OSN Tutorial-1

Kernel, Syscalls and Processes

11:50 AM

Agenda

Topics Covered

Kernel

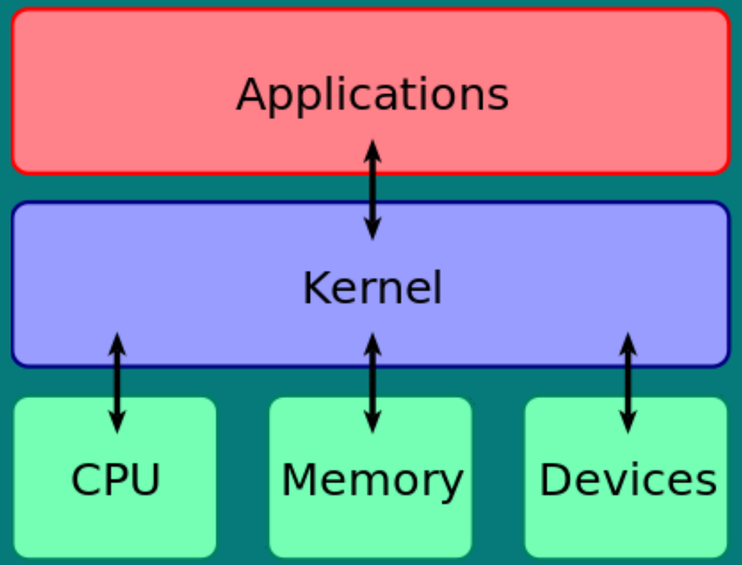
Syscalls

Processes

File Management

Start

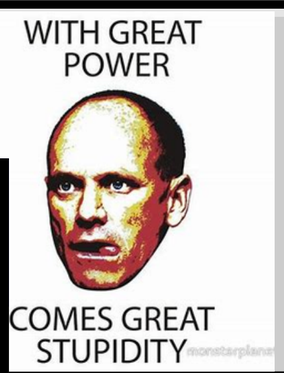
Kernel



The kernel is a computer program at the core of a computer's operating system and generally has complete control over everything in the system. The kernel is also responsible for preventing and mitigating conflicts between different processes. It is the portion of the operating system code that is always resident in memory and facilitates interactions between hardware and software components.

But Why cant I
rm -rf /
without sudo


What if You
Wanted to go to Heaven
But god said
SIGSEGV Segmentation Fault
made with mematic


WITH GREAT POWER
COMES GREAT STUPIDITY

But.....

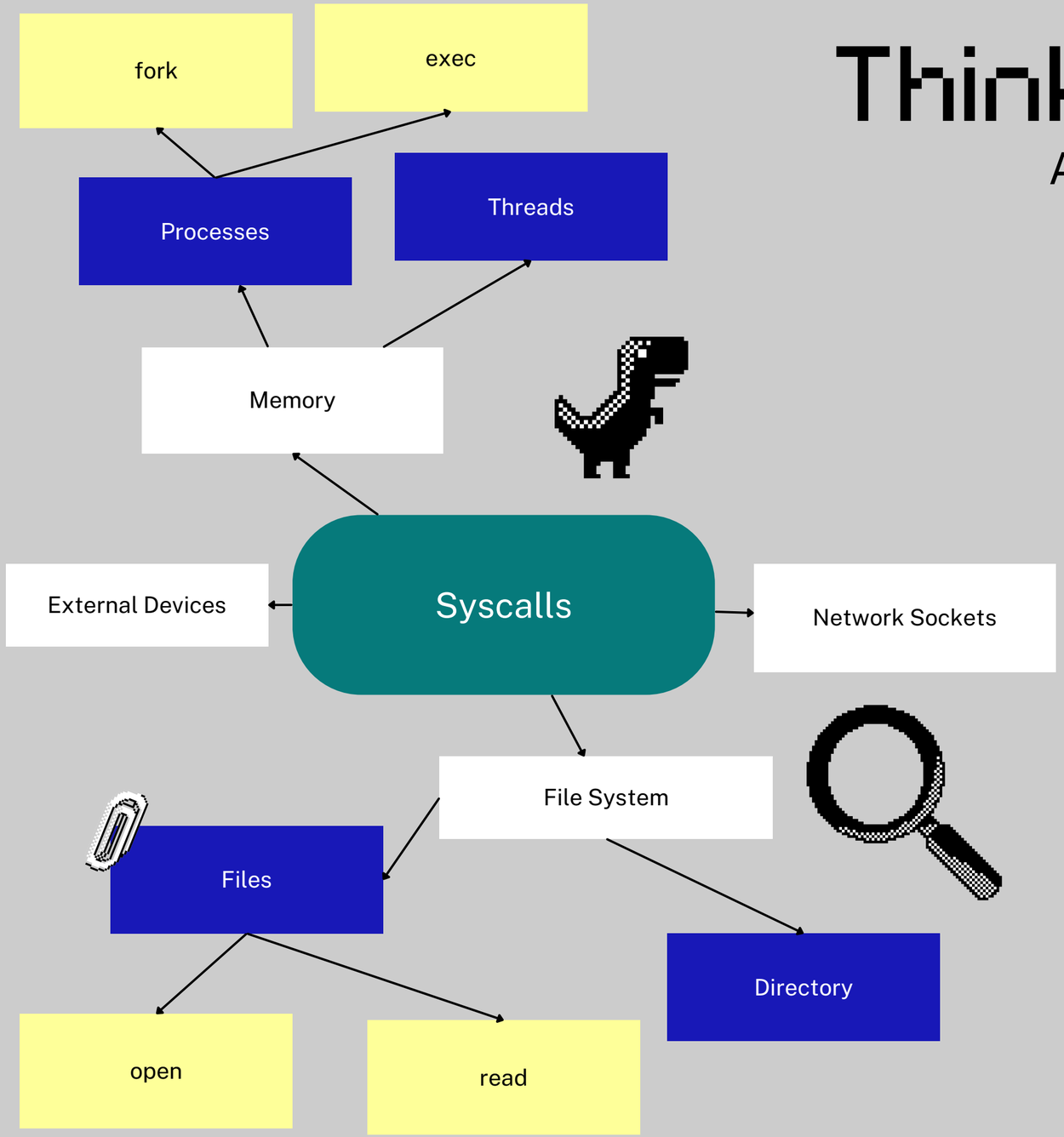
But what if I wanted to open a file,

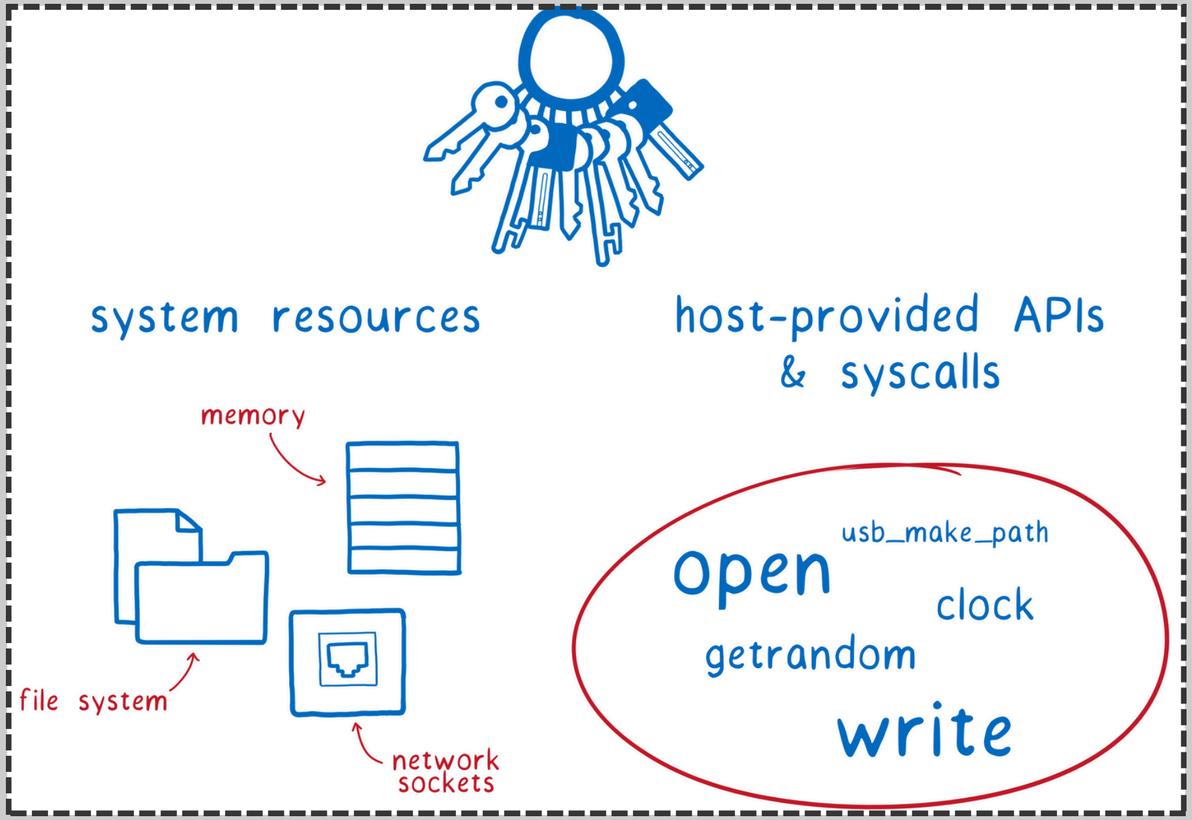
Syscalls

In computing, a system call (commonly abbreviated to syscall) is the programmatic way in which a computer program requests a service from the operating system on which it is executed. This may include hardware-related services (for example, accessing a hard disk drive or accessing the device's camera), creation and execution of new processes, and communication with integral kernel services such as process scheduling. System calls provide an essential interface between a process and the operating system.

Think of Syscall as keys !

And the action of unlocking as Interupts





List of Syscalls

Name	Basic Synatx	Description
fork	pid_t fork(void);	Creates a new process by duplicating the calling process.
open	int open(const char *pathname, int flags, mode_t mode);	Opens a file.
exec	exec() has family of system calls.(one of them) int execve(const char *filename, char *const argv[], char *const envp[]);	Replaces the current process image with a new process image.
write	ssize_t write(int fd, const void *buf, size_t count);	Writes data to a file descriptor
lseek	off_t lseek(int fd, off_t offset, int whence);	Repositions the offset of the file descriptor.

List of Syscalls

Name	Basic Synatx	Description
dup	int dup(int oldfd);	Duplicates a file descriptor.
dup2	int dup2(int oldfd, int newfd);	Duplicates a file descriptor to a specified descriptor number
waitpid	pid_t waitpid(pid_t pid, int *status, int options);	Waits for a child process to change state.
munmap	int munmap(void *addr, size_t length);	Unmaps files or devices from memory.
mmap	void *mmap(void *addr, size_t length, int prot, int flags, int fd, off_t offset);	Maps files or devices into memory.

WHAT IS A PROCESS?

A process can be defined as an instance of a running program

. ...

What is a program?

It is a set of instructions that are used to complete a specific task.

Back to Agenda Page

PROCESS ID & GETPID()

- Process ID (PID) is a unique id that is assigned to every process in the system.
- It can be any number lesser than 32768 (2^{15}) and can be increased to 4194304 (2^{22}) on 64-bit systems.
- getpid() function can be called which returns the process ID of the calling process.
- Process management refers to the activities involved in managing the execution of multiple processes in an operating system

Back to Agenda Page

FORK( ) AND SPAWNING PROCESSES

fork() system call can be made to spawn a new process.

This new process is called the child of the process which made the system call

The original process making the system call is called the parent process.

The child process is an exact clone of the parent process and both of them execute the next instruction after fork()

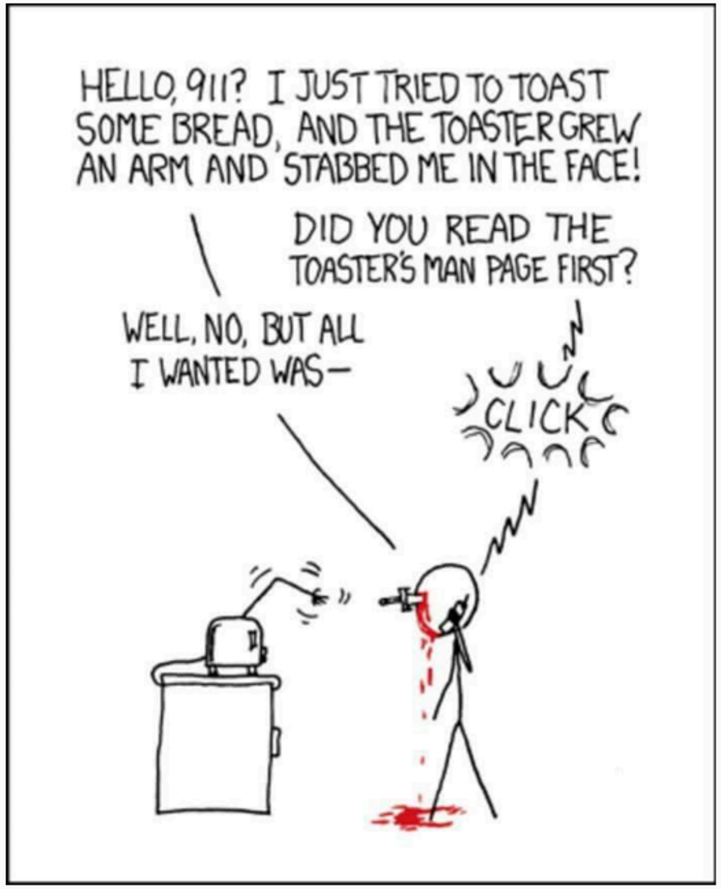
SOME GENERAL ADVICE

Read MAN pages.

Write modular code.

Start on time.

Legacy



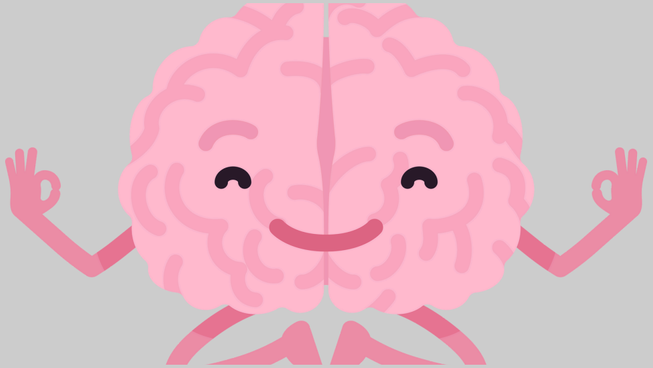
MAN PAGES

man [option] [section number] [command name]
man -f [command name]
man [section number] [command name]

- Name: The name of the command.
- Synopsis: The command's syntax.
- Configuration: Configuration details for a device.
- Description: A description of the command.
- Examples: Several examples demonstrating the use of the command.
- Defaults: The default functions of the command and how they can be overridden.
- Options: A list of options and flags that the command accepts.
- Exit Status: A list of possible exit status values for the command.
- Environment: A list and description of environment variables that affect the command.
- Files: A list of files used by the command.
- See also: Commands related to the described topic.
- Authors: The people who wrote or maintain the command.
- History: Command development history.
- Notes: Various notes, including permissions required, dependencies, etc.
- Bugs: Any known issues in this program version.

Back to Agenda Page

Is this useful for mp -0 ?????



ONE *TIP*
FOR GREAT
FUN IN MP-O : OSTEP INTERLUDE