Developed and Presented By Dr. Mehrdad Sepehri Sharbaf
CSUDH
Computer Science Department

http://csc.csudh.edu/

The some of the materials are excerpted from Michael T. Goodrich & Roberto Tamassia's Book, and Ross Anderson's Book

ACCESS CONTROL GOAL

ACCESS CONTROL

- Access Control is one of the most popular areas of Information Security.
- It consists of many levels and mechanisms.
  - Application level.
  - Middleware.
  - Operating system.
  - Hardware controls (e.g.memory management).

These are just the basic levels.

ACCESS CONTROL

- We see the following access control mechanisms.
  - OS access controls for user authentication, isomorphic to an ACL, CL, or ACM.
  - Groups (lists of principals).
  - Roles (fixed set of permissions that principals may assume).
  - Access Control Lists.
  - Capabilities.
- There are also a lot of granularity issues.

ACCESS CONTROL

- What goes wrong with Access Control?
  - Stack smashing.
  - Race conditions and other bugs.
  - Denial of service bugs.
  - User interface failures (Trojan horse).
  - Allowing wrong programs to run as root.
  - Allowing too much privilege.
- Problems are usually caused by structural bloat where the kernel gets too big to properly manage.

TOPIC: ACCESS CONTROL

- Users and groups
- Authentication
- Passwords
- File protection
- Access control lists

- Which users can read/write which files?
- Are my files really safe?
- What does it mean to be root?
- What do we really want to control?

ACCESS CONTROL MATRICES

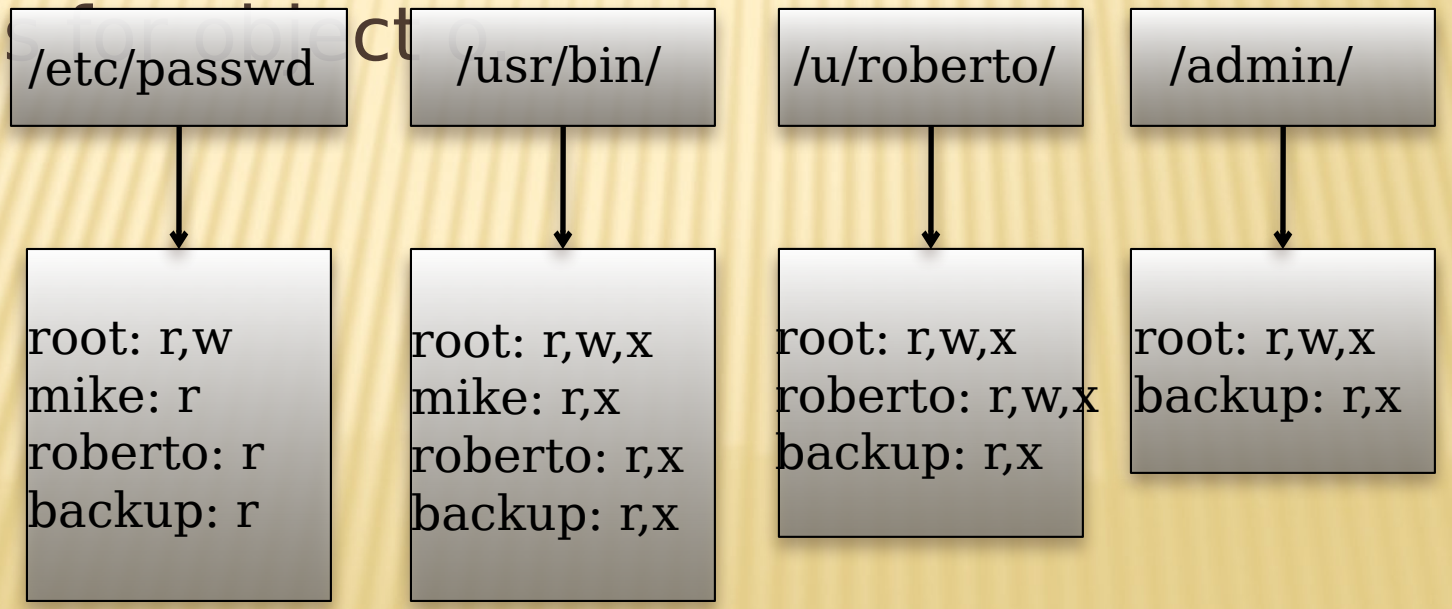
- **A table that defines permissions**.
  - Each row of this table is associated with a **subject,** which is a user, group, or system that can perform actions.
  - Each column of the table is associated with an **object,** which is a file, directory, document, device, resource, or any other entity for 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
. . .	. . .	. . .	. . .	. . .

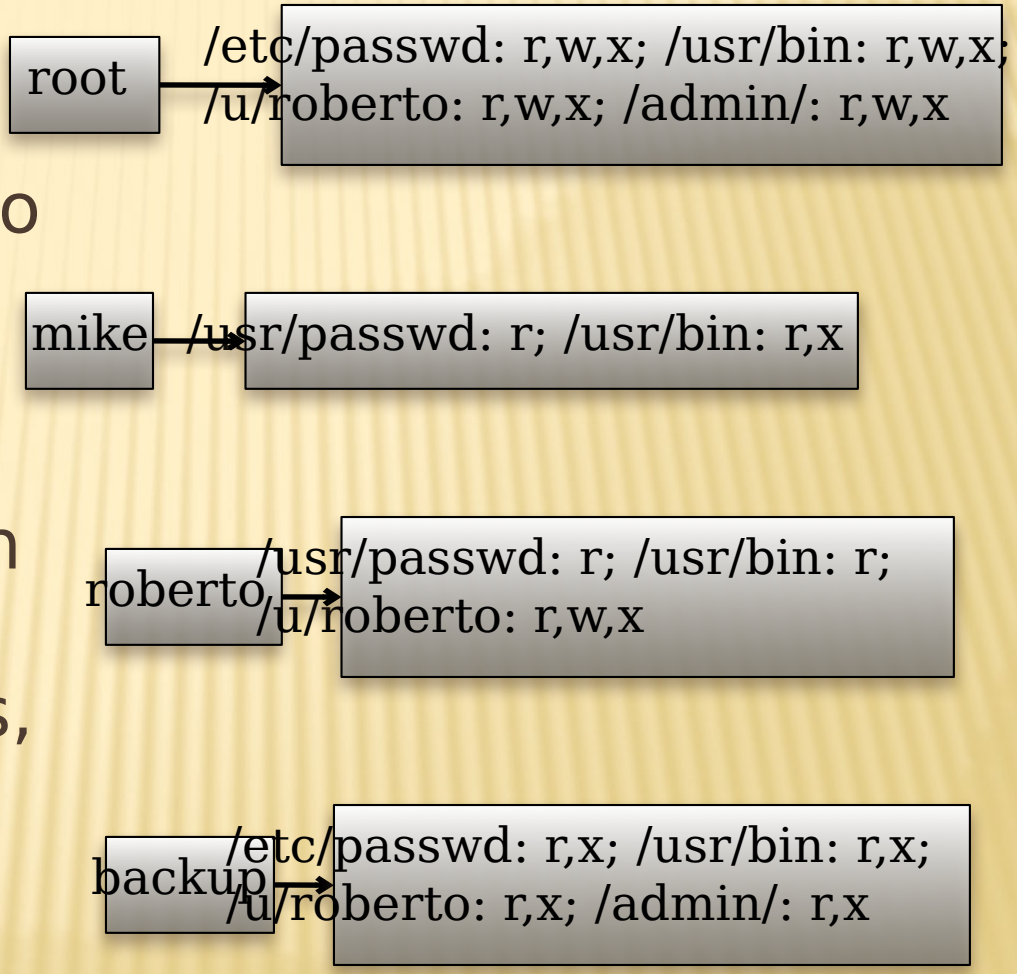
ACCESS CONTROL LISTS

- It 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.



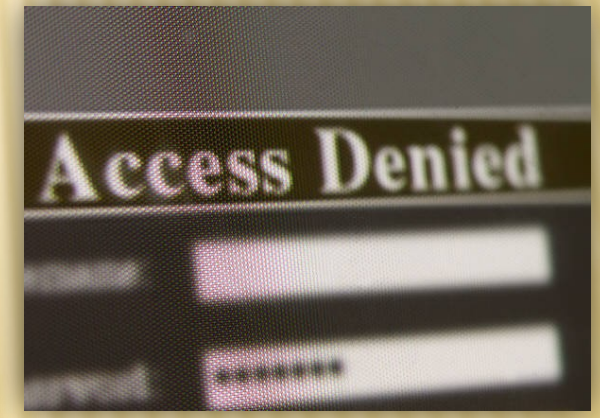
CAPABILITIES

- Takes 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.



ACCESS CONTROL MODELS

- Various models have been developed to formalize mechanisms to protect the confidentiality and integrity of documents stored in a computer system.
  - The Bell-La Padula (BLP) model
  - The Biba model
  - The Low-Watermark model
  - The Clark-Wilson model
  - The Chinese Wall model (The Brewer and Nash model)

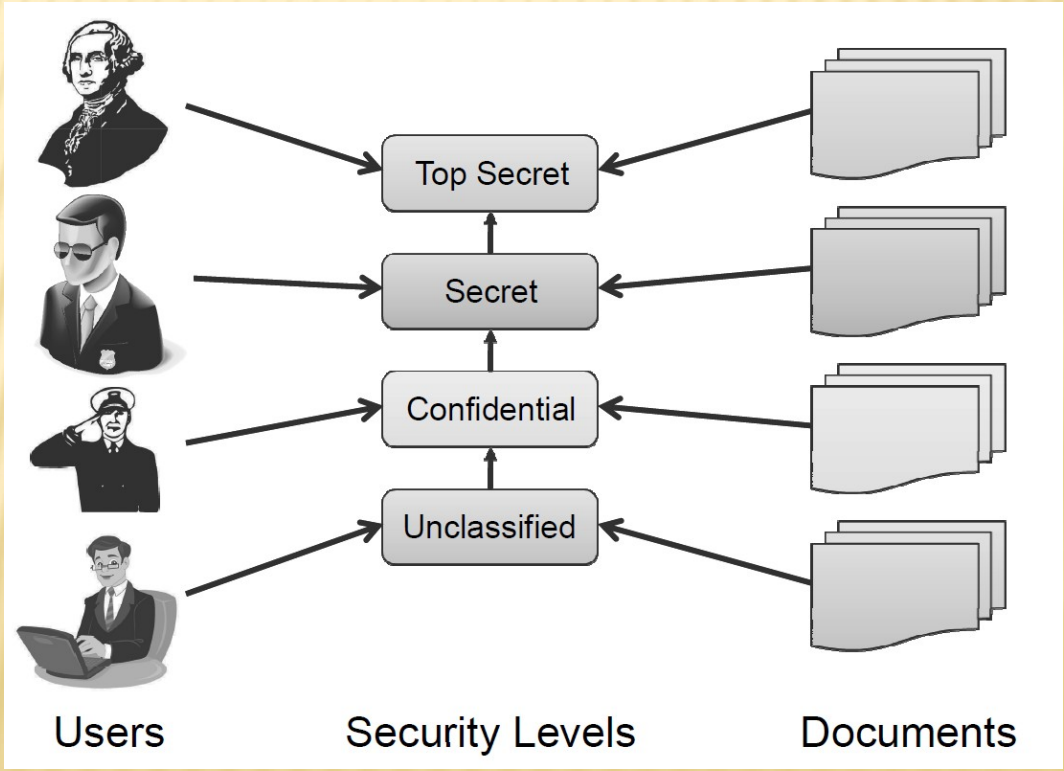


THE BELL-LA PADULA MODEL

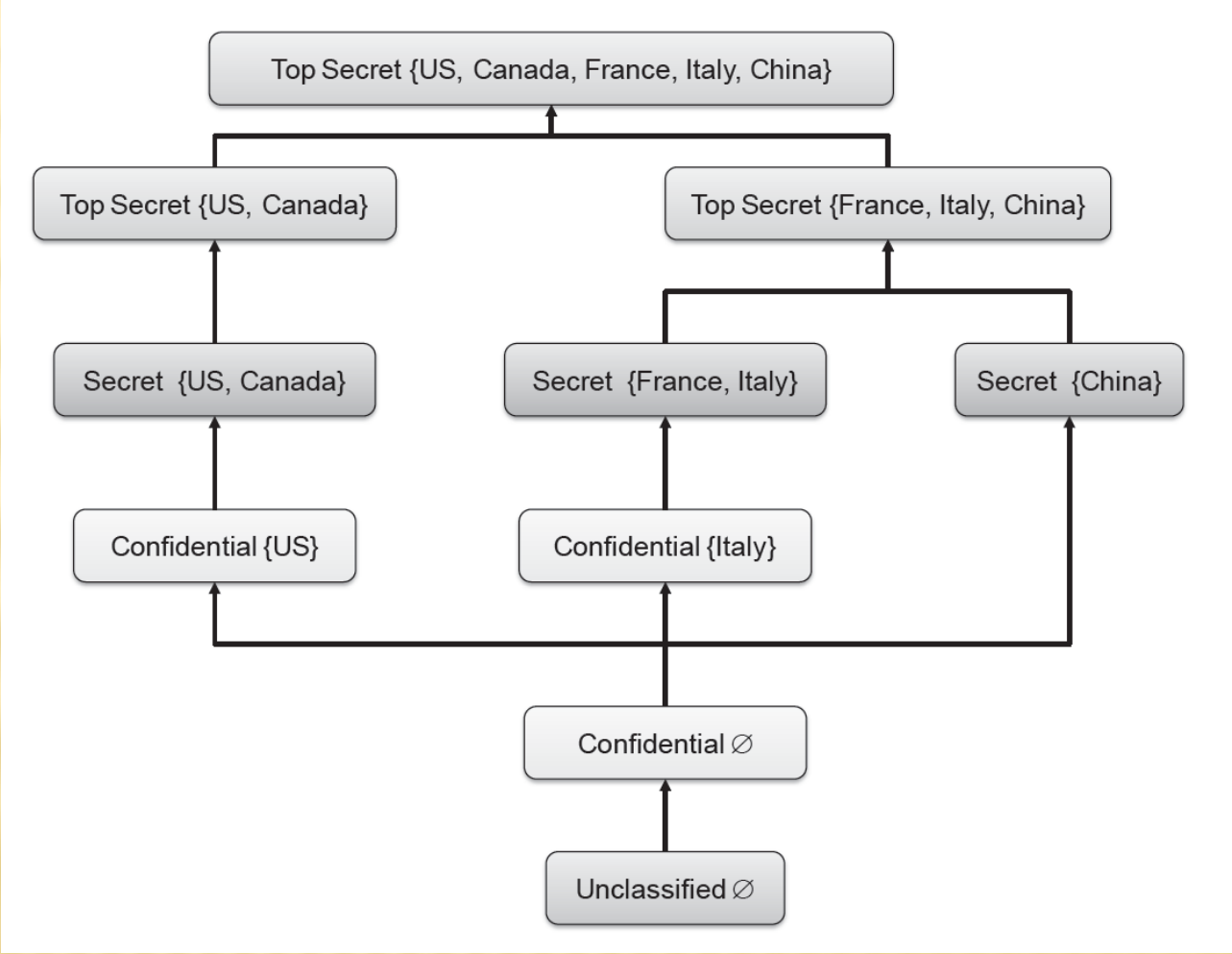
- The **Bell-La Padula (BLP) model** is a classic mandatory access-control model for protecting confidentiality.
- The BLP model is derived from the military **multilevel security paradigm,** which has been traditionally used in military organizations for document classification and personnel clearance.

THE BELL-LA PADULA MODEL

- The BLP model has a strict, linear ordering on the security of levels of documents, so that each document has a specific security level in this ordering and each user is assigned a strict level of access that allows them to view all documents with the corresponding level of security or below.



DEFINING SECURITY LEVELS USING CATEGORIES



THE BIBA MODEL

- The **Biba model** has a similar structure to the BLP model, but it addresses integrity rather than confidentiality.
- Objects and users are assigned **integrity levels** that form a partial order, similar to the BLP model.
- The integrity levels in the Biba model indicate degrees of trustworthiness, or accuracy, for objects and users, rather than levels for determining confidentiality.
  - For example, a file stored on a machine in a closely monitored data center would be assigned a higher integrity level than a file stored on a laptop.
  - In general, a data-center computer is less likely to be compromised than a random laptop computer. Likewise, when it comes to users, a senior employee with years of experience would have a higher integrity level than an intern.

THE BIBA MODEL RULES

- The access-control rules for Biba are the reverse of those for BLP. That is, Biba does not allow reading from lower levels and writing to upper levels.
- If we let I(u) denote the integrity level of a user u and I(x) denote the integrity level for an object, x, we have the following rules in the Biba model:
  - A user u can read an object x only if

    $I(u) \leq I(x)$.
  - A user u can write (create, edit or append to) an object x only if

    $I(x) \leq I(u)$.
- Thus, the Biba rules express the principle that information can only flow down, going from higher integrity levels to lower integrity levels.

THE LOW-WATERMARK MODEL

- The **low-watermark model** is an extension to the Biba model that relaxes the “no read down” restriction, but is otherwise similar to the Biba model.
- In other words, users with higher integrity levels can read objects with lower integrity levels.
- After such a reading, the user performing the reading is demoted such that his integrity level matches that of the read object.

THE CLARK-WILSON MODEL

- Rather than dealing with document confidentiality and/or integrity, the **Clark-Wilson (CW)** model deals with systems that perform transactions.
- It describes mechanisms for assuring that the integrity of such a system is preserved across the execution of a transaction. Key components of the CW model include the following:
  - **Integrity constraints** that express relationships among objects that must be satisfied for the system state to be valid. A classic example of an integrity constraint is the relationship stating that the final balance of a bank account after a withdrawal transaction must be equal to the initial balance minus the amount withdrawn.
  - **Certification methods** that verify that transactions meet given integrity constraints. Once the program for a transaction is certified, the integrity constraints do not need to be verified at each execution of the transaction.
  - **Separation of duty rules** that prevent a user that executes transaction from certifying it. In general, each transaction is assigned disjoint sets of users that can certify and execute it, respectively.

THE CHINESE WALL MODEL

- The **Brewer and Nash model,** commonly referred to as the **Chinese wall model,** is designed for use in the commercial sector to eliminate the possibility of conflicts of interest.
- To achieve this, the model groups resources into “conflict of interest classes.”
- The model enforces the restriction that each user can only access one resource from each conflict of interest class.
  - In the financial world, such a model might be used, for instance, to prevent market analysts from receiving insider information from one company and using that information to provide advice to that company’s competitor.
- Such a policy might be implemented on computer systems to regulate users’ access to sensitive or

ROLE-BASED ACCESS CONTROL

- The **role-based access control (RBAC)** model can be viewed as an evolution of the notion of group-based permissions in file systems.
- An RBAC system is defined with respect to an organization, such as company, a set of resources, such as documents, print services, and network services, and a set of users, such as employees, suppliers, and customers.

U.S. Navy image in the public domain.

RBAC COMPONENTS

- A **user** is an entity that wishes to access resources of the organization to perform a task. Usually, users are actual human users, but a user can also be a machine or application.
- A **role** is defined as a collection of users with similar functions and responsibilities in the organization. Examples of roles in a university may include “student,” “alum,” “faculty,” “dean,” “staff,” and “contractor.” In general, a user may have multiple roles.
  - Roles and their functions are often specified in the written documents of the organization.
  - The assignment of users to roles follows resolutions by the organization, such as employment actions (e.g., hiring and resignation) and academic actions (e.g., admission and graduation).
- A **permission** describes an allowed method of access to a resource**.**
  - More specifically, a permission consists of an operation performed on an object, such as “read a file” or “open a network connection.” Each role has an associated set of permissions.
- A **session** consists of the activation of a subset of the roles of a user for the purpose of performing a certain task.

HIERARCHICAL RBAC

- In the role-based access control model, roles can be structured in a hierarchy similar to an organization chart.
- More formally, we define a partial order among roles by saying that a role R1 **inherits role R2,** which is denoted

  $R1 \geq R2$,

  if R1 includes all permissions of R2 and R2 includes all users of R1.
- When $R1 \geq R2$, we also say that role R1 is **senior** to role R2 and that role R2 is **junior** to role R1.
  - For example, in a company, the role “manager” inherits the role “employee” and the role “vice president” inherits the role “manager.”
  - Also, in a university, the roles “undergraduate student” and “graduate student” inherit the role “student.”

VISUALIZING ROLE HIERARCHY