UNIX Commands

Some common UNIX commands to research and become proficient with are:

COMMON UNIX COMMANDS	SOLARIS COMMANDS	LINUX COMMANDS
chmod	pargs	atop
chown	pcred	htop
fuser	pfiles	lsof
kill	pldd	pstree
pgrep	preap	
pkill	prstat	
pmap	prun	
ps	psig	
top	pstop	
	ptime	
	ptree	
	pwdx	

What Is a Process?

A process is a program in execution with all its resources. Such resources include the address space of the process that is composed of:

- The process stack: It's the area of the process address space that is used for the execution of functions. Local function variables are stored there as well as the functions call sequence.
- The process heap: It is the area of memory that is used for bulk storage. The process can request more memory there for various reasons.
- The process code: This is where the actual instructions that run on the processor are stored
- The process data: It's the area of memory where global and static variables are stored.

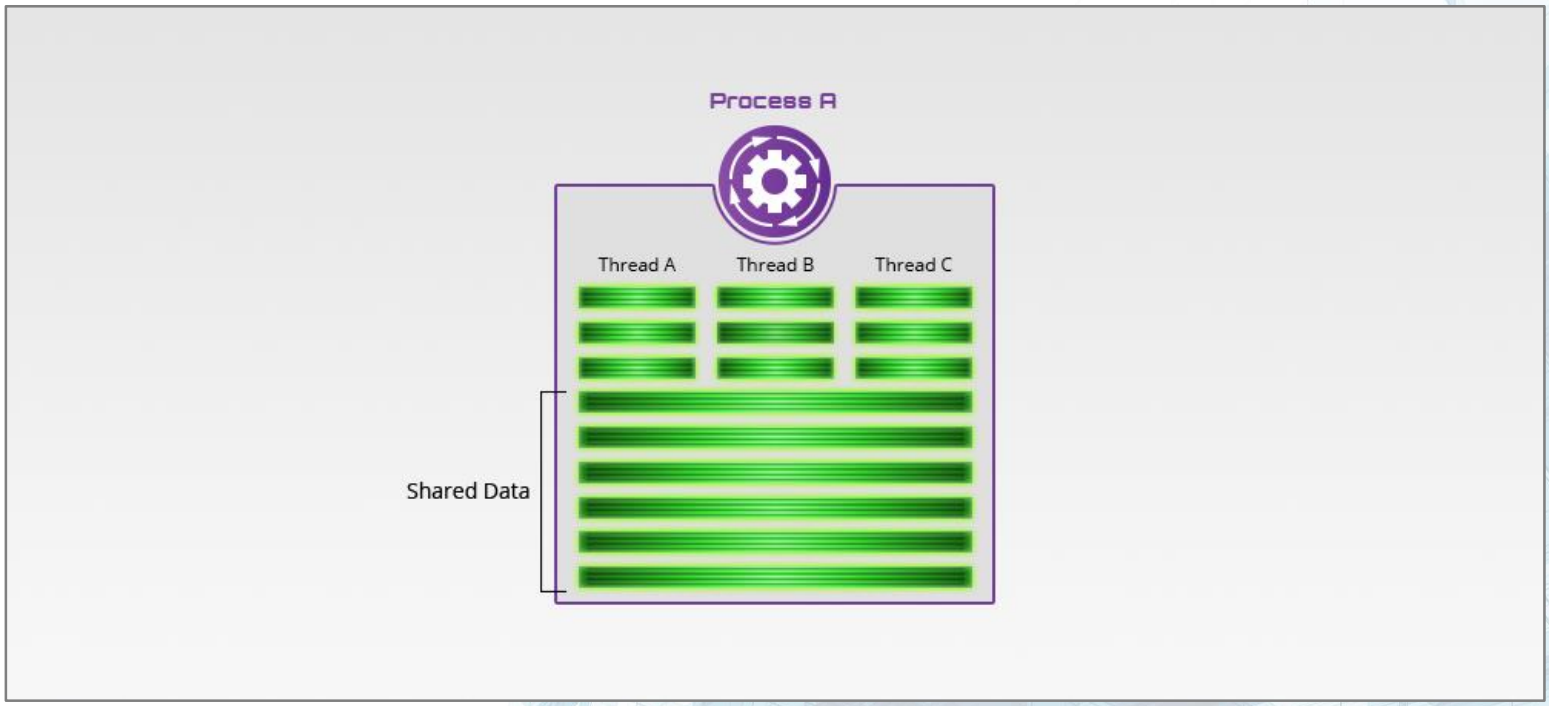
Additional resources include: the process descriptor which is used for the management and scheduling of the process by the system. The process descriptor holds fields for process signaling, file descriptors, context switching and so on. Pretty much any operation that can be performed on a process also has one or more corresponding fields to support that operation.

The UNIX kernel maintains the list of all running processes in a table called the process table.

The attached diagram is a simple representation of a process on a UNIX system.

Multithreaded Model

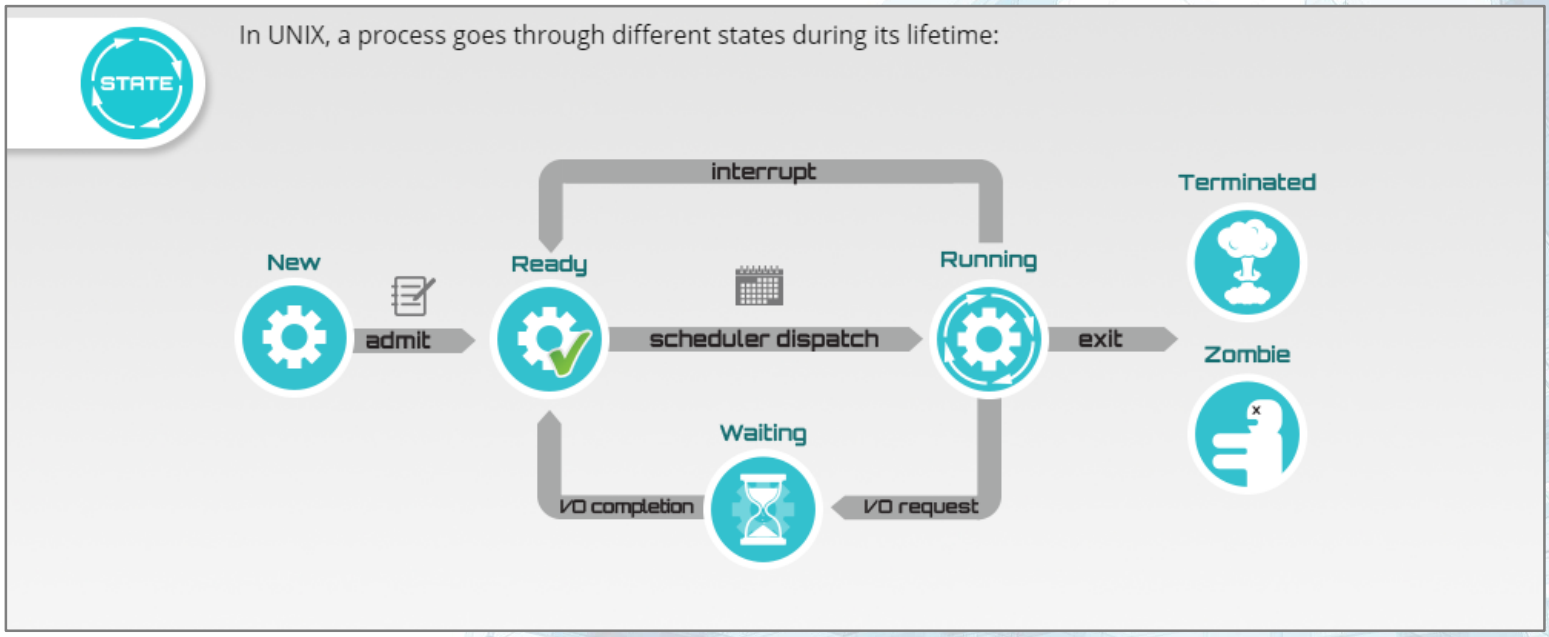
In the multithreaded model, a process can run multiple threads of execution concurrently. Thread share the process code, process data, and loaded libraries. However, each thread can have its own private data section, its own stack, and a thread descriptor. Because of shared resources, communication between threads is very efficient.



Process States

At any given time, the process can be in either of the following states:

- New: The process is being created, the process descriptor is allocated but the binary has not been mapped to memory yet.
- Ready: The process is fully created and mapped in memory. It can now be schedule for running.
- Running: The process is being executed by the processor
- Terminated: The process execution is finished or the process execution is forcibly ended.
- Waiting/Suspended: The process is waiting on an event or the completion of an I/O operation. Hence, the process is put to sleep until the event or the I/O is completed, then the process is returned to the ready state.



UNIX Signals and Signal Handling

Traditionally, signals go through two defined stages: generation and delivery. Signal generation is the signal sending phase. An example of signal generation is the use of the kill command (which has an equivalent system call that runs whenever the command is executed). The delivery phase is whenever the signal is handled which includes ignoring the signal, performing the default action, or executing an installed handler. The way the signal is handled by a process or the system is also known as signal disposition.

Signals can also be blocked by using signal masks. Whenever that is done, the signal delivery is postponed and the signal is said to be pending.

The following table is a list of signals with their default disposition:

Signal	Description	Default Action
SIGHUP	Hangup. Usually means that the controlling terminal has been disconnected. This signal is generated when a user presses Ctrl+D at the shell command line.	Terminate process.
SIGINT	Interrupt from keyboard. This signal is generated when a user presses Ctrl+C at the shell command line.	Terminate process.
SIGQUIT	Quit from keyboard. This signal is generated when a user presses Crtl+\ at the command line.	Terminate process and create a core dump.
SIGKILL	Kill signal. This is a sure kill because it cannot be trapped (caught) or ignored (masked).	Terminate the process.
SIGTERM	Termination signal. A gentle kill that gives processes a chance to clean up.	Terminate the process.
SIGCHLD	Child status changed (stopped or terminated).	Ignore the signal.
SIGSTOP	Stop signal. Pauses a process. This signal cannot be caught or ignored.	Stop process.
SIGCONT	Resume signal. Can be caught/trapped but cannot be ignored/masked.	Continue the process if it is currently stopped. Otherwise, ignore.
SIGTSTP	Stop signal. Pauses a process. Can be caught or ignored.	Stop process.
SIGTTIN	Terminal Input from background process.	Stop process.
SIGTTOU	Terminal output from background process.	Stop process.
SIGFPE	Arithmetic exception. Informs a process of a floating-point error.	Terminate process and create a core dump.
SIGSEGV	Segmentation Fault (or Invalid Memory Access).	Terminate process and create a core dump.
SIGSYS	Bad system call.	Terminate process and create a core dump.

Signal	Description	Default Action
SIGILL	Illegal instruction.	Terminate process and create a core dump.
SIGPIPE	Broken pipe. Write to pipe with no reader.	Terminate process.
SIGALRM	Timer signal from an Alarm Clock.	Terminate process.
SIGUSR1	User-defined signal 1.	Terminate process.
SIGUSR2	User-defined signal 2.	Terminate process.
SIGPWR	Power failure or system restart.	Terminate process.

Executable and Linkable Format (ELF)

The Executable and Linkable Format is a common standard file format for executables, object code, share libraries, and core dumps. ELF is flexible and extensive by design and is not bound to a particular processor or architecture. It is used on all major variants of UNIX including System V, Solaris, Linux and BSD.

Each running program on UNIX exists as an ELF file on the filesystem before it gets loaded an executed by the operating system. When the program is loaded, the sections of the ELF file get mapped into memory as well as libraries on which the program depends.

Analyzing a program ELF file can help you gleaned additional information about a specific process and the system. For example, signatures can be extracted from an executable binary and fed to an Antivirus product or an Intrusion Detection System in order to prevent future attacks on other machines.

There are various utilities that can be used to retrieve information from an ELF file:

Utility	Description
strings	The strings command/utility finds and prints all printable strings in an object or binary file. It's one of the first tool used when it comes to analyzing a binary file to retrieve signatures.
file	The file utility can display some information about ELF files, including the architecture for which the code in an executable or shared object file is intended, or on which an ELF core dump was produced.
elfdump	The elfdump utility displays information about ELF files. It is available mostly on FreeBSD and Solaris.
objdump	The objdump utility displays information from object files. It is mostly available on BSD and Linux.

Recommended Readings

- UNIX and Linux System administration Handbook 4th Edition (Chapter 5)

Recommended Internet Sites

- Process List Analysis for Solaris 8: https://web.archive.org/web/20160726215841/http://www.symantec.com/connect/articles/back-basics-solaris-default-processes-and-initd-pt-i
- The TTY Demystified: https://web.archive.org/web/20160726215944/http://www.linusakesson.net/programming/tty/
- Process credentials man page: https://web.archive.org/web/20160726220031/http://man7.org/linux/man-pages/man7/credentials.7.html
- UNIX Signals on Wikipedia: https://web.archive.org/web/20160726220116/https://en.wikipedia.org/wiki/Unix_signal
- Introduction to UNIX Signals and System Calls: https://web.archive.org/web/20150109205332/http://ph7spot.com/musings/introduction-to-unix-signals-and-system-calls

Please contact the Course Coordinators if you are unable to access any of the Recommended Internet Sites.