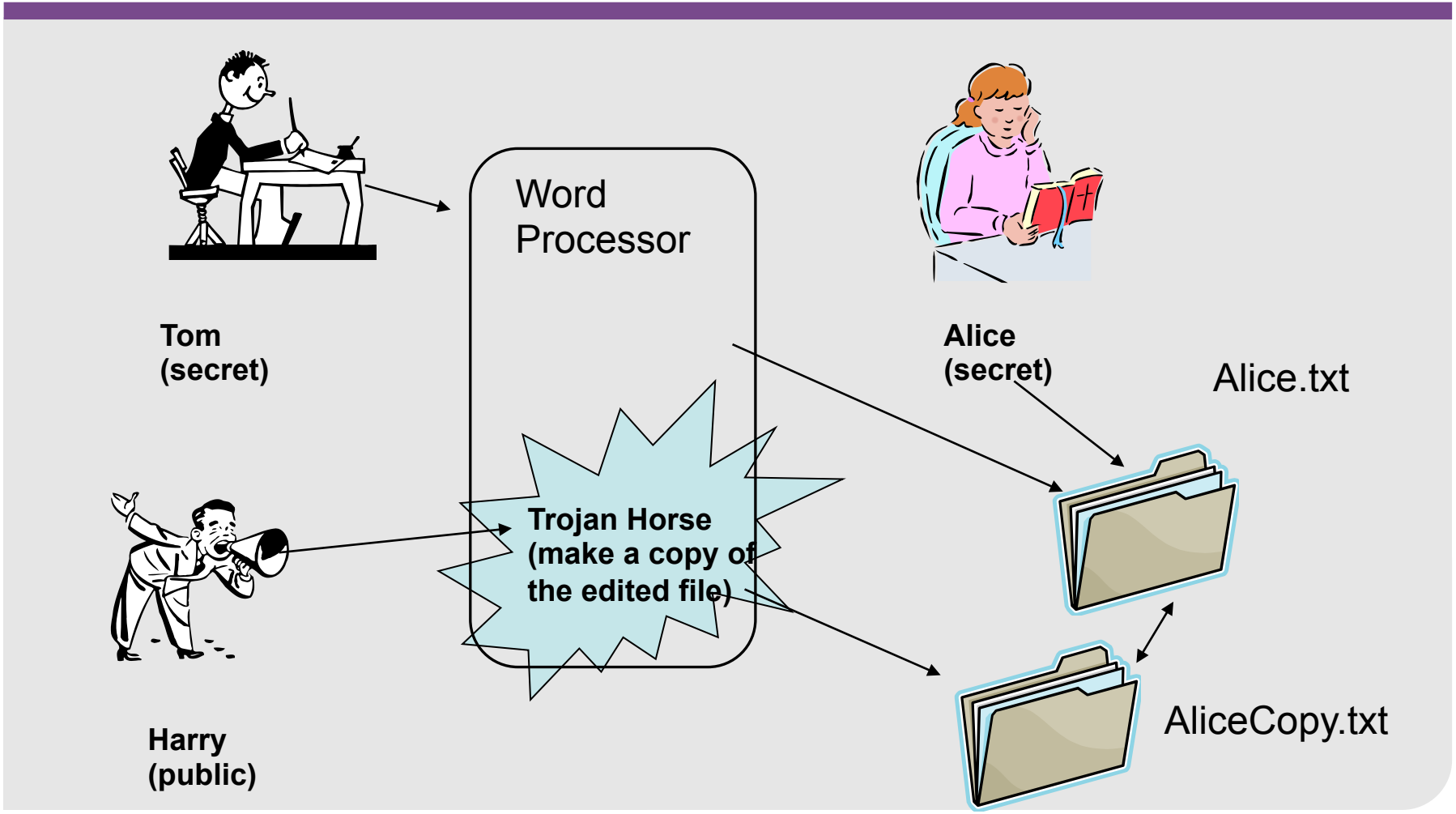
- **David Bell and Leonard LaPadula 1973**
- **BLP aims to protect the *confidentiality* property.**
- **Multi-level security model.**
- **Each of subject and object is assigned to a particular security level in the hierarchy.**
- **A subject has a clearance level.**
- **An object has a classification level.**
- **Three main conditions in the model:**
  - No READ UP **- NRU** (**simple security property**)
    - > A subject can only read object that is classified lower than or equal to its clearance level.
  - No WRITE DOWN- **NWD** (***property => star property**)
    - > A subject can only write to an object that is classified higher than or equal to its clearance level.
  - DISCRETIONARY access control – ds property
    - > a subject can exercise only accesses for which it has the necessary authorization and which satisfy the MAC rules.

BLP - Example

CL=clearance,
CS=classification

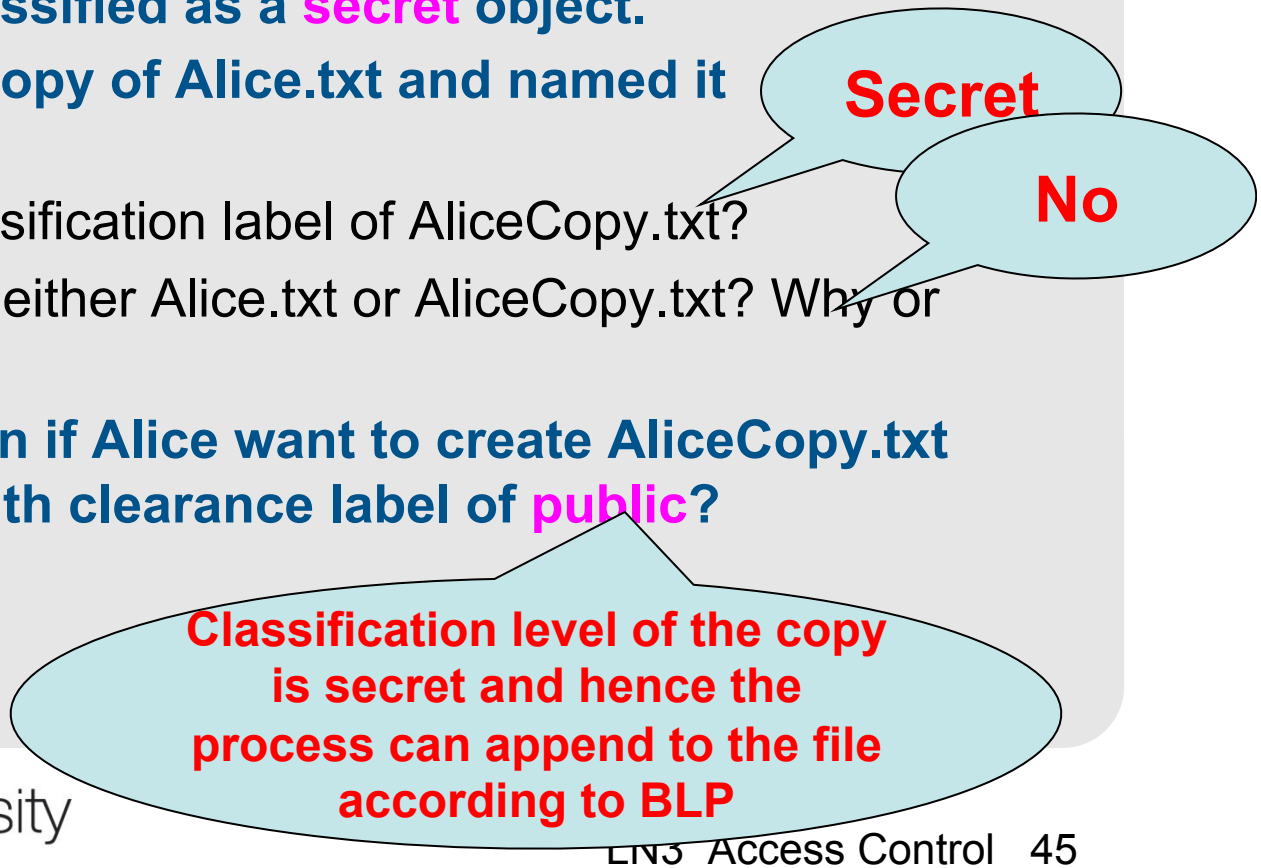


BLP solution for DAC problem?



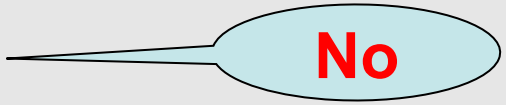
BLP solution (1)

- **Tom creates Alice.txt using a word processor that is executed at the clearance level of secret.**
- **Alice.txt will be classified as a secret object.**
- **Alice can make a copy of Alice.txt and named it AliceCopy.txt.**
  - What is the classification label of AliceCopy.txt?
  - Can Harry read either Alice.txt or AliceCopy.txt? Why or Why not?
- **What would happen if Alice want to create AliceCopy.txt using a process with clearance label of public?**



BLP solution (2)

- **Tom creates Alice.txt using the word processor that Harry has injected with Trojan Horse software.**
- **Assume Tom executes the word processor using clearance level of secret.**
  - What would be the classification labels of Alice.txt and AliceCopy.txt?
  - Can Harry read either Alice.txt or AliceCopy.txt?
- **Assume Tom executes the word processor using clearance level of public.**
  - Alice.txt can still be classified using "secret" label based on the *property.
  - The Trojan horse will have clearance level of public, hence cannot read Alice.txt based on the simple security property.

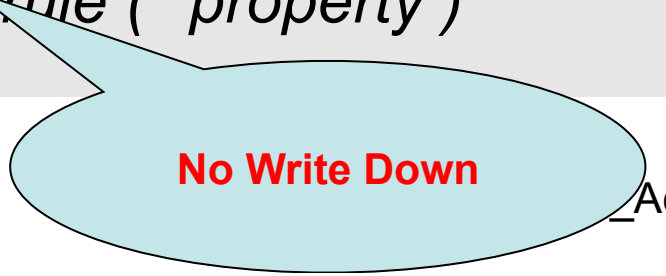


BLP – Problems?

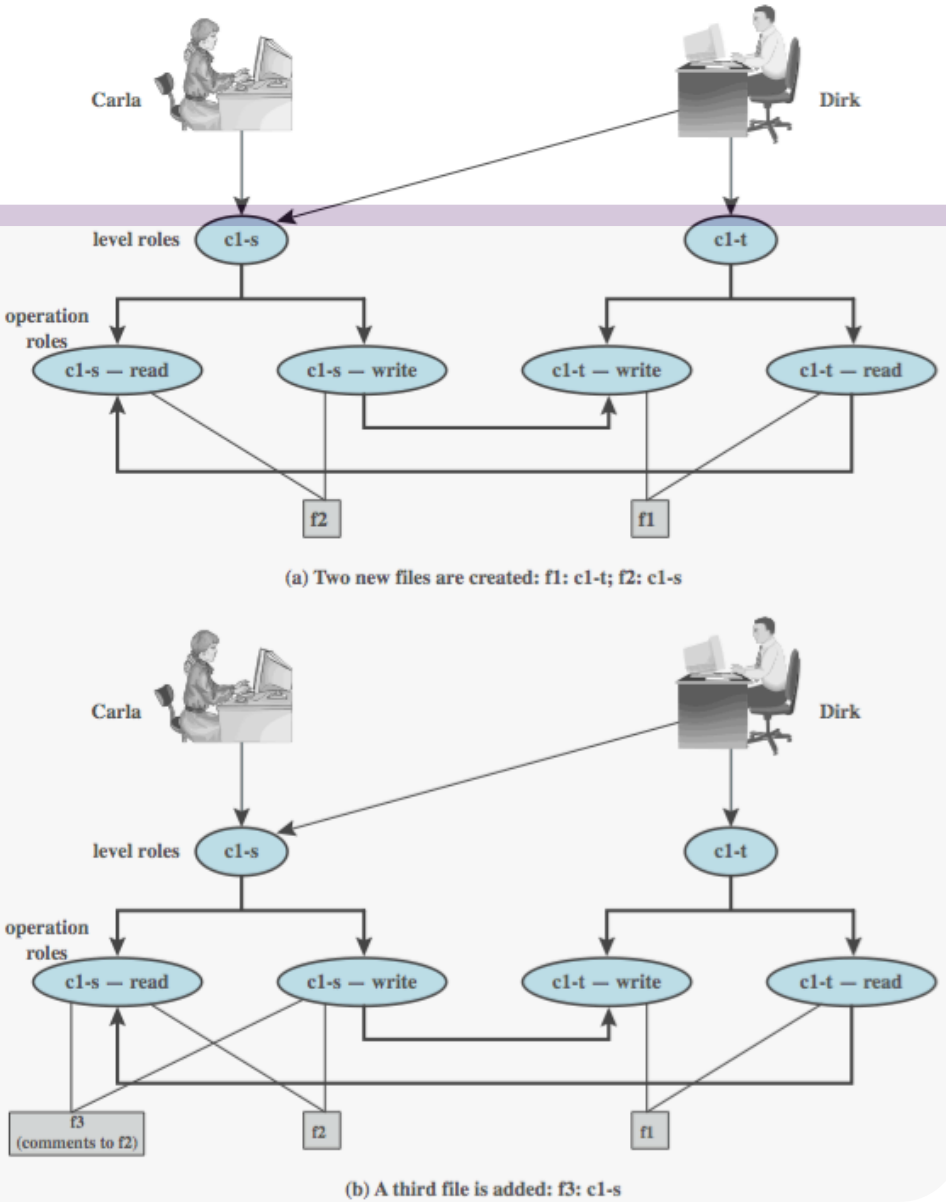
- **BLP is only concerned about confidentiality (but this is a design decision)**
- **Data integrity**
  - Blind write-ups are a threat to integrity (which is why many practical implementations allow writing only to objects at same level)
  - Classification creep: new document consolidates information from a range of sources and levels, some of that information is now classified at a higher level than it was originally.

BLP – Problems (contd)?

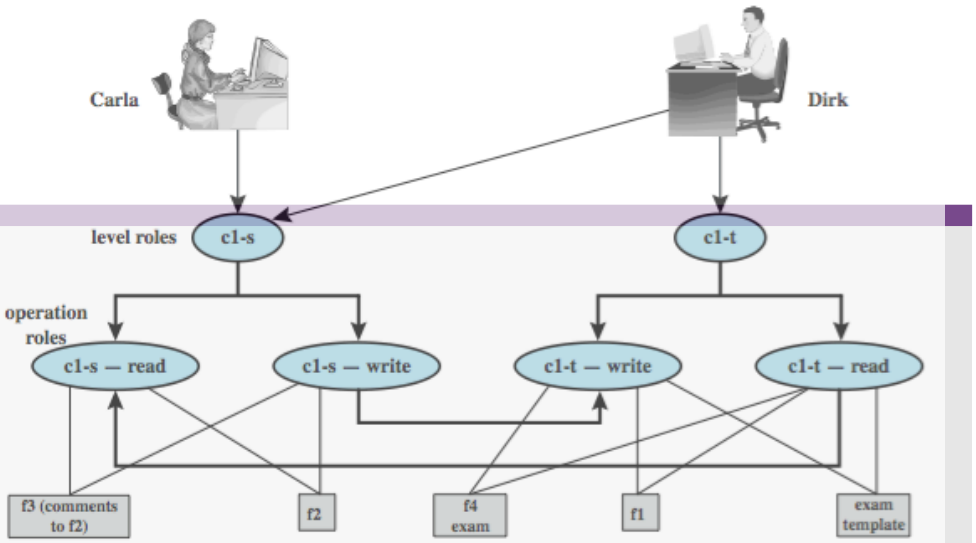
- **Read and write are not atomic operations.**
  - What would happen if a subject can lower its clearance level in between read and write operations?
  - Tranquility property: security levels and access rights never change
- **Privileged Processes**
  - How would a program such as encryption that read highly classified object and turn it to a public object (public cannot read it as it is encrypted) be able to perform its function?
  - Solution: Trusted Object – *subjects that are allowed to break the NWD-rule (* property)*



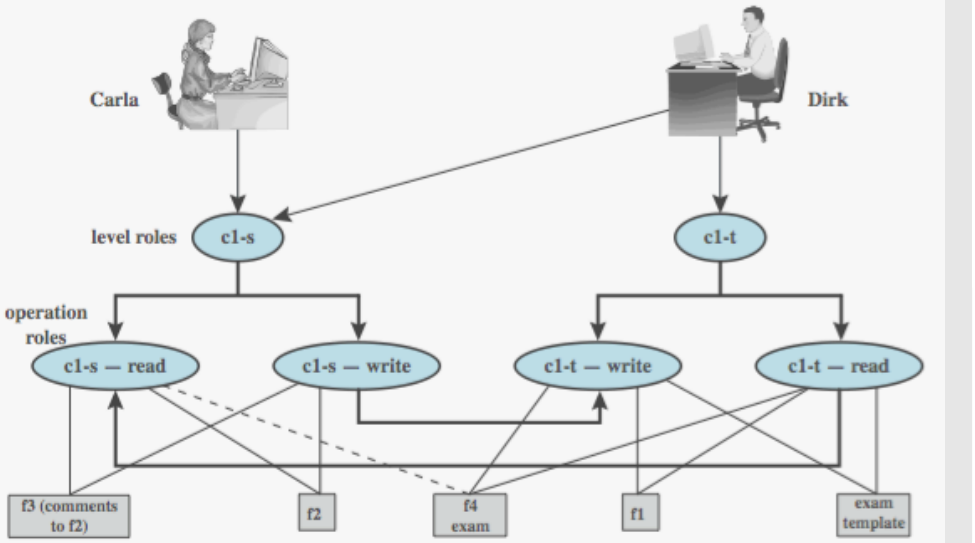
BLP Example


Carla
Dirk
level roles
c1-s
c1-t
operation roles
c1-s — read
c1-s — write
c1-t — write
c1-t — read
f2
f1
(a) Two new files are created: f1: c1-t; f2: c1-s
Carla
Dirk
level roles
c1-s
c1-t
operation roles
c1-s — read
c1-s — write
c1-t — write
c1-t — read
f3
(comments to f2)
f2
f1
(b) A third file is added: f3: c1-s

BLP Example cont.

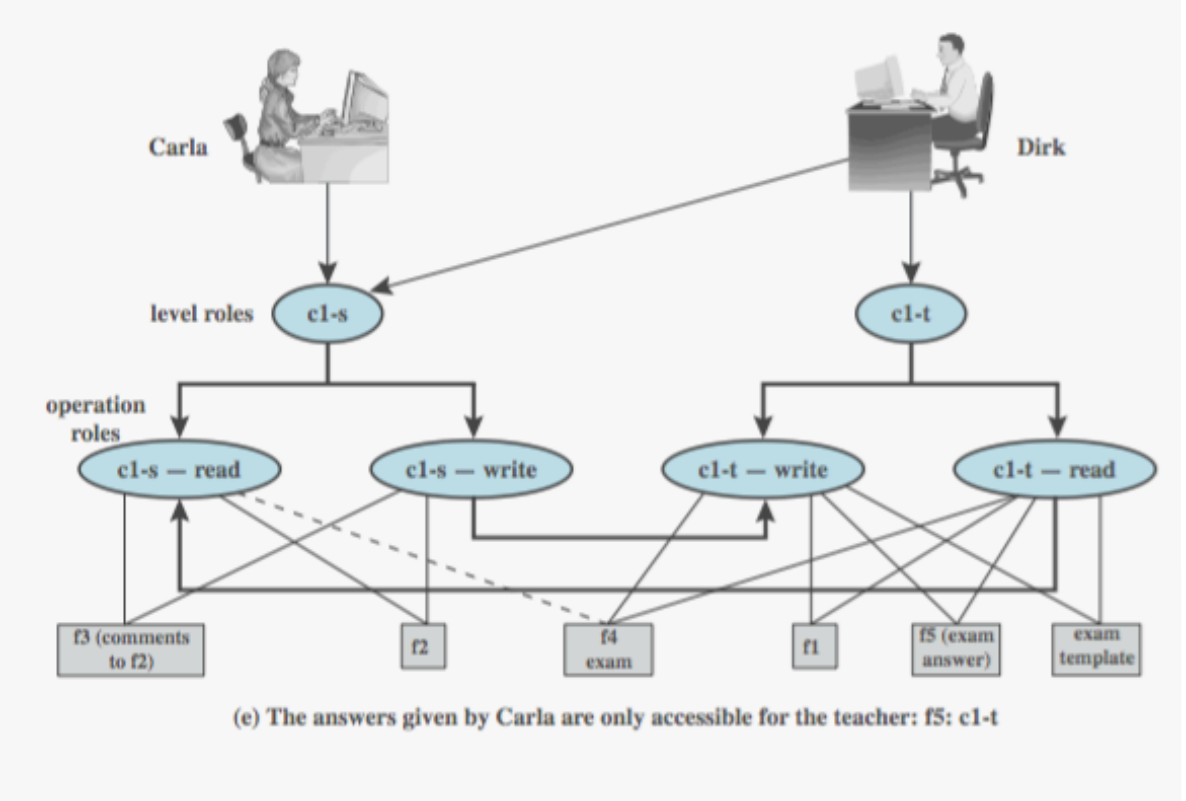


(c) An exam is created based on an existing template: f4: c1-t



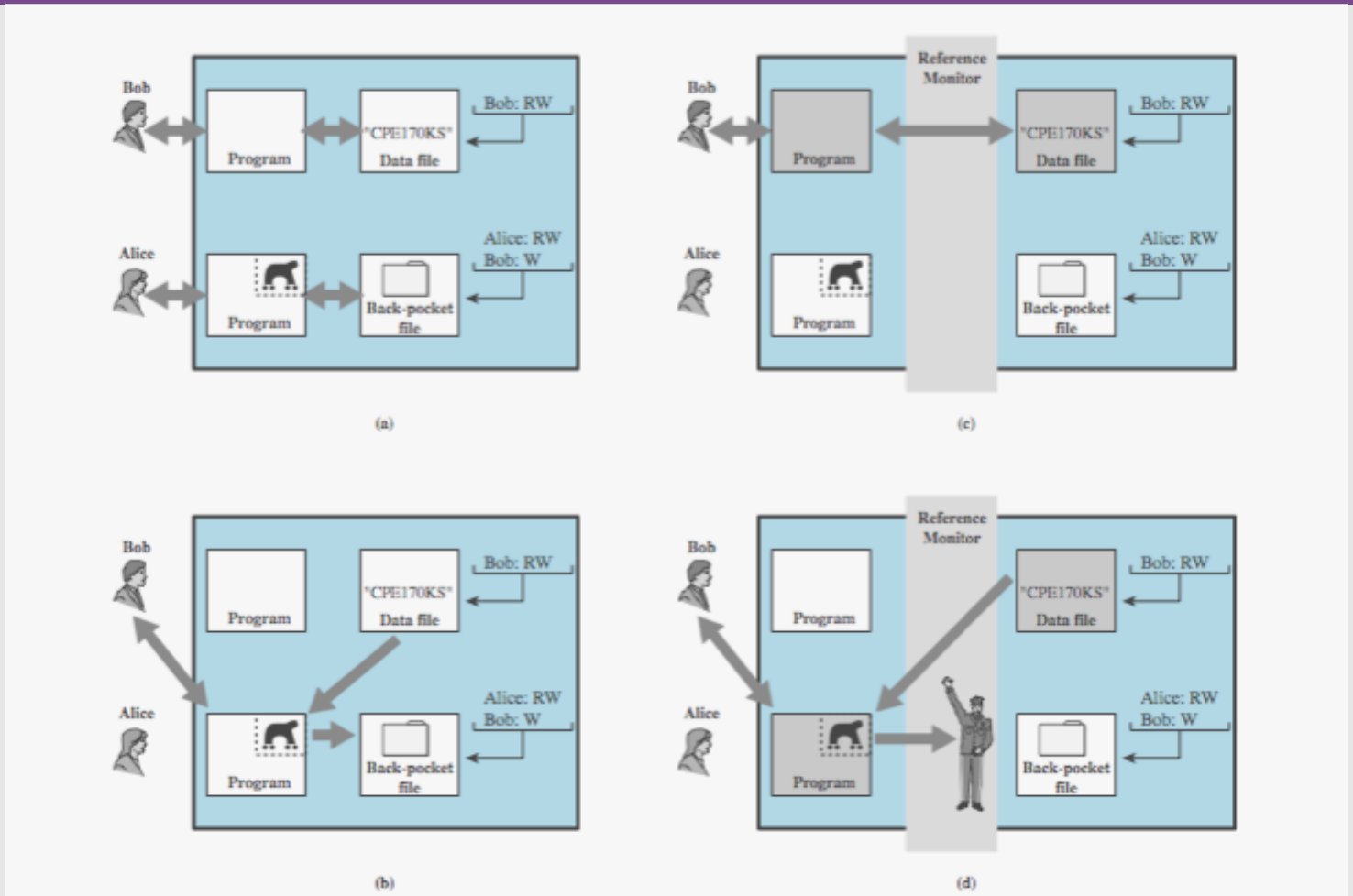
(d) Carla, as student, is permitted acess to the exam: f4: c1-s

BLP Example cont.



(e) The answers given by Carla are only accessible for the teacher: f5: c1-t

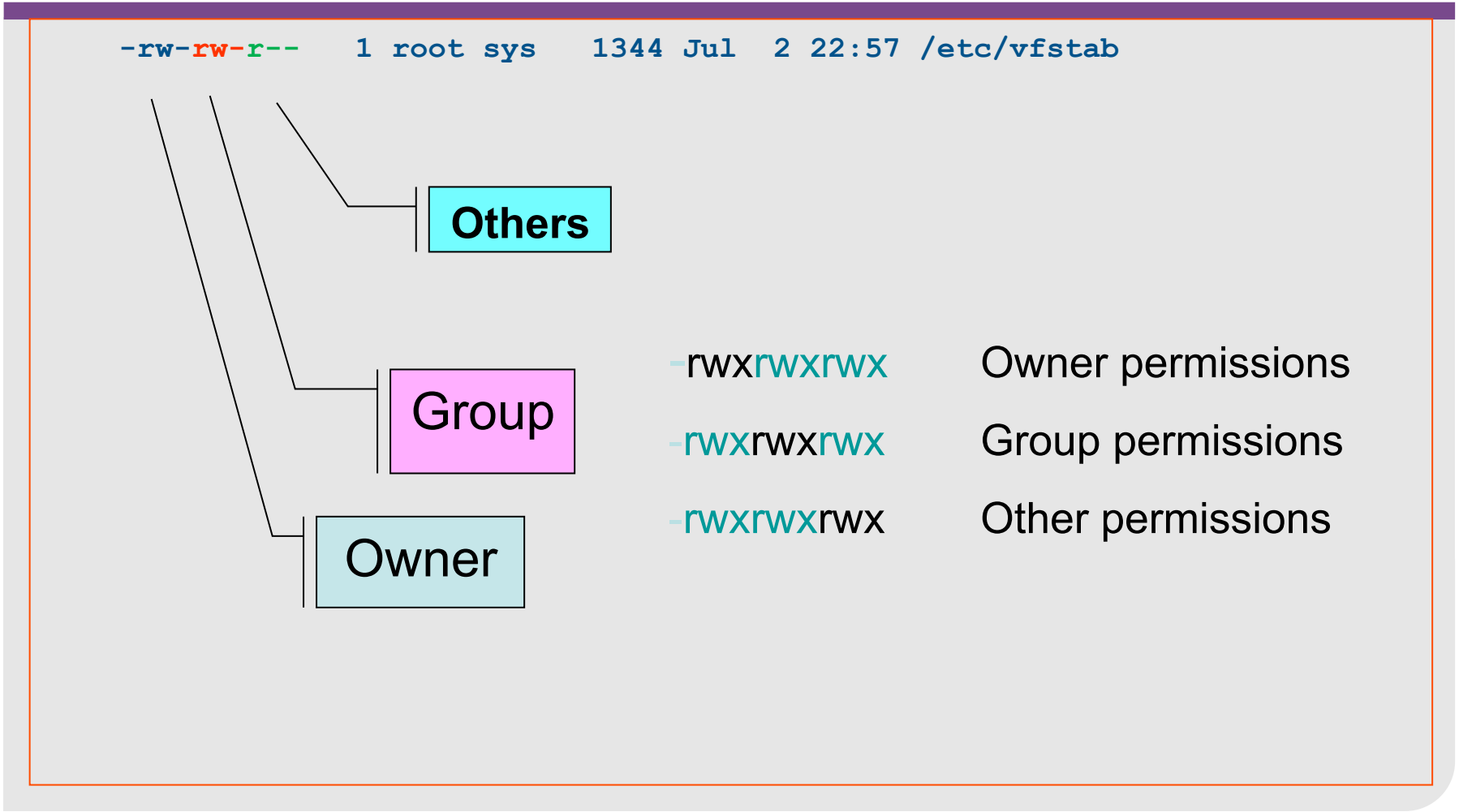
Trojan Horse Defence



MONASH University
Information Technology

UNIX Access Control

Basic file security



Basic file security

- **Important system files must have appropriate file permissions**
- **e.g:**

```
-r--r--r--   1 root other /etc/passwd
-r--------   1 root sys   /etc/shadow
-rw-r--r--   1 root sys   /etc/profile
drwxr-xr-x  18 root sys   /usr
```

File attributes

$ ls -ls exercise
2 -rw-r--r-- 1 srini users 1512 Jul 14 13:32 exercise

- Occupies 2 blocks of physical storage
- The permission mode of the file is read and write for the owner, read only for the group and read only for others
- There is only 1 hard link
- The user-id of the file's owner is *srini*
- The group id of the file is *users*
- The size of the file is 1512 bytes
- The file was last modified on July 14 at 13:32 hours
- The file name is exercise

The option l in the command is to request the output in long format
The option s is to get the size of the file in blocks

Root/Superuser role

- **Root or superuser is privileged user in UNIX.**
- **Root can access all components (files, devices, process, program,…) of the UNIX system.**
- **Protection of root password is paramount in UNIX security.**

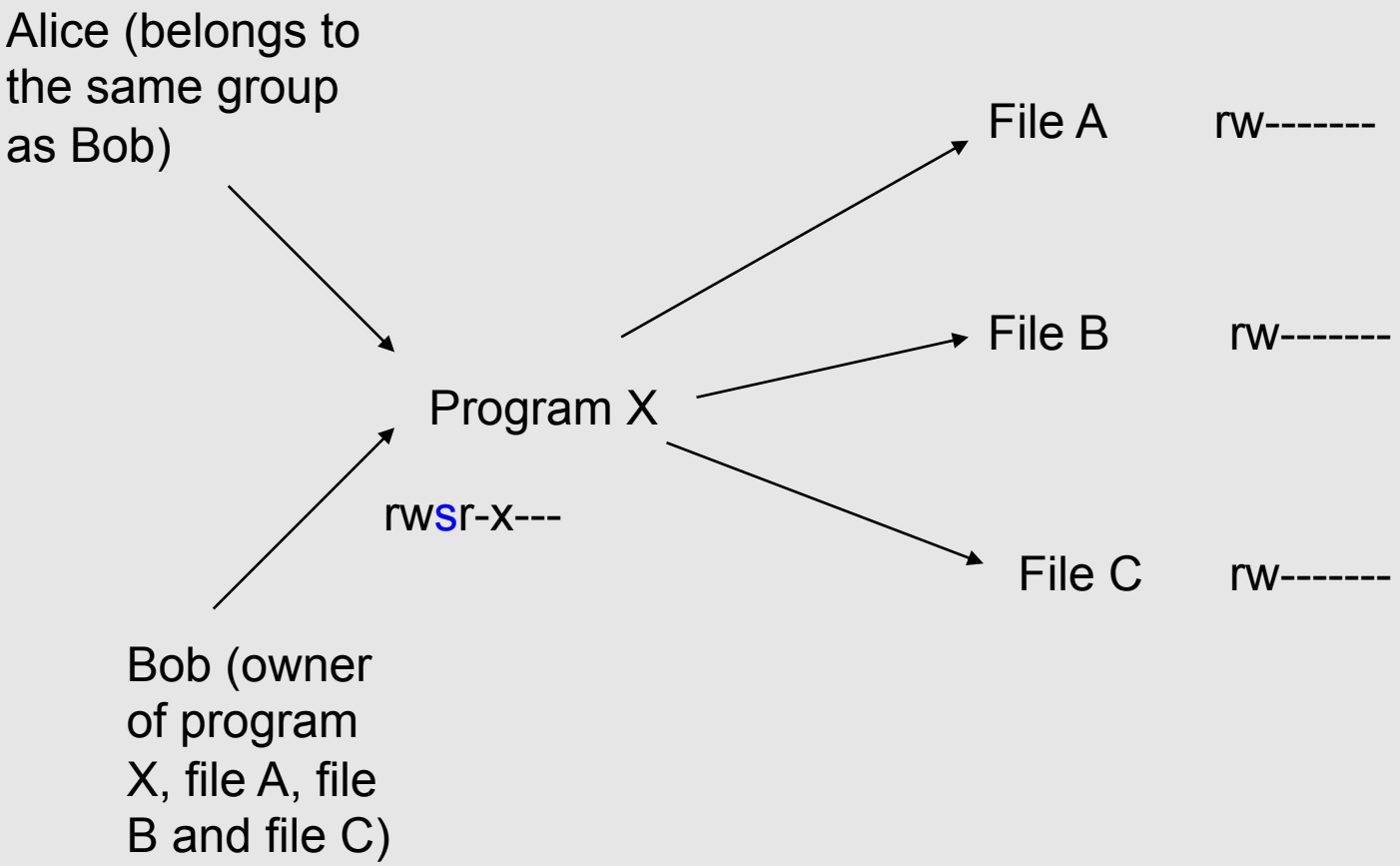
File access for processes

- **When a process executes, it has four id's:**
  - a real user id
  - an effective user id
  - a real group id
  - an effective group id
- **When you log on, your shell process has its real and effective user and group id's same as your user and group id's.**
- **A process can access a file with permissions:**
  - If the process's effective user id is same as the owner of the file then *User* permissions apply
  - Otherwise, if the process's effective group id is the same as file's group id then *Group* permissions apply
  - Otherwise, *Others* permissions apply

Changing ‘operational’ level of access control

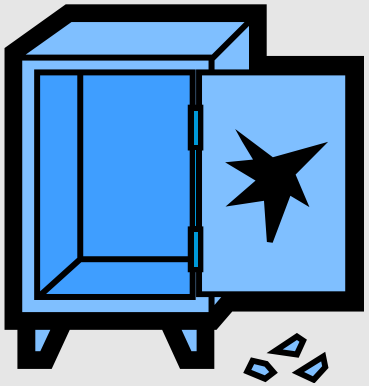
- **Using the SUID and SGID.**
- **Access control in UNIX is defined by the owner of the file.**
- **It is possible to grant a temporary access to files for a specific process (executed program) by setting the SUID or the SGID permission of the program.**
- **A user who can execute the program (has the execute permission to the program) will have access to all the files that the program can read/write, regardless whether this user has read/write permission to the files.**
- **The user received a temporary permission by mean of the SUID or SGID assigned to the program.**

SUID/SGID example

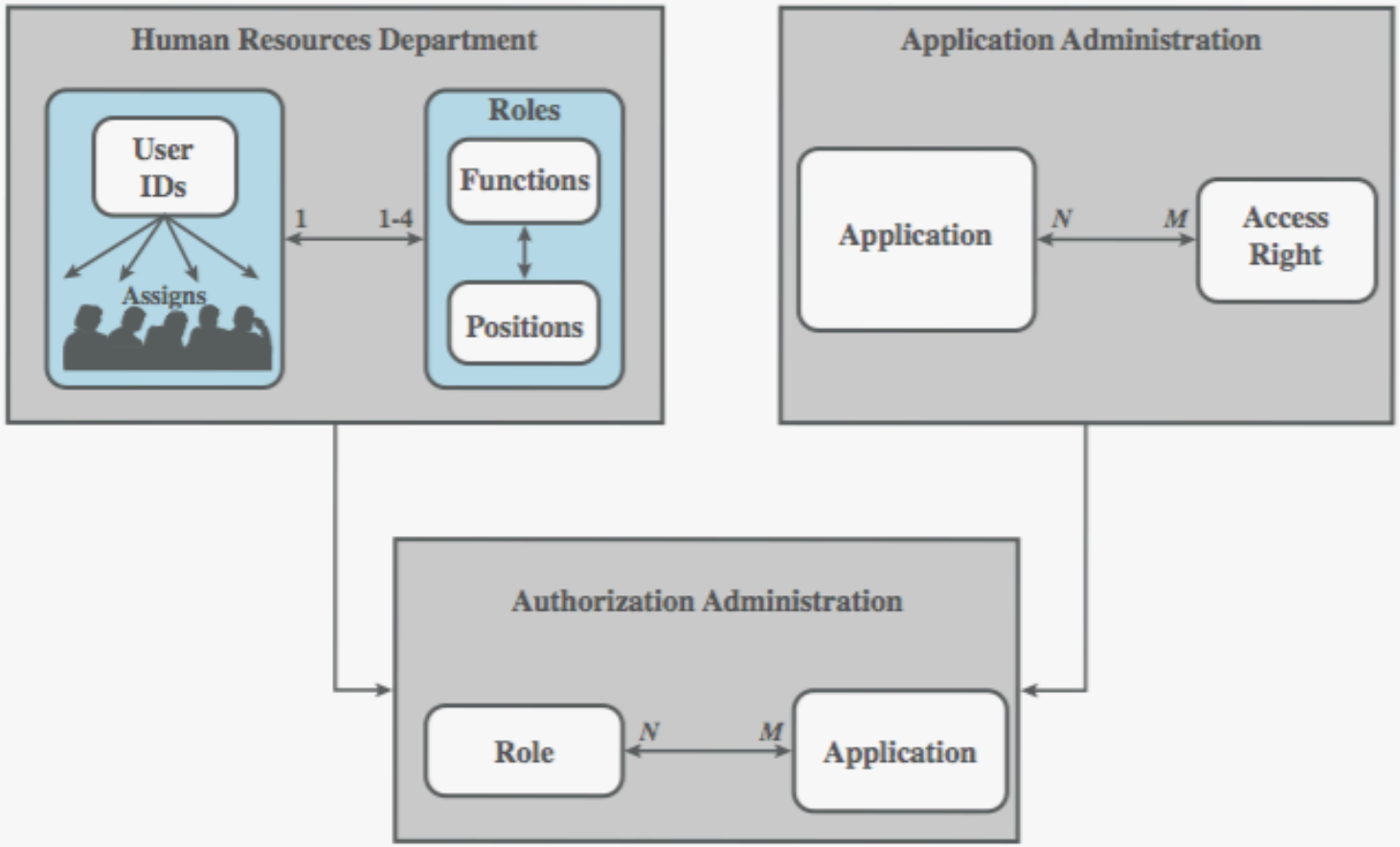


Problems…

- **What happens if someone get access to a program with root SUID?**



RBAC For a Bank



Summary

- **introduced access control principles**
  - subjects, objects, access rights
- **discretionary access controls**
  - access matrix, access control lists (ACLs), capability tickets
- **role-based access control**
- **Security Models**
  - Bell-Lapadula
- **UNIX Access Control mechanisms**
- **case study**

Further Reading

- **Chapters 4 & 13 of the textbook: *Computer Security: Principles and Practice" by William Stallings & Lawrie Brown,* Prentice Hall, 2015**

- **Acknowledgement: part of the materials presented in the slides was developed with the help of Instructor's Manual and other resources made available by the author of the textbook.**