CISC 3320

C30c Role-based and Mandatory Access Control and Others

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Acknowledgement

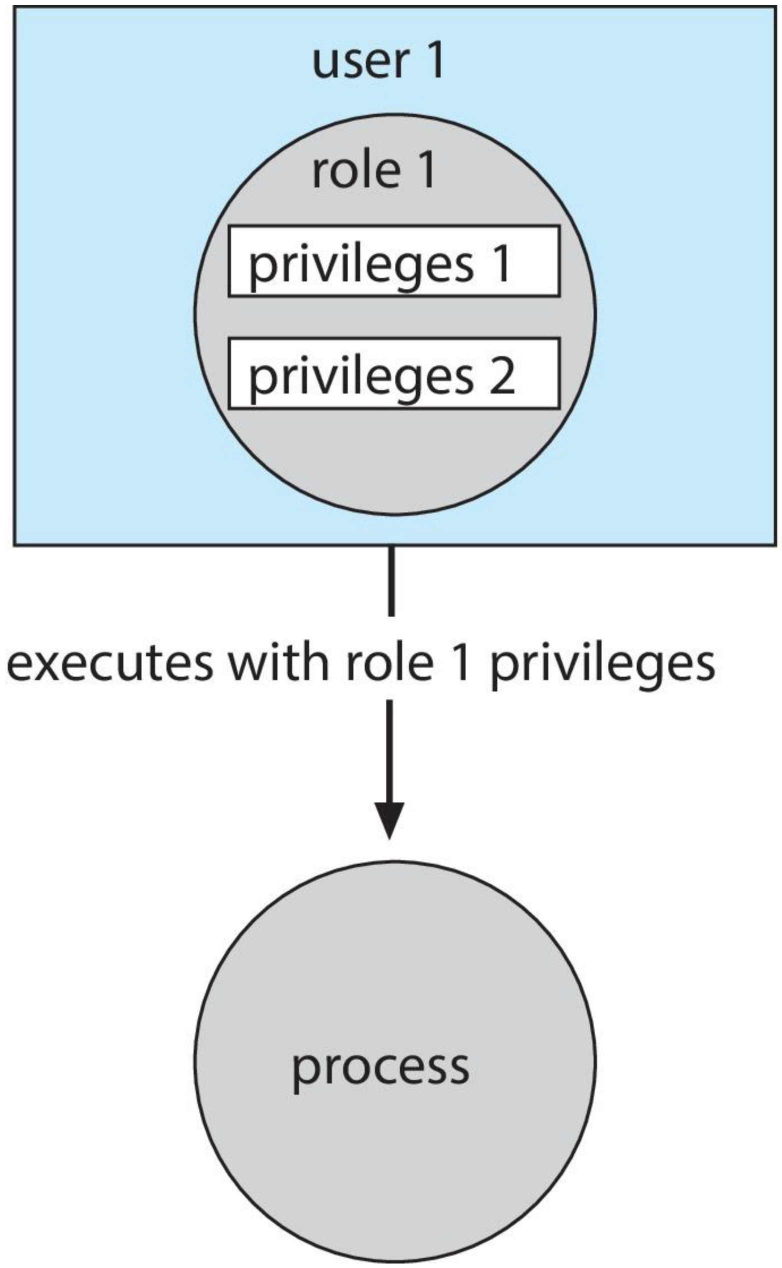
- These slides are a revision of the slides provided by the authors of the textbook via the publisher of the textbook

Outline

- Role-based Access Control
- Mandatory Access Control (MAC)
- Capability-Based Systems
- Other Protection Implementation Methods
- Language-based Protection

Role-based Access Control

- Protection can be applied to non-file resources
- Oracle Solaris 10 provides **role-based access control** (**RBAC**) to implement least privilege
    - ***Privilege*** is right to execute system call or use an option within a system call
    - Can be assigned to processes
    - Users assigned ***roles*** granting access to privileges and programs
        - Enable role via password to gain its privileges
    - Similar to access matrix


user 1
role 1
privileges 1
privileges 2
executes with role 1 privileges
process

Discretionary Access Control (DAC)

- Individual user sets access control mechanism to allow or deny access to an object
  - Operating systems traditionally had discretionary access control (DAC) to limit access to files and other objects
  - *e.g.*, UNIX file permissions and Windows access control lists (ACLs)
- Discretionary is a weakness
  - Because users / admins need to do something to increase protection

Mandatory Access Control (MAC)

- System mechanism controls access to object, and individual cannot alter that access
- Sometimes called *rule-based access control*
- Stronger form is mandatory access control, which even root user can't circumvent
  - Makes resources inaccessible except to their intended owners
- Modern systems implement both MAC and DAC
  - Usually MAC acts as a more secure, optional configuration
  - e.g., Trusted Solaris, TrustedBSD (used in macOS), SELinux, Windows MAC

MAC and Labels

- At its heart, labels assigned to objects and subjects (including processes)
- When a subject requests access to an object, policy checked to determine whether or not a given label-holding subject is allowed to perform the action on the object

Capability-Based Systems

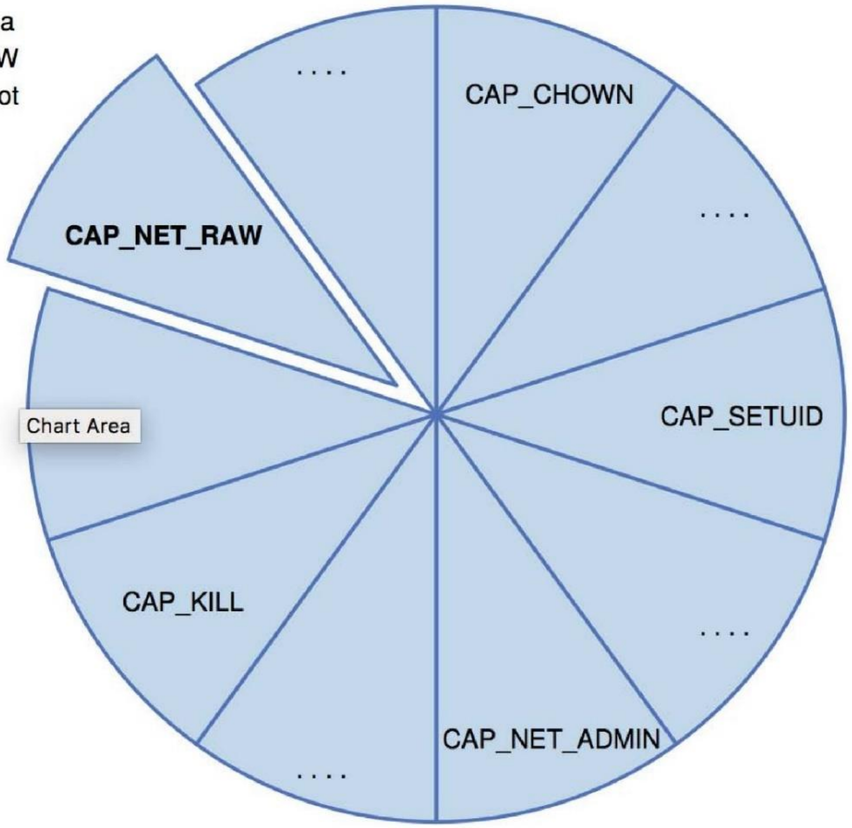
- Hydra and CAP were first capability-based systems
- Now included in Linux, Android and others, based on POSIX.1e (that never became a standard)
  - Essentially slices up root powers into distinct areas, each represented by a bitmap bit
  - Fine grain control over privileged operations can be achieved by setting or masking the bitmap
  - Three sets of bitmaps – permitted, effective, and inheritable
    - Can apply per process or per thread
    - Once revoked, cannot be reacquired
    - Process or thread starts with all privs, voluntarily decreases set during execution
    - Essentially a direct implementation of the principle of least privilege
- An improvement over root having all privileges but inflexible (adding new privilege difficult, etc)

Capabilities in POSIX.1e

In the old model, even a simple `ping` utility would have required root privileges, because it opens a raw (ICMP) network socket

Capabilities can be thought of as "slicing up the powers of root" so that individual applications can "cut and choose" only those privileges they actually require

With capabilities, ping can run as a normal user, with CAP_NET_RAW set, allowing it to use ICMP but not other extra privileges



Other Protection Improvement Methods

- System integrity protection (SIP)
- System-call filtering
- Sandboxing
- Code signing
- Language-Based Protection

System Integrity Protection (SIP)

- Introduced by Apple in macOS 10.11
- Restricts access to system files and resources, even by root
- Uses extended file attribs to mark a binary to restrict changes, disable debugging and scrutinizing
- Also, only code-signed kernel extensions allowed and configurably only code-signed apps

System-Call Filtering

- Like a firewall, for system calls
- Can also be deeper –inspecting all system call arguments
- Linux implements via SECCOMP-BPF (Berkeley packet filtering)

Sandboxing

- Running process in limited environment
- Impose set of irremovable restrictions early in startup of process (before `main()`)
- Process then unable to access any resources beyond its allowed set
- Java and .net implement at a virtual machine level
- Other systems use MAC to implement
- Apple was an early adopter, from macOS 10.5's "seatbelt" feature
    - Dynamic profiles written in the Scheme language, managing system calls even at the argument level
    - Apple now does SIP, a system-wide platform profile

Code Signing

- Code signing allows a system to trust a program or script by using crypto hash to have the developer sign the executable
  - So code as it was compiled by the author
  - If the code is changed, signature invalid and (some) systems disable execution
  - Can also be used to disable old programs by the operating system vendor (such as Apple) cosigning apps, and then invaliding those signatures so the code will no longer run

Language-based Protection

- Specification of protection in a programming language allows the high-level description of policies for the allocation and use of resources
- Language implementation can provide software for protection enforcement when automatic hardware-supported checking is unavailable
- Interpret protection specifications to generate calls on whatever protection system is provided by the hardware and the operating system
- Example
  - Java

Protection in Java 2

- Protection is handled by the Java Virtual Machine (JVM)
- A **class** is assigned a protection domain when it is loaded by the JVM
- The protection domain indicates what operations the class can (and cannot) perform
- If a library **method** is invoked that performs a privileged operation, the stack is **inspected** to ensure the operation can be performed by the library
- Generally, Java's load-time and run-time checks enforce **type safety**
- Classes effectively **encapsulate** and protect data and methods from other classes

Questions?

- Role-based Access Control
- Mandatory Access Control (MAC)
- Capability-Based Systems
- Other Protection Implementation Methods
- Language-based Protection