Operating Systems Basics

Tony Espinoza

am.espinoza@utexas.edu

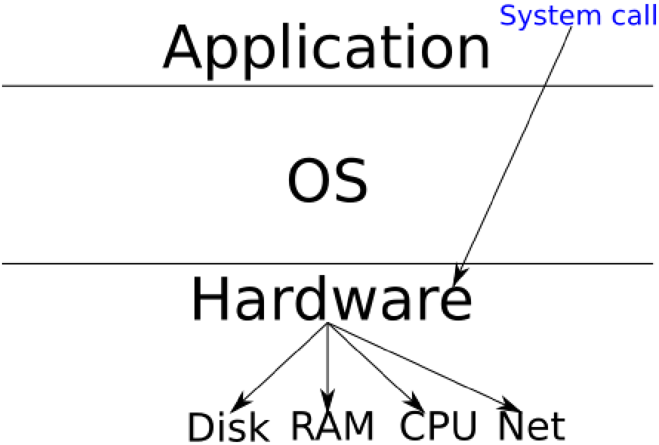
Operating System

- System software that manages computer hardware, software resources, and provides common services for computer programs.
  - I/O
  - Memory allocation. . .

Operating Systems

- How do we interact with it (on the programming level)?
  - System calls.
- What is a system call?
  - The way for an application to interact with the hardware.
  - The way for an application to interact with privileged applications/data structures.

Operating System



Network system calls

- socket (domain, type, protocol)
- accept (socket, address, address length)
- bind (socket, address, address length)
- listen (socket, backlog)

File I/O system calls

- read (file descriptor, buffer, size)
  - Used to read a file.
- write (file descriptor, buffer, size)
  - Used to write a file.
- open (path name, flags)
  - Returns the file descriptor of the file pointed to by path name..
- close (file descriptor)
  - Closes a file descriptor.

CPU/Process system calls

- execve (pathname, argv, envp))
  - Executes program referenced by pathname, args are arguments, envp are the environment variables.
- fork ()
  - No arguments, creates a new process. Return value is 0 in the child and the process identification number of the child in the parent.
- clone ()
  - Variable arguments, creates a new process and can share parts of its context with the parent process.

Memory system calls

- brk (address)
  - Sets the end of the data segment to the value specified by address.
- sbrk (increment)
  - Increments the program's data space by increment bytes.
- mmap (addr, length, prot, flags, fd, offset)
  - Map files or devices into memory.

Low level (x86)

- read (file descriptor, buffer, size)
- rax = 0 , rdi=file descriptor, rsi=buffer, rdx=size.
- int 80, syscall
- Linux syscall table.

strace

- Traces system calls and signals.
- strace
  - ls
  - echo this
  - etc.
- strace arguments
  - -e trace=network
  - -e trace=memory
  - -c

System calls

- How would I look up what a system call does?
  - check the manual for it.
  - ex `man 2 sbrk`.
  - man 2 if for Linux system calls.
- Side Note:
  - manual (man) pages for Linux are similar to RFCs for the Internet.
  - `man man`

Who uses the OS?

- Users
  - Users can own files
    - Permission can be set to files.
  - Users can be part of groups.
  - Groups have permissions to read and write files.

UID

- Users all have a unique number.
  - Their unique identification number (UID)
  - The UID is associated with all of a users processes.
  - See your UID by typing `id -u`.
  - See all UIDs, `cat /etc/passwd`.

/etc/passwd

How to read `/etc/passwd` file.

- Username
- Password. An x character indicates that encrypted password is stored in /etc/shadow file.
- User ID (UID).
- Group ID (GID).
- User ID Info.
- Home directory.
- Command/shell.

/etc/shadow

- A file containing all hashed passwords for the system.
- Passwords are typically salted before being hashed.
  - Salting is used to make unique hashes and avoid precomputed attacks.
  - Typically the salted value is concatenated to the password before hashing.
- Look at precomputed attacks.

MD5

- A hashing algorithm.
- Is hard to figure out what the original source text was, but we can pre compute values.
  - Hashtable where key = hash, value = plaintext.
  - lookup(9dbb300e28bc21c8dab41b01883918eb) = “passwordpassword”
- From the command line type:
  - `echo -n "passwordpassword" | md5sum`
- Can test hash at https://md5.gromweb.com/.

EUIDs

- A file can have an effective user ID (EUID)
- The EUIDs allow for an unprivileged process to run with the privileges of the file.
  - Useful for files like /etc/passwd
  - `ls -alhs /bin/passwd`

Users and groups

- `ls -alhs`
  - Change permission with `chmod`
  - Add a user to a group `usermod -aG additional_groups username`
  - To view all groups `cat /etc/group`
  - change ownership on a file?
    - `chown`

Processes

- What is a process?
  - An instance of a specific running program.
- How do you refer to a specific process?
  - process ID or PID.
- How to look up a PID?
  - Use `ps`.
  - ex. `ps aux | grep firefox`.

Uses of the PID

- Kill a misbehaving program, `kill -9 PID`.
- Attach to a program with a debugger.
- Investigate the program with `strace -p PID`.

Block devices

- Devices that are read in chunks or blocks.
  - Hard drive
  - Flash drive
  - DVD
  - Card reader

Block devices

- How to display them?
  - `lsblk` list block devices.
- How to read them?
  - Mount them to the file system.
  - `mount /dev/sdb1 /mnt`
  - `umount /mnt`

Block devices

- Let's mount an ISO image to /mnt
- ISO image is a disk image of an optical disk

Recap

- OS controls system resources including hardware.
- Systems calls are the mechanisms which user space applications use to interact with the OS.
- PID is a unique process ID.
- UID is a unique user ID.
- Block devices can be mounted and read.

File structure

- / - called slash, the root directory.
- `/boot` - static files for the boot loader.
- `/home` - user directories.
- `/etc` - configuration files
- `/dev` - device files, HD, disk, etc.
- `/proc` - not actually on the disk.

/proc

- The proc filesystem is a pseudo-filesystem which provides an interface to kernel data structures.
- In your VM navigate to `/proc`
- What do you see?
- `man 5 proc`

/proc

- `cd /proc/sys/net/ipv4`
- Here you will see
  - `tcp_syncookies`
  - `tcp_max_syn_backlog`
- Can manipulate files here directly
  - `sudo tee tcp_syncookies <<< 0`

Containers

- Containers are a way to provide isolation.
  - chroot
  - cgroups
  - namespaces

chroot

- Chroot is a way to isolate a directory.
  - Makes the chrooted directory the root directory.
- Can not access anything not contained in the directory.
  - No ls,bash,vim . . .
- Copy and run `makebox.sh` from canvas.
- `sudo chroot $HOME/box /bin/bash`

Namespaces

- `man 7 namespaces`
- Namespaces provide a way to isolate.
  - Enables a process to have a different view of the system than other processes.
- There are 7 namespaces.

Namespaces

- Cgroup
  - Cgroup root directory
- IPC
  - System V, POSIX interprocess communication.
- Network
  - Network devices, stacks, ports. . .
- Mount
  - Mount points.
- PID
  - Process IDs.
- User
  - User and group IDs.
- UTS
  - Hostname and NIS domain name.

Namespace API

The following system calls are used to interact with namespaces:

- clone
  - Create a new process and if flags are passed create namespaces for the new process.
- setns
  - Join a namespace.
- unshare
  - Moves the calling process into a new namespace
- ioctl
  - Discover information about namespaces

unshare

- Let's use the unshare command to create a new hostname namespace
- Open 2 terminals
  - In 1 type `uname -n`
  - In the other
    - `sudo unshare -u /bin/bash`
    - `hostname bob`
    - `uname -n`

PID

- Processes are one big tree each with a parent process, and possibly children.
- What happens if we isolate the PID?
  - It will think it's the parent process.
  - It may not have any children.
- `sudo unshare --fork --pid --mount-proc.`
  - Run `top`.
- In another termina run `top`.

mount

- Do namespace mount example.
- `man user_namespaces`

cgroups

- cgroups is short for Control Groups.
- Developed in 2006 by 2 google engineers
- In 2008 it was added to the Linux kernel 2.6.24
- Used by many container projects, Docker, LXC . . .

cgroups

- Resource limiting
  - Groups can be set to not exceed a configured memory limit.
- Prioritization
  - Some groups may get a larger share of CPU utilization or disk I/O throughput.
- Accounting
  - Measures a group's resource usage.
- Control
  - Freezing groups of processes, their checkpointing and restarting.

cgroups

- Let's create cgroups
- Limit a program's memory usage.
- Limit a program's hard drive usage.

chroot, cgroups, namespaces

- Containers
- chroot restricts access to the filesystem.
- cgroups restricts access to the system resources.
- namespaces provide isolation.

Access control

- Access control determines how subjects have control over objects.
  - Subjects are users.
  - Objects are files/programs.
- Can think of as a matrix that describes how subjects and objects are related

Access control

- Addresses two important topics.
    - Confidentiality
        - Consealment of resources/information.
        - i.e. don’t leak secrets.
    - Integrity
        - Trustworthiness of resource.
        - i.e. has the data been altered.

Access control

- Discretionary Access Control.
  - Linux standard.
- Mandatory Access Control.
  - SELinux.
- Role Based Access Control.
  - Lab2 part 2.

Access control

- Subjects can be users, and objects can be files, processes...

	File 1	File2
User1	0	1
User2	1	0

Access control

- A simple binary representation is limiting.
- Can get more granular.

	File 1	File2
User1	rwx	r
User2	x	rwx

Discretionary access control.

- Base access rights on the identify of the subject and the identify of the object.
- Subjects can determine how other subjects can use(modify,view,execute) files they own at their discretion.
- Linux mode.

Mandatory Access Control (MAC)

- Access control is delegated by an administrator.
- Subjects do not have control over their row of the matrix.
- Owner of an object can not change access control of that object.

Role Based Access Control (RBAC)

- Subjects have roles and those roles have permissions associated with them.
- e.g.?
  - Someone with the role of Student may have access to the campus library.
  - Someone with the role of Teacher, can assign grades on Canvas.
- The permission is bound to the role the user has not the user itself.

SELinux

- Is a way to make Linux perform MAC.
- It is part of Linux as a Security Module (LSM).
- Has been in Android for years.

LSM

- Not integrated into the kernel.
- Provides hooks that happen before and after syscalls.
- Allows for security mechanisms to be implemented at the hooks.

SELinux

- Has three modes
  - Enforcing
    - Denies access based on rules.
  - Permissive
    - Logs access based on the rules but does not deny.
  - Disabled
    - Self explanitory.

SELinux

Rules in SELinux can be though of as "Subject x is allowed to do *access* on *object*".

- Subjects
  - Processes, and transitively users.
- Accesses
  - Read, write, execute.
- Objects
  - Resource on which an action applies.

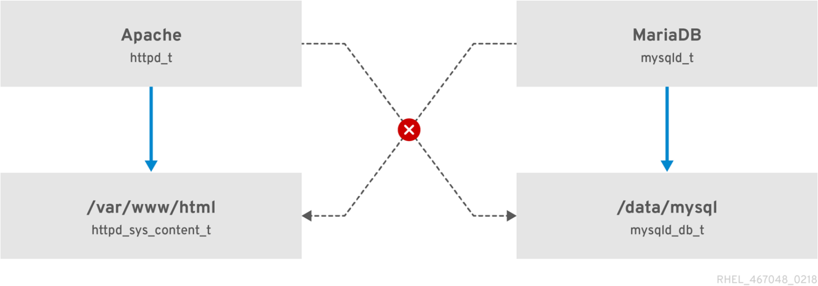
SELinux

- Context
  - Every process or resource has a context associated with it.
  - A context contains the: user, role, type, and security level.
  - Type is the most important.
  - Types end in `_t`
- Type enforcement

Type enforcement

- Type enforcement is implemented based on the labels of the subjects and objects.
- Processes with the label `user_t` can execute regular files labeled `bin_t`.

Type enforcement



Policy rule

- `allow Source Target:Class Permission;`
- Grant Permission to a process of type `Source` on objects of type `Target` and class `Class`.
- `allow unconfined_t mytype_t:file read ;`
- Allow processes in with type unconfined_t read permission on files of type mytype_t