

## CSI 34a: Security

- Overall outline
  - Basic Issues
  - Types of protection and security
  - Penetration of a computing facility
  - Access and information-flow control mechanisms
  - Protection Problems



---

---

---

---

---

---

---

---

## Protection for secondary storage

- Implicit access through virtual memory (page/seg faults)
- Explicit access through the file system
- Filesystems needs to keep information about access control
  - Access lists
  - Capability lists



---

---

---

---

---

---

---

---

## Access/capability lists

- Access table for each file, listing which subjects can access the file
- Capability for each process/user listing rights
- Unix
  - access list for each file: *user/group/other, rwx*
    - user: *jyh group: cs134 file perm: 4775*
  - capability for each user: *list of groups*
- Windows NT
  - access list for each kernel object
  - capability list for each user: *list of groups*



---

---

---

---

---

---

---

---

## Capabilities vs access lists

- Capability is like a *ticket* (to see *Godzilla vs Mothra*)
- Access list is like a *reservation* at a restaurant
- Are they the same?
  - Same amount of info
  - Capabilities can become the only mechanism for access



---

---

---

---

---

---

---

---

## Capabilities

- Capability list is like a segment table; it points to objects that may be accessed
  - segment tables are only valid for the life of a process
  - segment tables refer to primary storage
  - capabilities refer to secondary storage
- Capability-based addressing
  - combines access models
  - capabilities are the only means to access an object



---

---

---

---

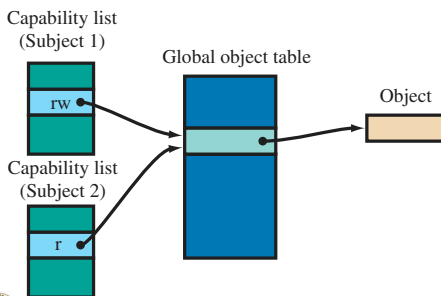
---

---

---

---

## Capability-based addressing



---

---

---

---

---

---

---

---

## Dynamic environments

- Execution environment is determined by segment table
- Segment table may *grow*; since segments are never unlinked, it never *shrinks*
- Dynamic environment needs additional protection
- MULTICS provides a scheme of concentric *rings*
  - Innermost rings are most highly protected
  - Access is only allowed to outer shell



---

---

---

---

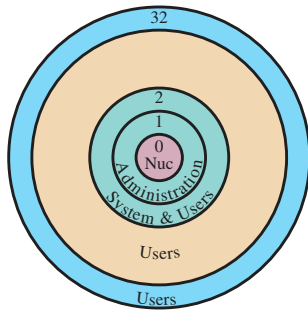
---

---

---

---

## MULTICS rings



---

---

---

---

---

---

---

---

## MULTICS Rings

- An inward call generates an interrupt
  - checks the reference based on access rights
- An outward call also generates an interrupt
  - Arguments must be copied to the outer procedures



---

---

---

---

---

---

---

---

## Rings

- Seems flexible, but requires strict ordering
- In general, objects may have references that form an arbitrary graph, and they may be difficult to partition



---

---

---

---

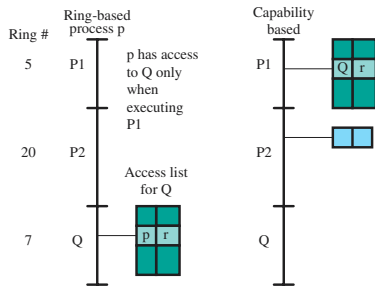
---

---

---

---

## Capability-based systems



---

---

---

---

---

---

---

---

## Privileged system states

- Supervisor/user mode
- VAX: kernel, executive, supervisor, user
- Capability-based has no need for privileged modes



---

---

---

---

---

---

---

---

## Protection problems

- We define schemes for explicit access
- What about information flow?
- A problem when mutually suspicious processes want to cooperate
- Example user interacting with a service



---

---

---

---

---

---

---

---

## Service goals

- No user should be able to steal the service by making a copy
- No user should be able to damage the service
- No user should be able to use the service without permission
- It should be possible to revoke access to the service
- No user should be able to prevent others from using the service



---

---

---

---

---

---

---

---

## User goals

- The service should not be able to steal or destroy any information or services that were not explicitly given to the service
- The service should be able to notify its owner with nonsensitive information (e.g. billing), but not sensitive information



---

---

---

---

---

---

---

---

## Access-control

- Theft and destruction can be solved with execute-only privileges for the service
- Unauthorized use is harder; could use access control
- Capability-based systems
  - *Capabilities can be copied and passed to other users*



---

---

---

---

---

---

---

---

## HYDRA

- Generic rights for an object  $o$ :  $r$ ,  $w$ ,  $s$ , and  $l$ 
  - $r$ : read,  $w$ : write
  - $s$ : right to copy a capability into object  $o$
  - $l$ : right to copy capability from object  $o$
- Capabilities a special right: the *environment* right, that allows them to be copied



---

---

---

---

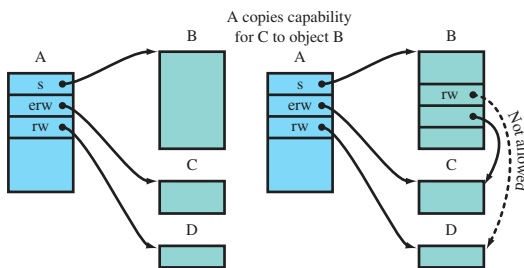
---

---

---

---

## Capability-passing



---

---

---

---

---

---

---

---

## Revocation of privileges

- Access lists: remove a user from the list
  - *What about a file open?*
- Capability-based
  - *Have a dummy pointer to an object alias*
  - *All references are through alias*
  - *All references can be removed by modifying alias*



---

---

---

---

---

---

---

---

## Other problems

- Denial of service
  - *Not possible to enforce in asynchronous systems*
  - *Can give each process a fixed time limit...*
- Trojan horse (service destroys or discloses information)
  - *Have the "gift" process run with restricted privileges*



---

---

---

---

---

---

---

---