

# CS 4803

## Computer and Network Security

Alexandra (Sasha) Boldyreva  
More on access control.  
General design principles

1

### Access to general objects

- Memory protection is only one example
- Need a way to protect more general objects
- Before we begin, some design principles...

2

### Overview

- Seminal article by Saltzer and Schroeder (1975)
  - Eight principles underlying design and implementation of security mechanisms
- Linked from the course homepage

3

### Key point I

- Simplicity
  - Make designs/mechanisms easy to understand
  - Less chance of error

4

## Key point II

- Restriction
  - Minimize the “power” of an entity
    - E.g., only allow access to information it needs
    - E.g., only allow necessary communication; restrict type of communication allowed
  - Less chance of harm!

5

## Principle 1

- “Principle of least privilege”
  - A subject should be given only the privileges it needs to accomplish its task
  - If a subject needs certain privileges only to complete a specific task, it should relinquish those privileges upon completion of the task

6

## In practice...

- Systems are often not designed with the necessary granularity
  - E.g., “append” may not be distinct from “write”
  - E.g., in UNIX, no access controls for *root*
    - Anyone who can make backup files can also delete those files

7

## Principle 2

- “Principle of Fail-Safe Defaults”
  - Unless a subject is given explicit access to an object, it should be denied access
    - I.e., the default is no access
  - E.g., a process reporting an error message should not try to expand its rights in an attempt to correct the error

8

### Principle 3

- “Economy of Mechanism”
  - Security mechanisms should be as simple as possible...
  - ...but no simpler!
  - Can simplify formal proofs of security (or even informal audits)

9

### Consequences

- If design/implementation are simple, less chance for error
- Software testing is also simpler
- Software interfaces especially suspect
  - Typically make assumptions about the input/output format of the other module
  - E.g., finger protocol: DoS attack by returning infinite stream of characters

10

### Principle 4

- “Principle of Complete Mediation”
  - All accesses to objects should be checked to ensure they are allowed
  - OS should mediate any request to read an object --- even on the second such request by the same subject!
    - Don’t cache authorization results

11

### Insecure example...

- In UNIX, when a process tries to read a file, the system checks access rights
- If allowed, it gives the process a file descriptor
- File descriptor is presented to OS for access
- If permissions are subsequently revoked, the process still has a valid file descriptor!
  - Insufficient mediation

12

## Principle 5

- “Open Design”
  - No “security through obscurity”
  - Security of a system should not depend on the secrecy of its implementation
    - Of course, secret *keys* do not violate this principle!

13

## Principle 6

- “Separation of Privilege”
  - (As much as is feasible...) a system should not grant permission based on a single condition
  - E.g., require more than one sys admin to issue a critical command, or more than one teller to issue an ATM card

14

## Principle 7

- “Principle of Least Common Mechanism”
  - Minimize mechanisms depended upon by all users
  - Shared mechanisms are a potential information path, and should not compromise security
  - Also expose the system to potential DoS attacks

15

## Principle 8

- “Psychological Acceptability”
  - Security mechanisms should not make access to the resource more difficult
  - If mechanisms are too cumbersome, they will be circumvented!
  - Even if they are used, they may be used incorrectly

16

## Back to specifics...

- File protection as the running example
  - But everything said here is more generally applicable

17

## Access control matrix

- One central matrix indexed by all subjects and objects
  - Characterizes rights of each subject with respect to each object
- Formally: set of objects  $O$  and subjects  $S$
- Matrix  $A$  contains an entry for every pair  $(S, O)$ 
  - The entry contains the rights for  $S$  on  $O$
  - Examples: read/write/execute/etc.

18

## More complex access control

- In general, "rights" may be functions
  - "Actual" rights depend on the system state
  - Equivalently, may depend on system history

19

## Drawbacks...

- Number of subjects/objects is very large
- Most entries blank/default
- One central matrix is modified every time subjects/objects are created/deleted or rights are modified

20

## Access control lists (ACLs)

- Can be viewed as storing the rows of the access control matrix with the appropriate object
- One list per object, showing all subjects with access and their rights
- Possible to assign “default rights” to an object
  - Easy to make an object public
- Example: access based on user, group, and compartment
  - Use of wildcards

21

## Some design decisions

- . How fine-grained to allow ACLs?
  - . E.g., user-level, group-level, or only public/private?
  - . Granularity of rights (e.g., “append”?)
- . How to handle conflicts if two subjects give different permissions on an object
  - . Disallow multiple owners
  - . Allow access if any entry gives rights
  - . Allow access only if no entry denies rights
  - . Apply first applicable entry
  - . Revocation?

22

## Capabilities

- Some burden for implementing protection placed on the user rather than just the OS
  - Analogy: user has a “ticket” which grants access to an object
  - A capability is an unforgeable token giving user access to an object and describing the level of allowable access
  - Object owners can specify new types of rights

23

## Two general approaches

- Ticket is held by OS, which returns to the subject a pointer to the ticket
- Ticket is held by the user, but protected from forgery by cryptographic mechanisms
  - How...?
  - **Not encrypted** as mistakenly claimed in book!
- Two possibilities: ticket verified by the object or by the OS itself
  - Who holds the key in each case...?

24

## Drawback

- Does not really satisfy principle of complete mediation
  - Can add automatic expiration to mitigate this