50004 - Operating Systems - Lecture 17

Oliver Killane

27/12/21

Lecture Recording

Lecture recording is available here

File Access on UNIX/Linux continued...

Capability Lists

Effectively lists of the rows of an access matrix, associated with the principals (users). If a capability is in the list, then the user associated with it has the capability to perform said action on said object.

- Capabilities are protected pointers to objects, which specify the permitted operations (e.g file descriptor, index of capability list).
- Often implemented as protected pointers to objects. The kernel holds the capability information (can create, destroy and modify it), providing access to the user indirectly (e.g user has a file descriptor, or a key of sorts).
- Alternatively the capability may be held in user-memory, but encrypted.

Access Control Lists vs Capabilities

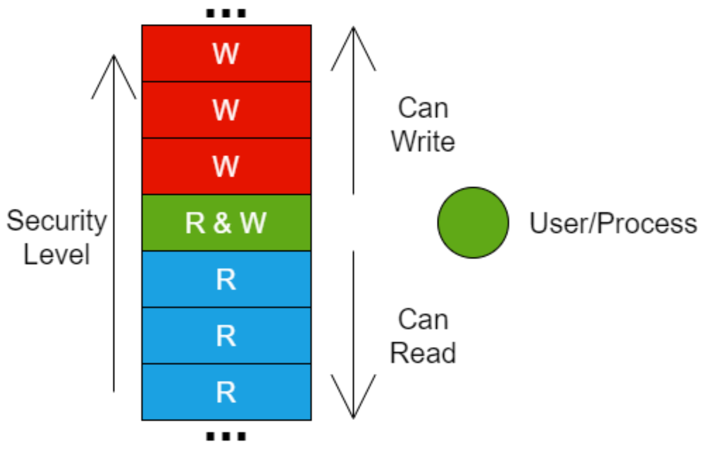
Action	**Best**	**Access Control Lists**	**Capabilities**
Principle of Least Privilege	Cap	Use of groups mean all users in a group get access rights.	Can be very fine grained, only giving capabilities to specific users.
Revocation	ACL	Can revoke all access by changing one ACL for the object.	Must search all capability lists, and remove capability.
Rights Transfer	Cap		Can transfer a single capability from one user to another.
Persistence	ACL	File systems are designed to be persistent.	Very complex.

DAC vs MAC

- **Discretionary Access Control (DAC)** Principals determine who may access their objects. This is the default in UNIX/Linux, file owners determine the access rights.
- **Mandatory Access Control (DAC)** Precise system rules determine object access. A policy determines access controls for files, devices etc.

Bell - La Padula Model

A **MAC** policy where information cannot travel down the security hierarchy.



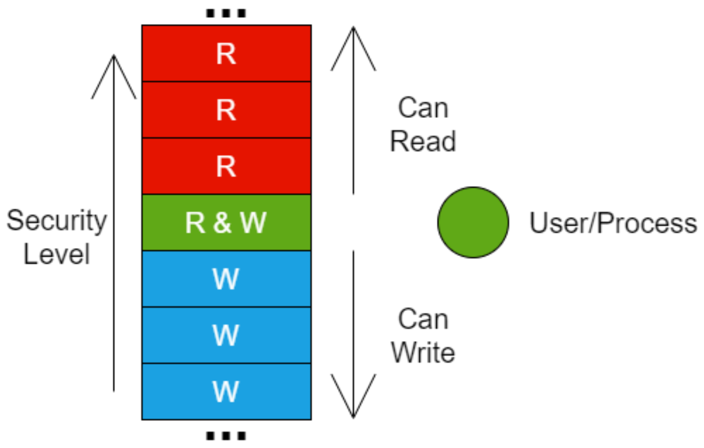
- Objects and principlals are assigned a security level (e.g unclassified, confidential, top secret)
- **Two Rules**

Simple Security Policy	Processes can only read objects at its security level or lower.
The * property	Processes can only write to objects at their security level or higher.

Information can only travel up the security levels. This ensures confidentiality, but not integrity.

Biba Model

A **MAC** policy guaranteeing data integrity.



- **Simple Integrity Principle** Can only write to objects of the same or lower security level.
- **Integrity * Property** Can only read objects of the same or higher security level.

Design Principles

- **Least Privelege** Each process should run with the lowest privelege possible. Default should be no access.

- **Simple & Uniform Mechanism** ensures design is easy to reason about, and can be applied to many types of principals & objects.
- **Psychologically Acceptable** If policies are too much of a burden, too complex, or difficult to deal with then people will not implement them.
- **Public** Security by obscurity is bad, by making the design public it can be critiqued for flaws.

Virtualisation

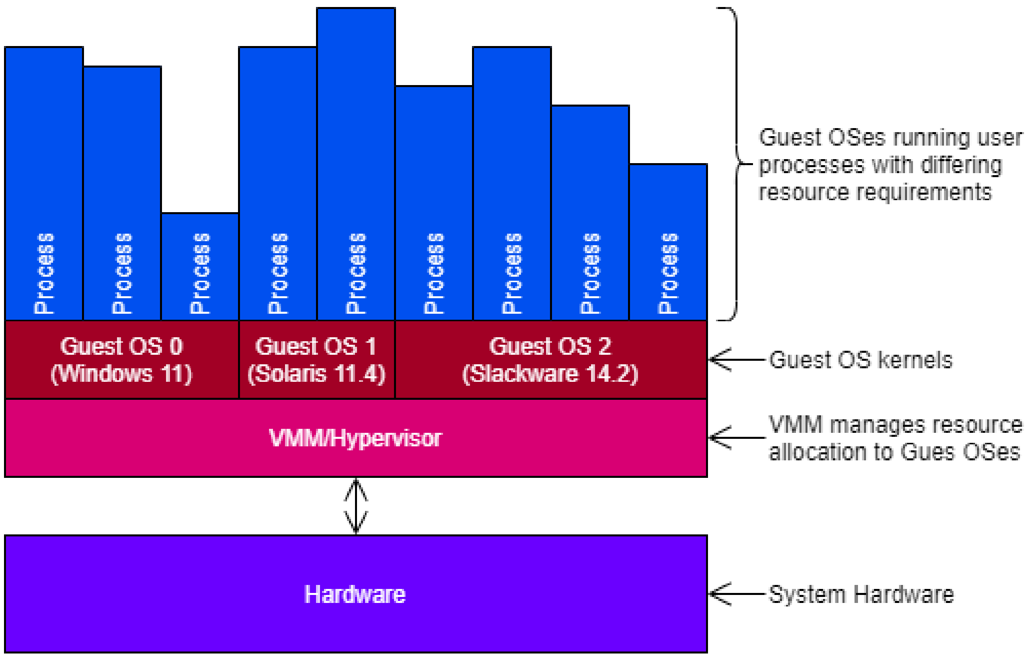
Operating systems abstract away hardware, providing abstractions such as virtual memory, signals, file systems, syscalls and more.

For some systems we may want to abstract hardware, providing virtualised CPUs, disks, interrupts, physical memory etc as a virtual machine. such that an entire operating system can run inside.

This is useful for:

- **Legacy software** Can run software designed for much older systems.
- **Security** Even if virtual machine compromised, the system has not. This is especially useful when deliberately compromising a machine to reverse engineer malware, virtual machine acts as a secure sandbox in which to experiment.
- **Software Management** avoids issues with conflicting software (versions and dependencies) by running them in separate virtual environments.

Top achieve this we use a **Virtual Machine Monitor** (**VMM**) or **Hypervisor** to partition resources, and provide these abstractions as a virtual machine on which guest Operating Systems can run.


Process
Process
Process
Process
Process
Process
Process
Process
Process
Guest OSes running user processes with differing resource requirements
Guest OS 0 (Windows 11)
Guest OS 1 (Solaris 11.4)
Guest OS 2 (Slackware 14.2)
Guest OS kernels
VMM/Hypervisor
VMM manages resource allocation to Gues OSes
Hardware
System Hardware