

## Outline for October 22, 2025

**Reading:** *text*, §16.2–16.4, 13

**Due:** Homework 2, due October 22; Project selection, due November 7

---

1. Greetings and Felicitations!
2. Capabilities
  - (a) Capability-based addressing
  - (b) Capabilities as security mechanisms
  - (c) Inheritance of C-Lists
  - (d) Revocation
3. MULTICS ring mechanism
  - (a) Rings, gates, ring-crossing faults
  - (b) Used for both data and procedures; rights are REWA  
( $b_1, b_2$ ) access bracket—can access freely; ( $b_3, b_4$ ) call bracket—can call segment through gate; so if  $a$ 's access bracket is (32, 35) and its call bracket is (35, 39), then assuming permission mode (REWA) allows access, a procedure in:  
rings 0–31: can access  $a$ , but ring-crossing fault occurs  
rings 32–35: can access  $a$ , no ring-crossing fault  
rings 36–39: can access  $a$ , provided a valid gate is used as an entry point  
rings 40–63: cannot access  $a$
  - (c) If the procedure is accessing a data segment  $d$ , no call bracket allowed; given the above, assuming permission mode (REWA) allows access, a procedure in:  
rings 0–32: can access  $d$   
rings 33–35: can access  $d$ , but cannot write to it (W or A)  
rings 36–63: cannot access  $d$
4. Lock and Key
  - (a) Associate with each object a lock; associate with each process that has access to object a key (it's a cross between ACLs and C-Lists)
  - (b) Example: cryptographic (Gifford).  $X$  object enciphered with key  $K$ . Associate an opener  $R$  with  $X$ . Then:  
**OR-Access:**  $K$  can be recovered with any  $D_i$  in a list of  $n$  deciphering transformations, so  $R = (E_1(K), E_2(K), \dots, E_n(K))$  and any process with access to any of the  $D_i$ 's can access the file  
**AND-Access:** need all  $n$  deciphering functions to get  $K$ :  $R = E_1(E_2(\dots E_n(K) \dots))$
5. Types and locks
6. Authentication
  - (a) Validating client (user) identity
  - (b) Validating server (system) identity
  - (c) Validating both (mutual authentication)
  - (d) Basis: what you know/have/are, where you are
7. Passwords
  - (a) Problem: common passwords, easy to guess passwords
  - (b) Best: use passphrases: goal is to make search space as large as possible, distribution as uniform as possible
8. Attacks
  - (a) Exhaustive search
  - (b) Guessing
  - (c) Scavenging: passwords often typed where they might be recorded as login name, in other contexts, etc.

(d) Ask the user: very common with some public access services

9. Defenses

(a) For trial and error at login: dropping or back-off

(b) For thwarting dictionary attacks: salting

10. Challenge-response systems

(a) Computer issues challenge, user presents response to verify secret information known/item possessed

(b) Example operations:  $f(x) = x + 1$ , random, string (for users without computers), time of day, computer sends  $E(x)$ , you answer  $E(D(E(x)) + 1)$

(c) Note: password never sent over network

11. One-Time Password

(a) Password is valid for only one use

(b) May work from list, or new password may be generated from old by a function or a hardware token

12. Biometrics

(a) Depend on physical characteristics

(b) Examples: pattern of typing (remarkably effective), retinal scans, etc.

13. Location

(a) Bind user to some location detection device (human, GPS)

(b) Authenticate by location of the device

14. Multi-factor authentication