Segurança de Sistemas e dados (MSI 2021/2022)

Aula 5

Rolando Martins
DCC – FCUP
Slides Adaptados do Prof. Manuel Eduardo Correia

Unix FS

Example

UNIX File Concepts

* UNIX files administered using inodes
  * control structure with key info on file
    * attributes, permissions of a single file
  * may have several names for same inode
  * have inode table / list for all files on a disk
    * copied to memory when disk mounted
* directories form a hierarchical tree
  * may contain files or other directories
  * are a file of names and inode numbers

UNIX File Access Control

```
rwxrw----
```

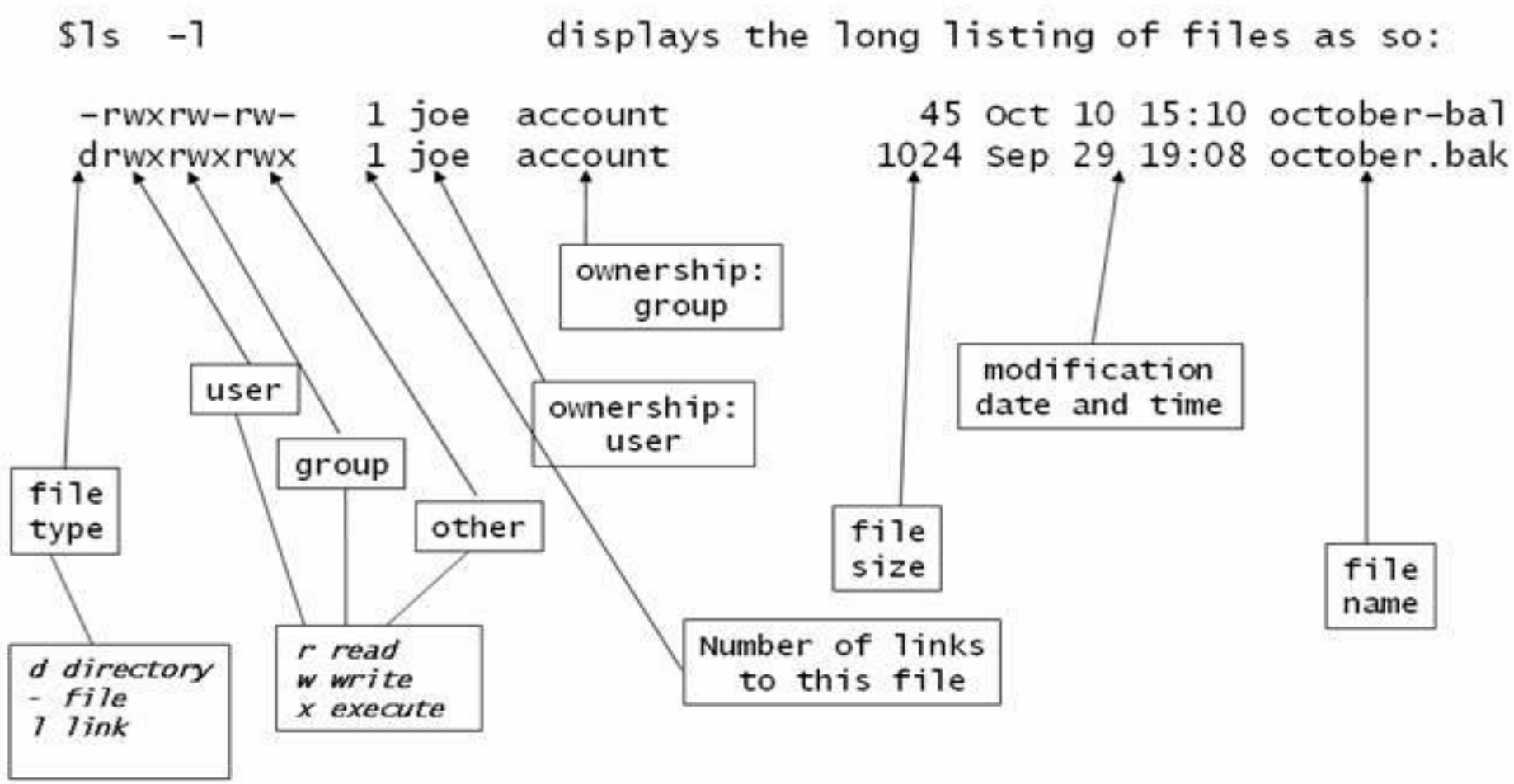
Owner can read, write and execute the file

Any user in the owner ‘s group can read, write the file

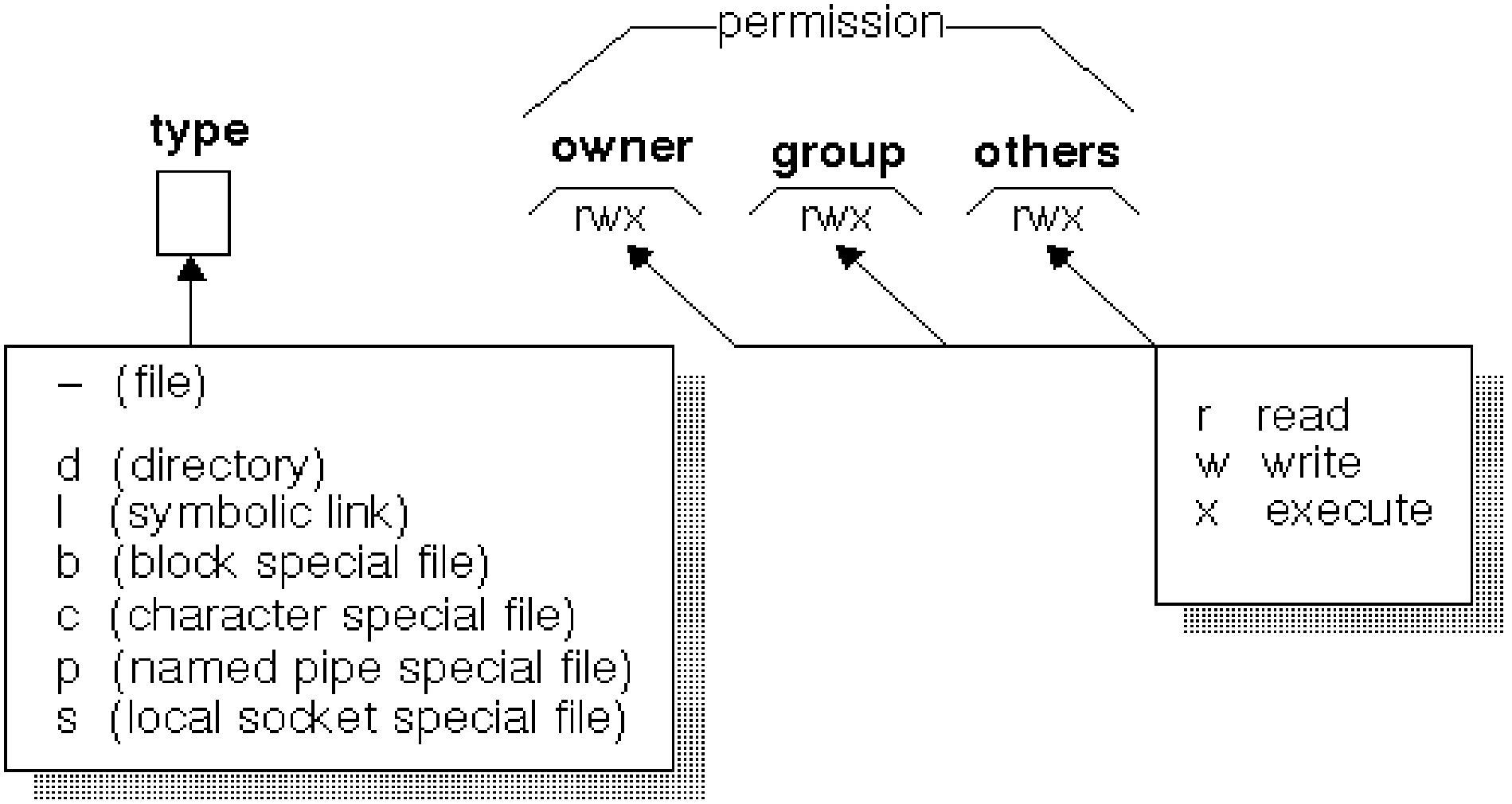
All other users cannot read, write or execute the file

Permissions

file information



Permissions



ZK-0536U-R

Permissions.

Access	File Semantics	Folder Semantics
R	Read the file content	Search the folder content (ex. ls)
W	Modify the file	Change the folder contents. (ex. Remove, rename and create new files inside the folder)
X	Run the file. (**Binary** or **Script** ("**sheebang operator**" #!)	Position the cwd ("current working directory") in the folder (ex. Execute a **"cd"** to that folder or **"cross"** the folder to access another folder inside that folder.).

Permissions examples.

	Minimum permissions required	
Command	For file	For folder
`cd /home/chavez`	N/A	X
`ls /home/chavez/*.c`	(none)	R
`ls -l /home/chavez/*.c`	(none)	R
`cat myfile`	R	X
`cat >>myfile`	W	X
`runme (executável)`	X	X
`cleanup.sh (script)`	RX	X
`rm myfile`	(none)	WX

Permissions for folder

Permissions	Semantics
`--X`	Permite o acesso aos ficheiros da pasta desde que o seu nome seja previamente conhecido.
`R-X`	Permite o acesso e a listagem dos ficheiros da pasta mas não permite que se criem ou apaguem ficheiros.
`-WX`	Usado como uma pasta "**drop in**". Os utilizadores podem posicionar-se na pasta e criar ficheiros mas não conseguem descobrir o nome de ficheiros criados por outros utilizadores. Costuma ser utilizado conjuntamente com o "**sticky** bit".
`rwx`	Acesso total (Também normalmente usado com o "**sticky** bit", exemplo: **/tmp** e **/var/tmp**).

UNIX File Access Control

* "set user ID"(SetUID) or "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
  * on directory limits rename/move/delete to owner
* superuser
  * is exempt from usual access control restrictions

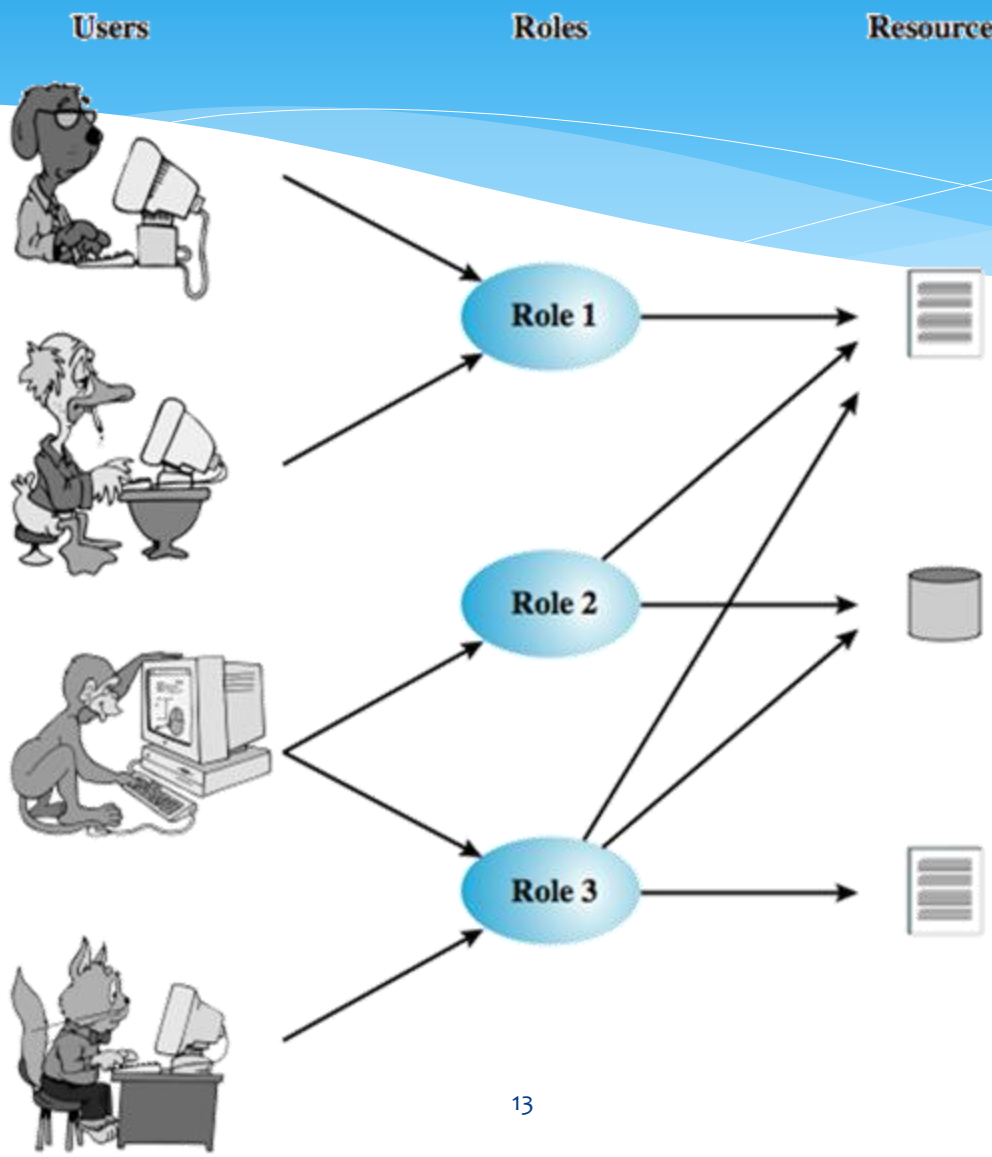
UNIX Access Control Lists

* modern UNIX systems support ACLs
  * See getfacl/setfacl for Linux
* can specify any number of additional users / groups and associated `rwx` permissions
* ACLs are optional extensions to std perms
* group perms also set max ACL perms
* when access is required
  * select most appropriate ACL
    * owner, named users, owning / named groups, others
  * check if have sufficient permissions for access

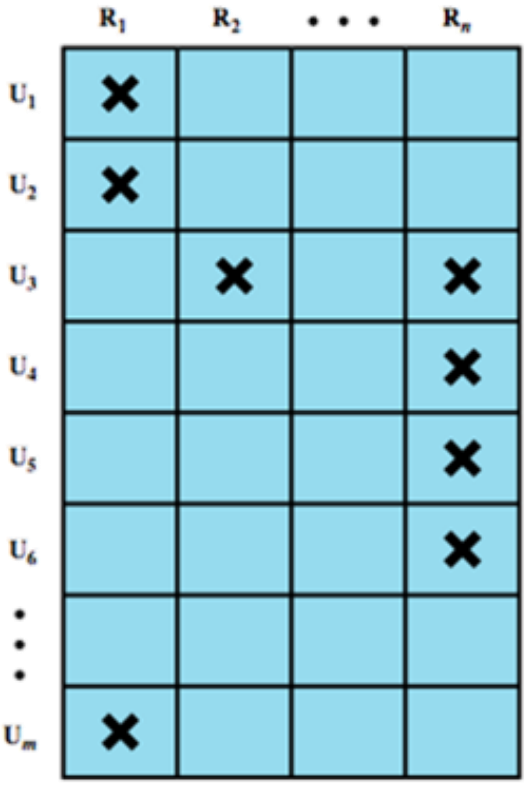
RBAC

Role Based Access Control

Role-Based Access Control

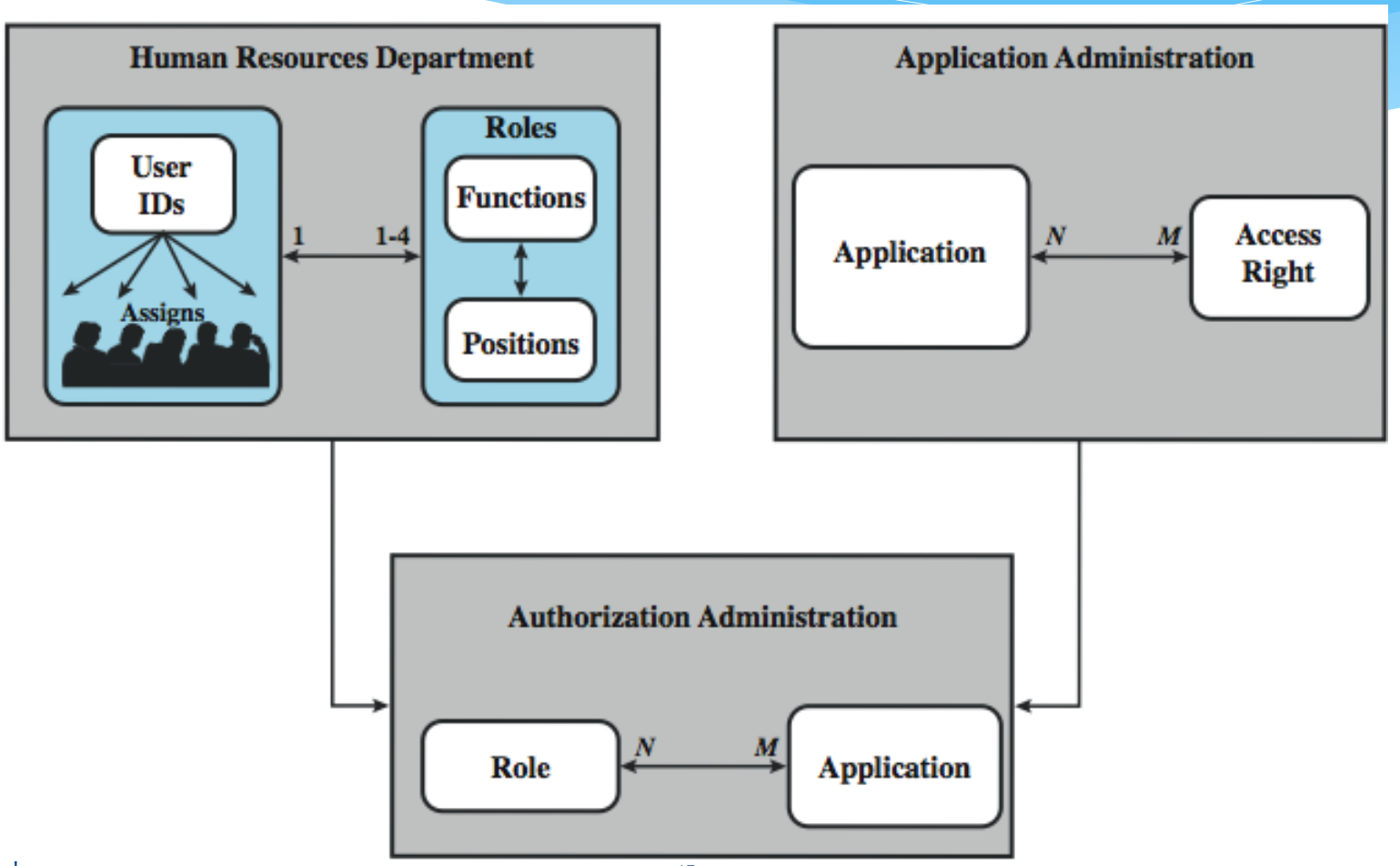


Role-Based Access Control



ROLES	OBJECTS								
	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			

RBAC For a Bank



Summary

* introduced access control principles
  * subjects, objects, access rights
* discretionary access controls
  * access matrix, access control lists (ACLs), capability tickets
  * UNIX traditional and ACL mechanisms
* role-based access control
* case study

MLS Models

Multi Level Security

Classifications and Clearances

* **Classifications** apply to **objects**
* **Clearances** apply to **subjects**
* US Department of Defense (DoD) uses 4 levels:

  TOP SECRET
  SECRET
  CONFIDENTIAL
  UNCLASSIFIED

Clearances and Classification

* To obtain a **SECRET** clearance requires a routine background check
* A **TOP SECRET** clearance requires extensive background check
* Practical classification problems
  * Proper classification not always clear
  * Level of granularity to apply classifications
  * Aggregation — flipside of granularity

Subjects and Objects

* Let O be an **object**, S a **subject**
  * O has a classification
  * S has a clearance
  * Security **level** denoted L(O) and L(S)
* For DoD levels, we have

  TOP SECRET > SECRET >

  CONFIDENTIAL > UNCLASSIFIED

Multilevel Security (MLS)

* MLS needed when subjects/objects at different levels use/on **same system**
* MLS is a form of **Access Control**
* Military and government interest in MLS for many decades
  * Lots of research into MLS
  * Strengths and weaknesses of MLS well understood (but, almost entirely theoretical)
  * Many possible uses of MLS outside military

MLS Applications

* Classified government/military systems
* **Business example:** info restricted to
  * Senior management only, all management, everyone in company, or general public
* Network firewall
* Confidential medical info, databases, etc.
* Usually, MLS not a viable technical system
  * More of a legal device than technical system

MLS Security Models

* MLS models explain **what** needs to be done
* Models **do not** tell you **how** to implement
* Models are descriptive, not prescriptive
  * That is, high level description, not an algorithm
* There are many MLS models
* We'll discuss simplest MLS model
  * Other models are more realistic
  * Other models also more complex, more difficult to enforce, harder to verify, etc.

Bell-LaPadula

* BLP security model designed to express essential requirements for MLS
* BLP deals with **confidentiality**
  * To prevent unauthorized reading
* Recall that O is an object, S a subject
  * Object O has a classification
  * Subject S has a clearance
  * Security level denoted $L(O)$ and $L(S)$

Bell-LaPadula

* BLP consists of

  Simple Security Condition: S can read O if and only if $L(O) \leq L(S)$

  ***-Property** (**Star Property**): S can write O if and only if $L(S) \leq L(O)$

* **No read up, no write down**

McLean's Criticisms of BLP

* McLean: BLP is "so trivial that it is hard to imagine a realistic security model for which it does not hold"
* McLean's "system Z" allowed administrator to reclassify object, then "write down"
* Is this fair?
* Violates spirit of BLP, but **not** expressly forbidden in statement of BLP
* Raises fundamental questions about the nature of (and limits of) modeling

B and LP's Response

* BLP enhanced with **tranquility property**
  * Strong tranquility: security labels never change
  * Weak tranquility: security label can only change if it does not violate "established security policy"
* Strong tranquility impractical in real world
  * Often want to enforce "least privilege"
  * Give users lowest privilege for current work
  * Then upgrade as needed (and allowed by policy)
  * This is known as the **high water mark** principle
* Weak tranquility allows for **least privilege** (high water mark), but the property is vague

BLP: The Bottom Line

* BLP is simple, probably too simple
* BLP is one of the few security models that can be used to prove things about systems
* BLP has inspired other security models
  * Most other models try to be more realistic
  * Other security models are more complex
  * Models difficult to analyze, apply in practice

Covert Channel

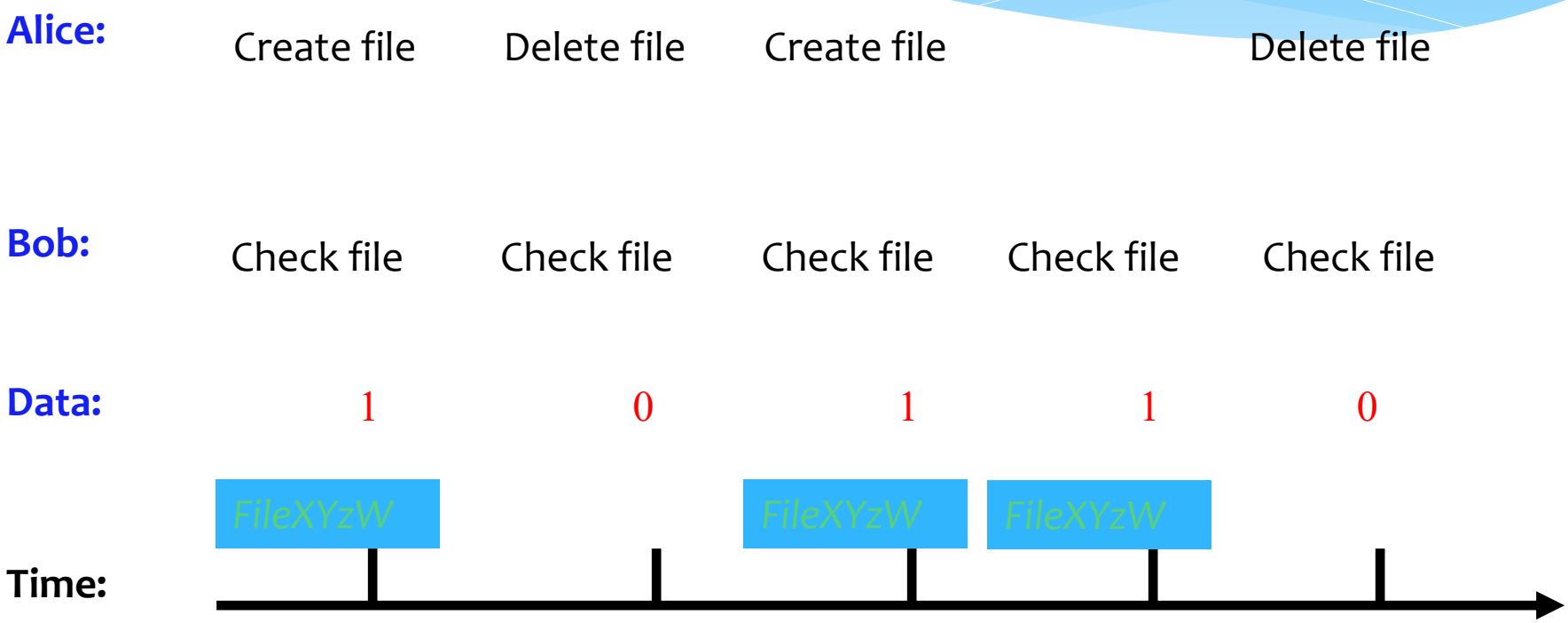
Covert Channel

* MLS designed to restrict legitimate channels of communication
* May be other ways for information to flow
* For example, resources shared at different levels could be used to "signal" information
* **Covert channel**: a communication path not intended as such by system's designers

Covert Channel Example

* Alice has **TOP SECRET** clearance, Bob has **CONFIDENTIAL** clearance
* Suppose the file space shared by all users
* Alice creates file FileXYzW to signal "1" to Bob, and removes file to signal "0"
* Once per minute Bob lists the files
  * If file FileXYzW does not exist, Alice sent 0
  * If file FileXYzW exists, Alice sent 1
* Alice can leak **TOP SECRET** info to Bob!

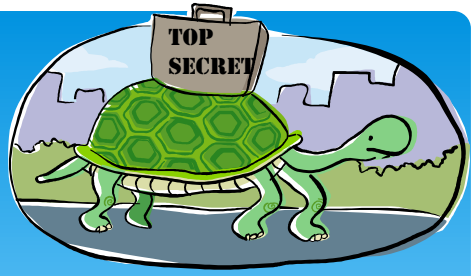
Covert Channel Example



Covert Channel

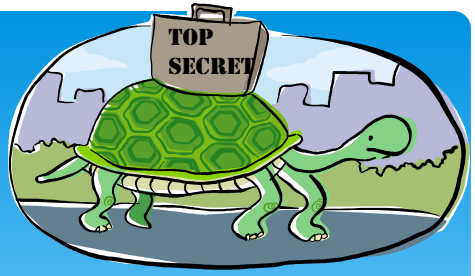
* Other possible covert channels?
  * Print queue
  * ACK messages
  * Network traffic, etc.
* When does covert channel exist?
  1. Sender and receiver have a shared resource
  2. Sender able to vary some property of resource that receiver can observe
  3. "Communication" between sender and receiver can be synchronized

Covert Channel



* So, covert channels are everywhere
* "Easy" to eliminate covert channels:
  * Eliminate all shared resources…
  * … and all communication
* Virtually impossible to eliminate covert channels in any useful system
  * DoD guidelines: **reduce covert channel capacity** to no more than 1 bit/second
  * Implication? DoD has given up on *eliminating* covert channels!

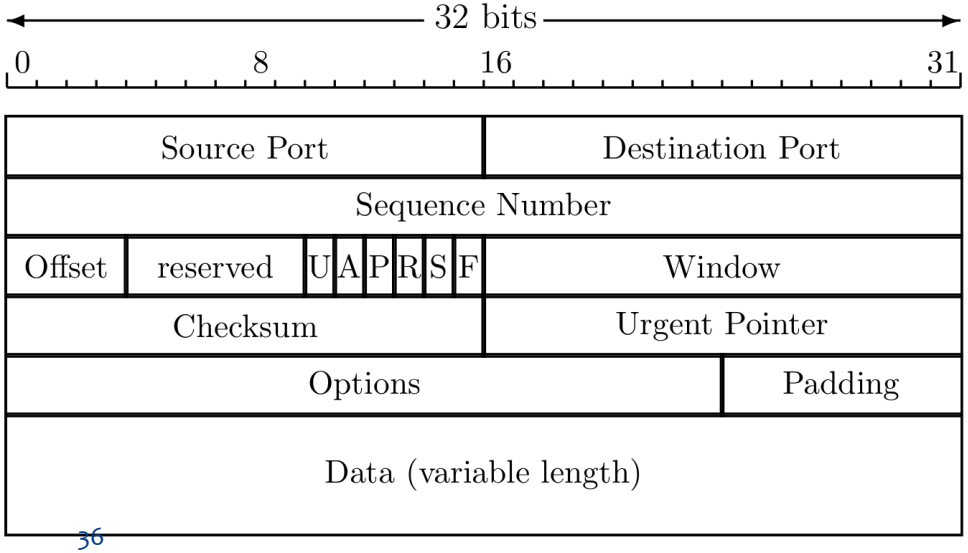
Covert Channel



* Consider 100MB **TOP SECRET** file
  * Plaintext stored in **TOP SECRET** location
  * Ciphertext (encrypted with AES using 256-bit key) stored in **UNCLASSIFIED** location
* Suppose we reduce covert channel capacity to 1 bit per second
* It would take more than 25 years to leak entire document thru a covert channel
* But it would take less than 5 minutes to leak 256-bit AES key thru covert channel!

Real-World Covert Channel

* Hide data in TCP header "reserved" field
* Or use covert_TCP, tool to hide data in
  * Sequence number
  * ACK number



Real-World Covert Channel

* Hide data in TCP sequence numbers
* Tool: covert_TCP
* Sequence number X contains covert info