Advanced Unix System Administration

Lecture 4
February 8, 2007

Steven Luo
<sluo+decal@OCF.Berkeley.EDU>

Processes

- Scheduling considerations
  - Priority: higher-priority tasks should run more often
  - Starvation: processes that haven't run in a long time should run
  - (SMP systems) Processor affinity
  - Locks held by processes; priority inversion
  - Different workloads benefit from different algorithms for sorting this out

Processes

- Process creation
  - fork() and friends – creates a copy of the parent
  - If a new program is being invoked, the exec family of functions overwrites the address space with the appropriate code
  - Dynamic binaries: the dynamic linker loads code (more below)
  - Start of program execution

Processes

- The process tree
  - Every process has a parent – the process from which it fork()ed
  - Parent has privileges (and responsibilities) with regards its children
  - Parent and children form a process group, which is also assigned an ID number
  - The start of the process tree is init (always PID 1)
  - Orphaned processes are inherited by init

Processes

- Signals
  - Allow processes to communicate with each other and the kernel
  - Provide primitive mechanism for implementing callbacks – signals can be trapped and a “signal handler” called
  - If not handled, signals perform a default action (usually exit)
  - Signal programming is tricky because of synchronization and syscall restarting issues
  - Try `man kill` or `kill -L` for more information

Processes

- Threads
  - Recall that the kernel keeps lots of state for each process
  - But if the processes are related, we might be able to get away with less of that
  - Threads = “lightweight processes”
  - When threads have kernel support, they're much faster to create and switch
  - Shared resources means programming is more difficult

Shared Libraries

- The dynamic linker
  - Binaries have a “symbol table” containing functions, etc. and their locations
  - Dynamic binaries have tables with blanks – it's the responsibility of the dynamic linker to resolve these
  - Linker loads listed dynamic libraries and tries to resolve the symbols
  - Allows shared code, but incurs a performance penalty on most architectures