Solaris Process And Thread Management

PREPARED BY
Ahmed Salah Aldain
ECE-MASTAER

What is a Process

- The most basic abstraction provided by an operating system.
- A process is an executable object located at physical memory pages.
- Contains specific memory segments with executable instructions, stack space, data space, and other components needed for execution.
- Each process is identified uniquely with a positive integer number named PID.

Introduction to process Management

- Solaris is a multi thread operating system
- every tasks performed by operating system are execute like kernel threads.
- User threads are created in order to execute user processes, these user threads are created with a lightweight process or LWP linked to it..
- LWP is a kernel object which lets user threads enter to the kernel and execute by themselves.

Process Mgmt.

- Despite user LWP and kernel LWP have different structures, they are so integrated that can be seen like an unique execution entity.

- Process state is represented as a set of bits used by the kernel for process managing. From the process' "point of view" kernel puts all the execution resources in a virtual machine for its execution.

Process Mgmt.

- Kernel maintains a process structure named proc_t for each process in the system; proc_t contains and references all the process state data, is located at kernel address space and has restricted access.

- Every process starts from an executable disk file. A process image is loaded in memory by the kernel for its execution. This happens when a fork() system call is called. A PID is assigned to the recently created process. The process who called fork() is named parent process, and the new process is named child process.

Process Mgmt.

- A process can be ended due to:

·The process calls exit() system call, and all its threads end.

· Function abort() is called, which sends a SIGABRT   signal to the .process.

· The process ends its execution normal.

- In all of the previous cases exit() kernel function is executed, this function frees all the resources assigned to the process, and set process' status to zombie. A zombie process needs that its parent calls wait() system call, which captures exit status of its child process, and frees its entry from process table.

Distinction between threads and processes.

Processes:

- Virtual address space
- Protected access to processors, other process, files and I/O resources
- suspension and termination

Threads:

- execution state
- save context
- an execution stack
- some per-thread static storage
- access to the memory and resources of its process, shared with other threads of the process

Basic Process Management Commands in Solaris

- consists of listing active processes (PS command)
- terminating processes (kill command) .
- changing the execution priority of a process (nice and priocntl commands).

Process Management contd.

- Processes are represented in a treelike fashion by the process number in the /procpseu do file system.
- Most processes are associated with a terminal, a process without a terminal is called a Daemon.
- A child process is a process started and controlled by another process.
- A zombie process is a process that have hung, usually it’s a child process who's parent process has terminated without cleaning up after itself.

List of Solaris commands for process management

apptrace	for tracing library calls
dtrace	debugger, new in version 10
pargs	get list of arguments and environment variables with which process was started
pfiles	list of file descriptors, associated with process
pgrep	get the PID's of processes by name i.e. Something like PS -efl|grep -v grep|grep process name
pkill	send signal to process. For example pkill -9 in it :-P
pldd	– list dynamic libraries, associated with process, similar to ldd for executable
plockstat	see list of locked by process files. Lock can be mutex i.e. exclusive and reader/writer for shared access
pmap	– get memory map (segments) of process
preap	try to kick-off zombie process

List of Solaris commands for process management(contd.)

prstat	Full screen view of processes sorted by different criteria, similar to Linux top command
prun	Continue hold with pstop process
PS	Print process information and status. In Solaris exist SYSV and BSD variants, respectively /usr/bin/PS and /usr/ucb/PS
psig	List signals that can be handled by process
pstack	Get back trace stack of process for debugging purposes
pstop	Temporary hold process
ptree	Print the tree of processes
pwait	Wait till process finish
pwdx	List working directory for process, like pwd command
truss	For tracing user/library/system calls and signals

User-level and Kernel-level Threads

User-level threads:

- Thread management is internal to the process
- The kernel is unaware of their existence.
- A threads library provides code for thread management needs of the process.

Kernel level threads:

- Management of threads is done entirely at the kernel level
- An application programming interface is provided for thread creation, termination and synchronization

THANK YOU