Intrusion Detection

Continued

Tom Longstaff

CERT^{SM} Coordination Center
Software Engineering Institute
Carnegie Mellon University
Pittsburgh PA 1521

The CERT Coordination Center is sponsored by the Advanced Research Projects Agency (ARPA). The Software Engineering Institute is sponsored by the U.S. Department of Defense.

Intrusion Detection Types

Pattern Detection
Anomaly Detection
Policy-Based Detection

handout

Main Points on IDS

Most do not add any protection at all, but can help with recognition

Not a lot of science involved in IDS design

Measurement is problematic

The future of IDS is uncertain

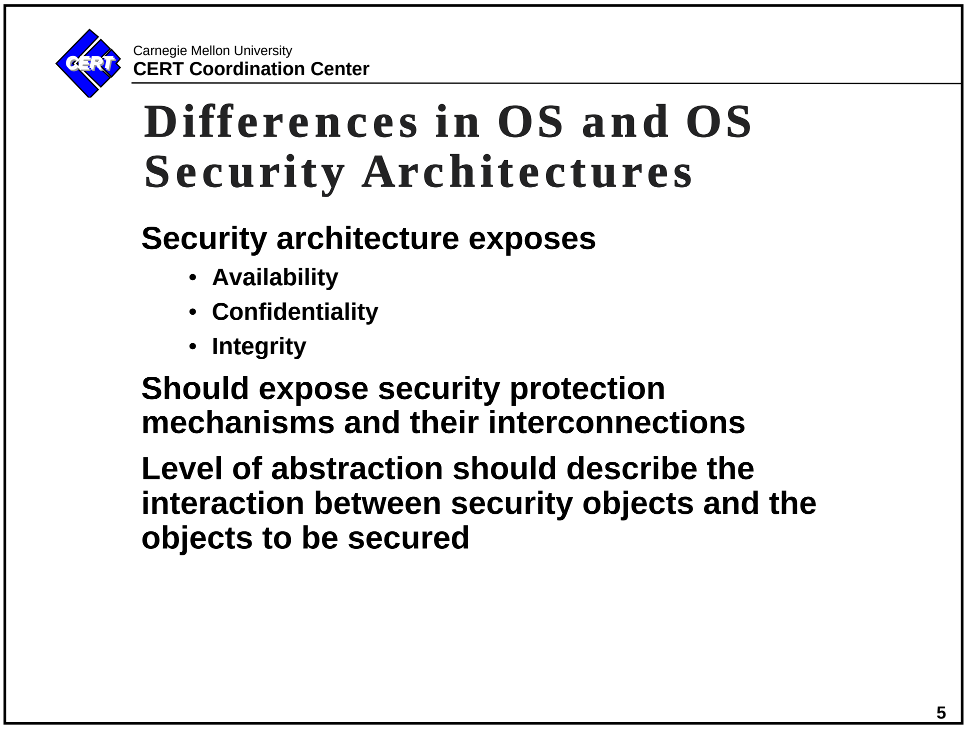
Real key is to focus on incident response

Operating System Security Architectures

Tom Longstaff

CERT Coordination Center
Software^{SM} Engineering Institute
Carnegie Mellon University
Pittsburgh PA 1521

The CERT Coordination Center is sponsored by the Advanced Research Projects Agency (ARPA). The Software Engineering Institute is sponsored by the U.S. Department of Defense.



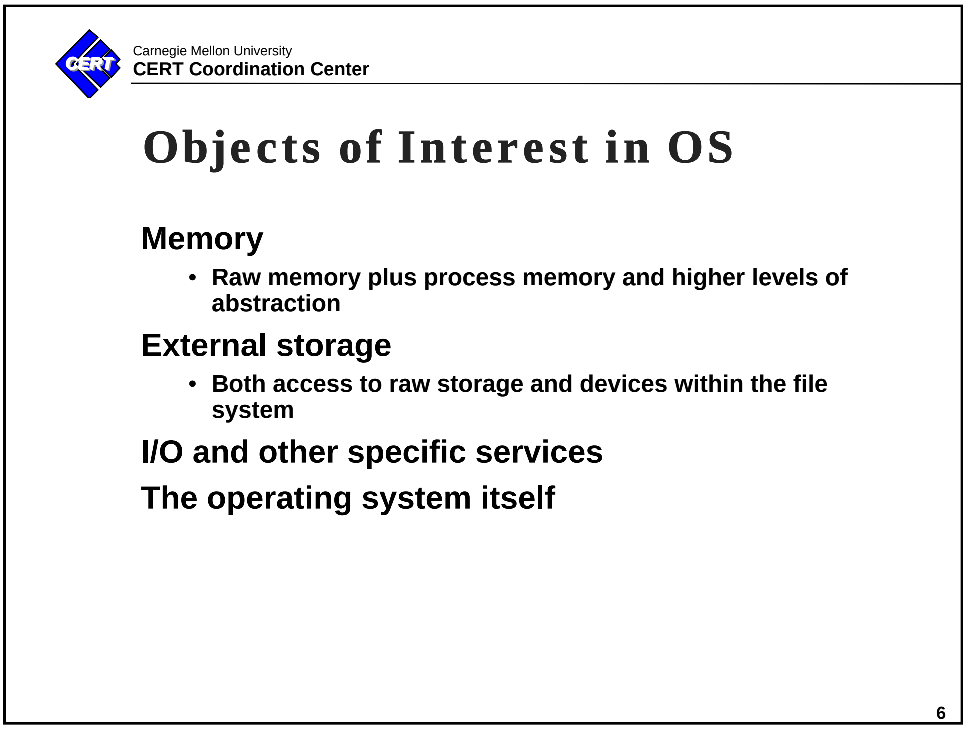
Differences in OS and OS Security Architectures

Security architecture exposes

- Availability
- Confidentiality
- Integrity

Should expose security protection mechanisms and their interconnections

Level of abstraction should describe the interaction between security objects and the objects to be secured



Objects of Interest in OS

Memory

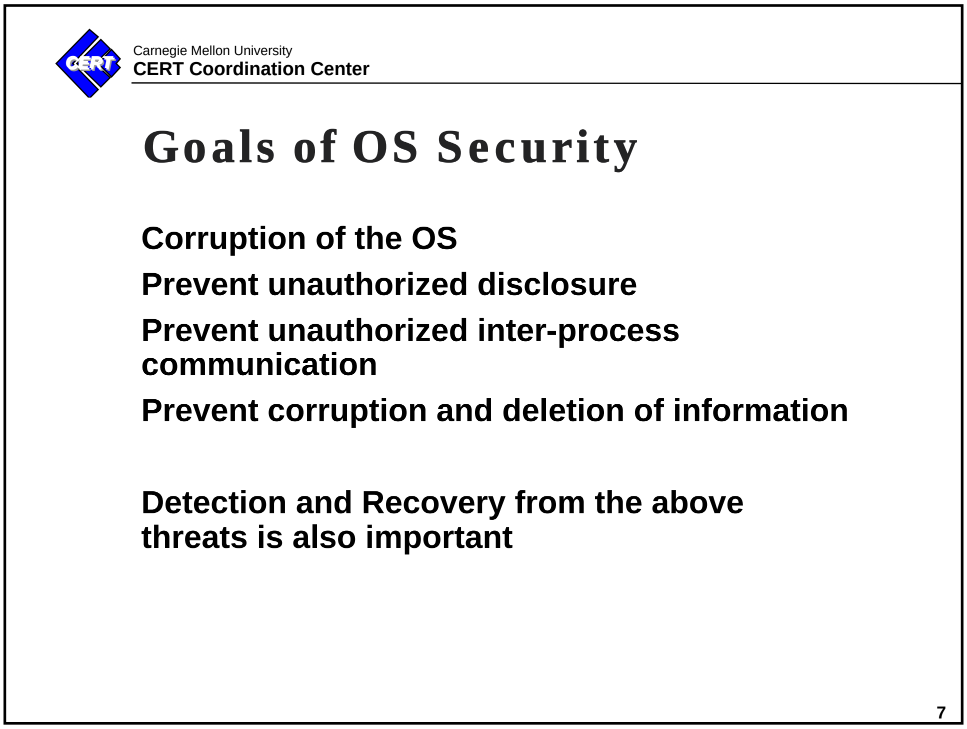
- Raw memory plus process memory and higher levels of abstraction

External storage

- Both access to raw storage and devices within the file system

I/O and other specific services

The operating system itself



Goals of OS Security

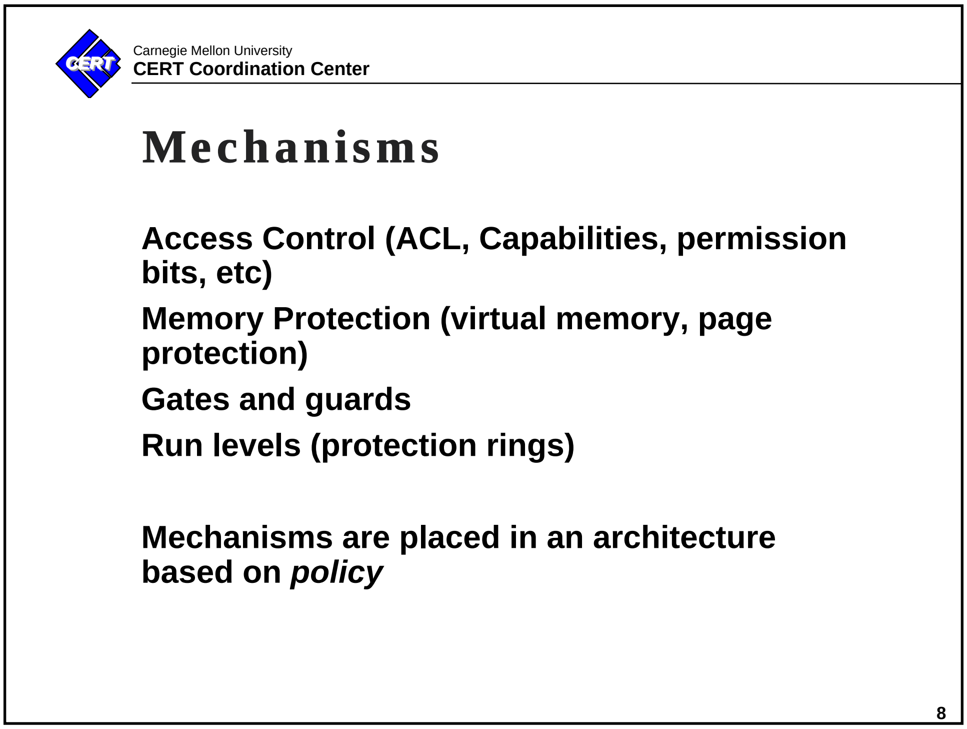
Corruption of the OS

Prevent unauthorized disclosure

Prevent unauthorized inter-process communication

Prevent corruption and deletion of information

Detection and Recovery from the above threats is also important



Mechanisms

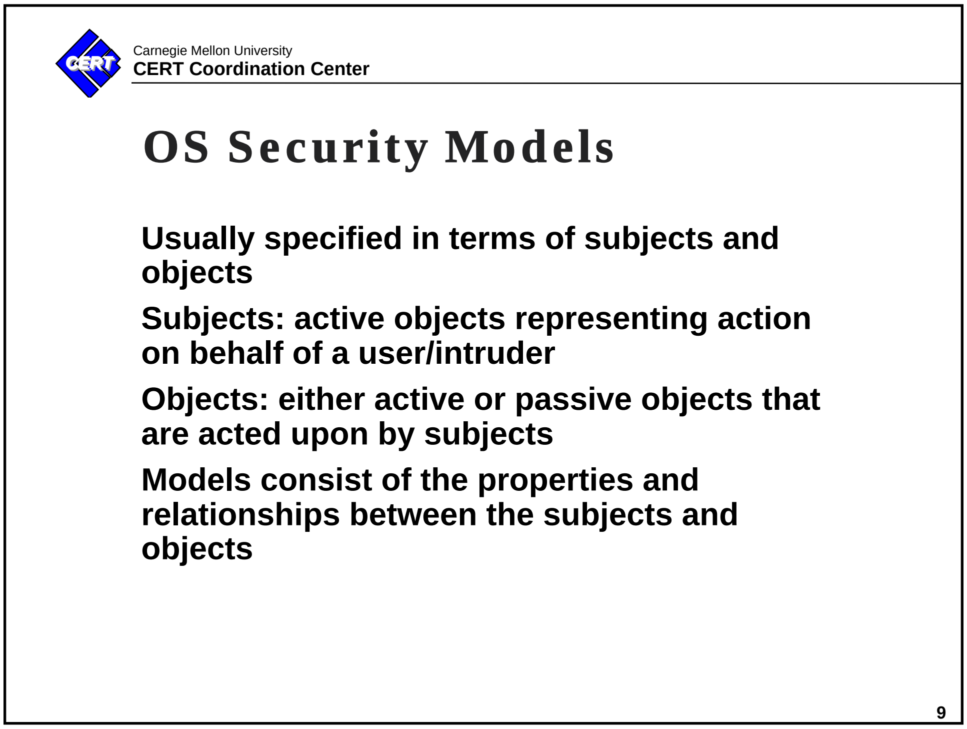
Access Control (ACL, Capabilities, permission bits, etc)

Memory Protection (virtual memory, page protection)

Gates and guards

Run levels (protection rings)

Mechanisms are placed in an architecture based on *policy*



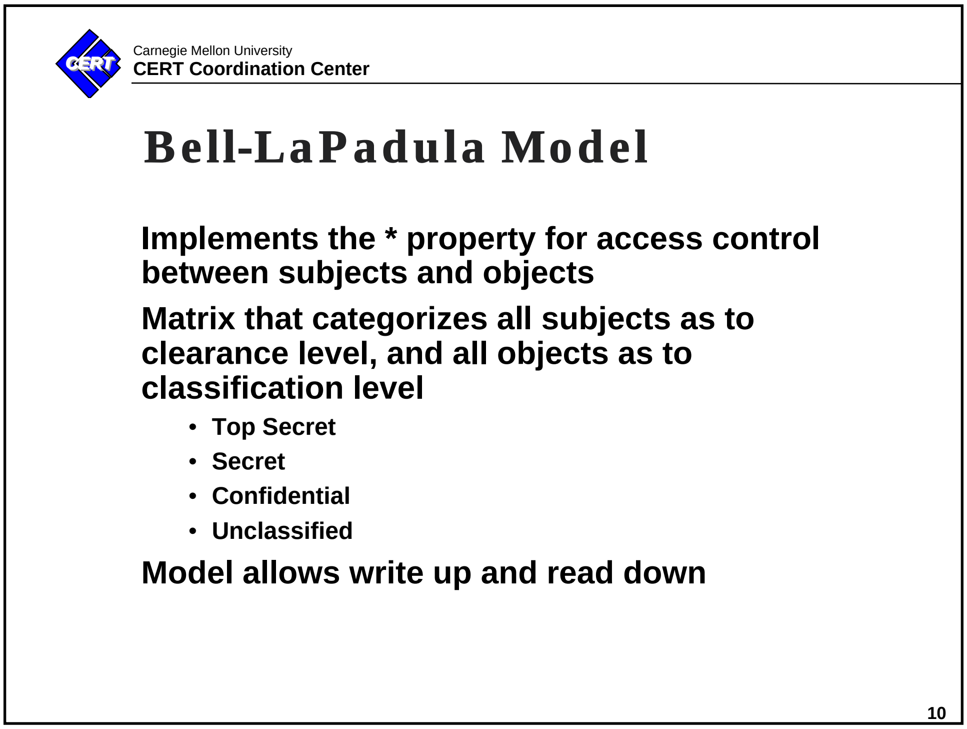
OS Security Models

Usually specified in terms of subjects and objects

Subjects: active objects representing action on behalf of a user/intruder

Objects: either active or passive objects that are acted upon by subjects

Models consist of the properties and relationships between the subjects and objects



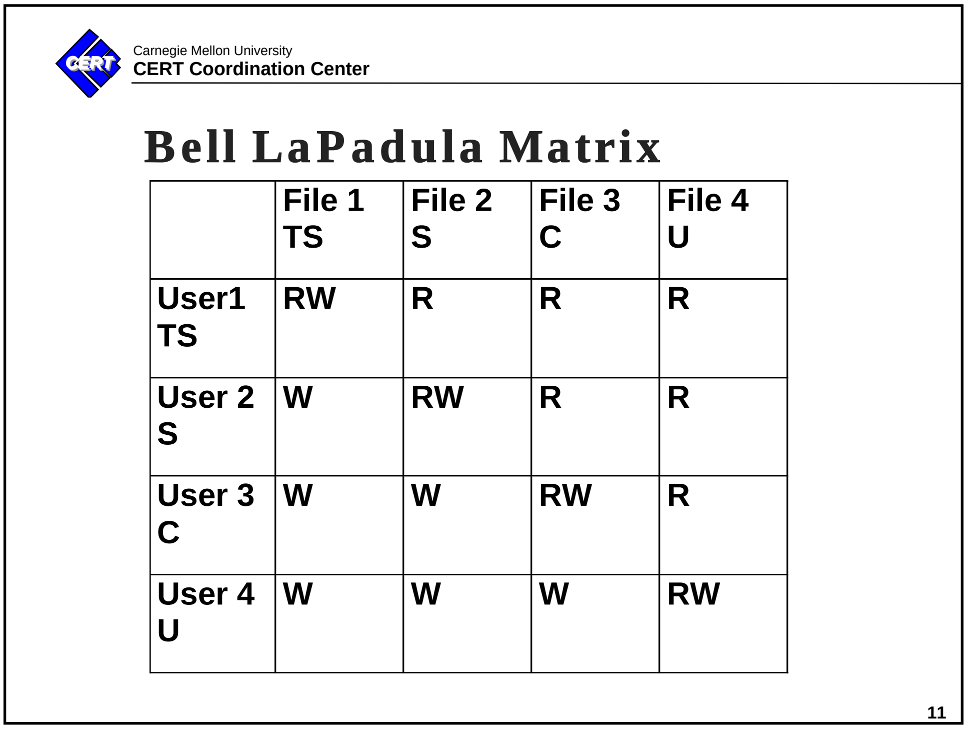
Bell-LaPadula Model

Implements the * property for access control between subjects and objects

Matrix that categorizes all subjects as to clearance level, and all objects as to classification level

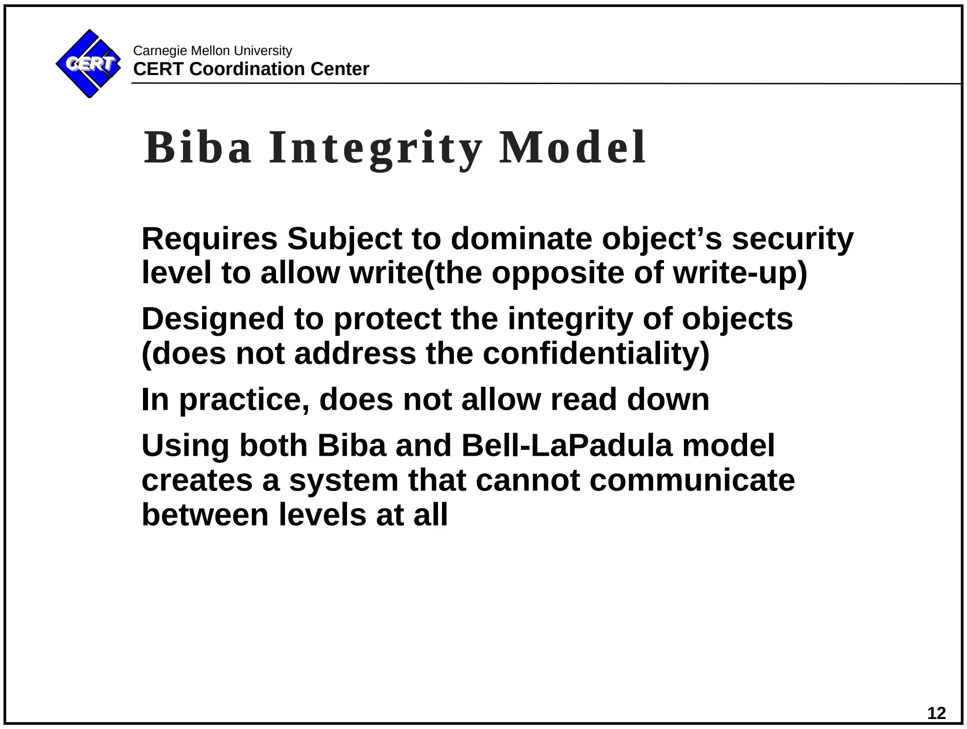
- Top Secret
- Secret
- Confidential
- Unclassified

Model allows write up and read down



Bell LaPadula Matrix

	File 1 TS	File 2 S	File 3 C	File 4 U
User1 TS	RW	R	R	R
User 2 S	W	RW	R	R
User 3 C	W	W	RW	R
User 4 U	W	W	W	RW



Biba Integrity Model

Requires Subject to dominate object’s security level to allow write(the opposite of write-up)

Designed to protect the integrity of objects (does not address the confidentiality)

In practice, does not allow read down

Using both Biba and Bell-LaPadula model creates a system that cannot communicate between levels at all

Biba Matrix

	File 1 TS	File 2 S	File 3 C	File 4 U
User 1 TS	RW	RW	RW	RW
User 2 S	*	RW	RW	RW
User 3 C	*	*	RW	RW
User 4 U	*	*	*	RW

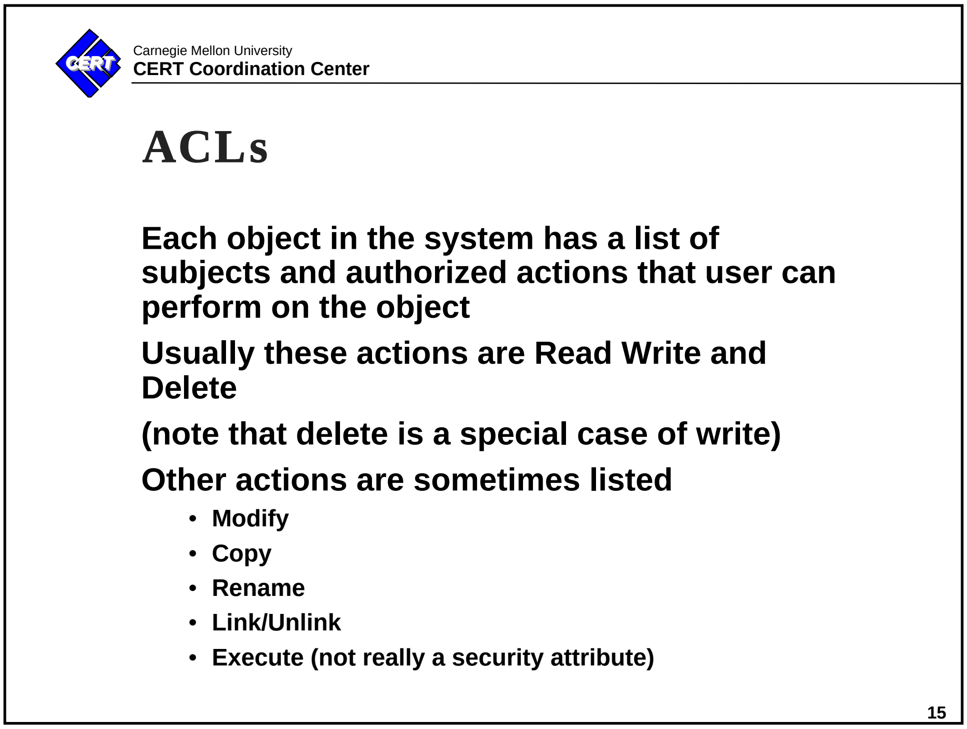
Access Control

Two primary types:

- Access control lists
- Capability lists

Not just about files, but includes

- Services
- Resources
- Processes
- Network connections
- Dial-out lines
- etc



ACLs

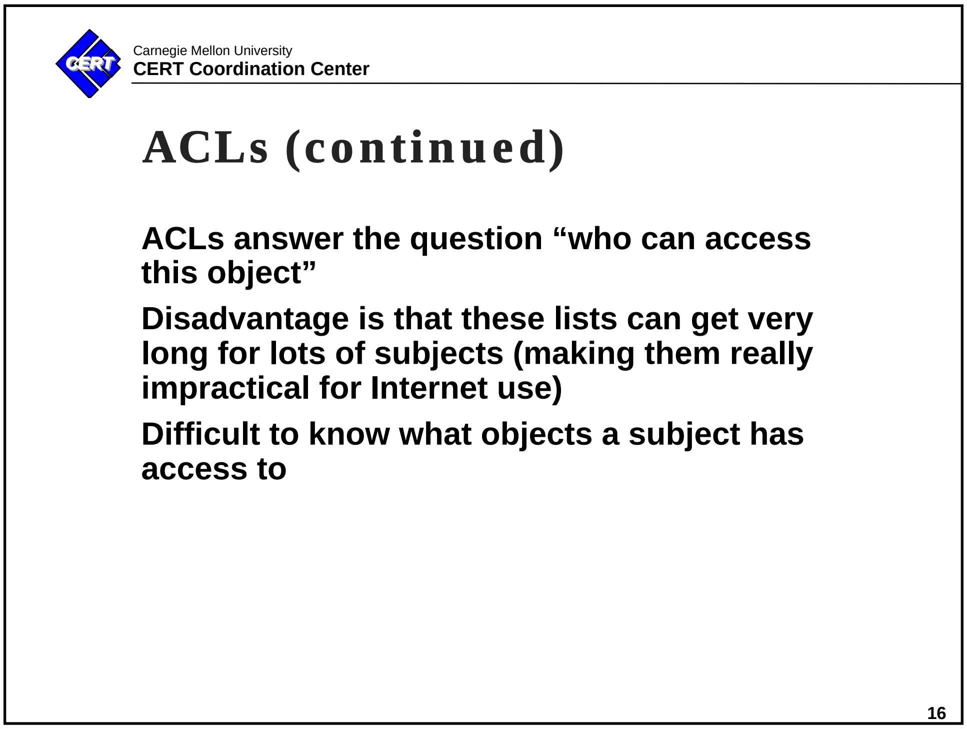
Each object in the system has a list of subjects and authorized actions that user can perform on the object

Usually these actions are Read Write and Delete

(note that delete is a special case of write)

Other actions are sometimes listed

- Modify
- Copy
- Rename
- Link/Unlink
- Execute (not really a security attribute)

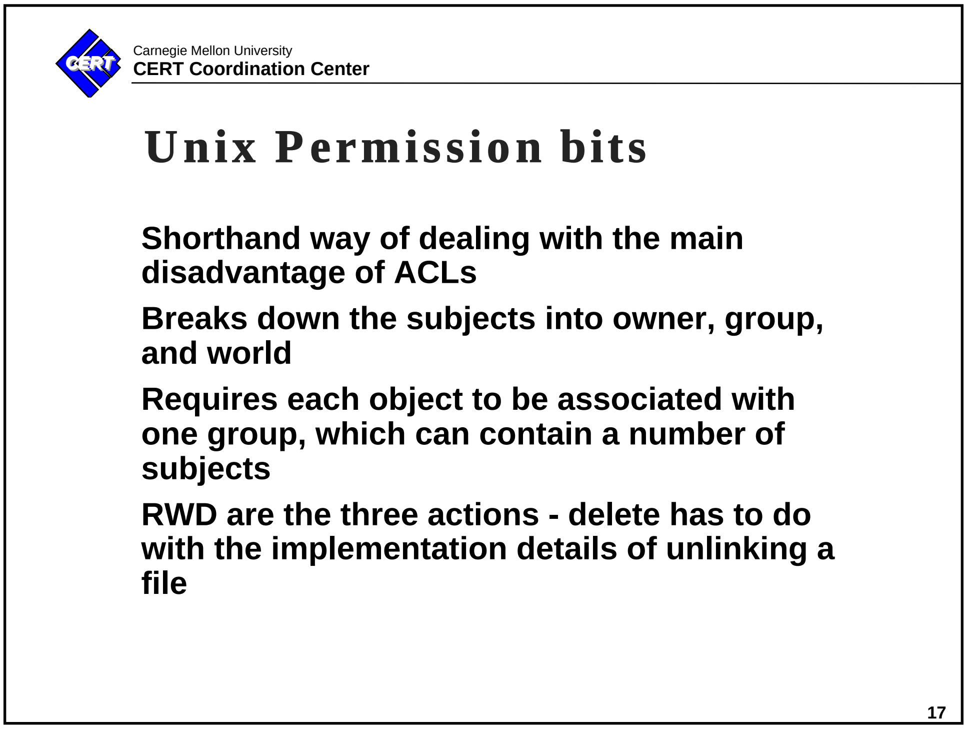


ACLs (continued)

ACLs answer the question “who can access this object”

Disadvantage is that these lists can get very long for lots of subjects (making them really impractical for Internet use)

Difficult to know what objects a subject has access to



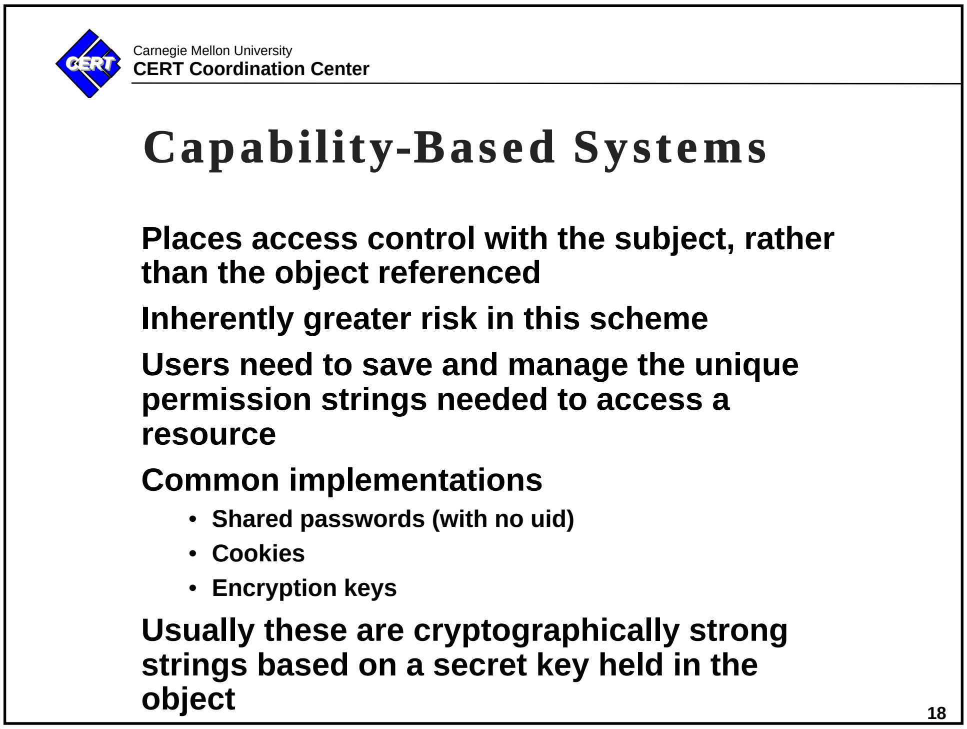
Unix Permission bits

Shorthand way of dealing with the main disadvantage of ACLs

Breaks down the subjects into owner, group, and world

Requires each object to be associated with one group, which can contain a number of subjects

RWD are the three actions - delete has to do with the implementation details of unlinking a file



Capability-Based Systems

Places access control with the subject, rather than the object referenced

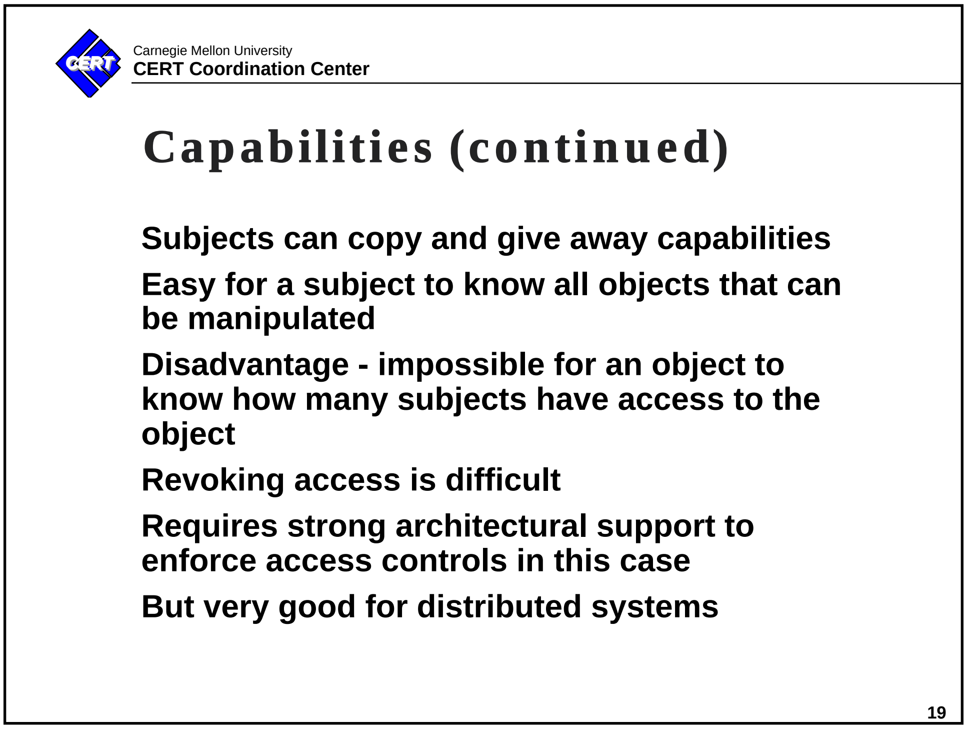
Inherently greater risk in this scheme

Users need to save and manage the unique permission strings needed to access a resource

Common implementations

- Shared passwords (with no uid)
- Cookies
- Encryption keys

Usually these are cryptographically strong strings based on a secret key held in the object



Capabilities (continued)

Subjects can copy and give away capabilities

Easy for a subject to know all objects that can be manipulated

Disadvantage - impossible for an object to know how many subjects have access to the object

Revoking access is difficult

Requires strong architectural support to enforce access controls in this case

But very good for distributed systems