Linux System Administration-I
CSE-4043

Chapter 5:Controlling Processes

NIBEDITA JAGADEV

Department of CSE

Asst. Professor

SOA Deemed to be University, Bhubaneswar, Odisha , India

nibeditajagadev@soa.ac.in

Contents

- **PROCESS**
- **COMPONENTS OF A PROCESS**
- **THE LIFE CYCLE OF A PROCESS**
- **SIGNALS**

Process

- A process is the abstraction used by UNIX and Linux to represent a running program. It's the object through which a program's use of memory, processor time, and I/O resources can be managed and monitored.

COMPONENTS OF A PROCESS

- The process's address space map
- The current status of the process (sleeping, stopped, runnable, etc.)
- The execution priority of the process
- Information about the resources the process has used
- Information about the files and network ports the process has opened
- The process's signal mask (a record of which signals are blocked)
- The owner of the process

COMPONENTS OF A PROCESS

- PID: process ID number
- PPID: parent PID
- UID and EUID: real and effective user ID

COMPONENTS OF A PROCESS

- GID and EGID: real and effective group ID
- Niceness
- Control terminal

THE LIFE CYCLE OF A PROCESS

- To create a new process, a process copies itself with the fork system call. Fork creates a copy of the original process; that copy is largely identical to the parent.
- The new process has a distinct PID and has its own accounting information.

SIGNALS

- Signals are process-level interrupt requests. About thirty different kinds are defined, and they’re used in a variety of ways.

SIGNALS

- The signals KILL, INT, TERM, HUP, and QUIT all sound as if they mean approximately the same thing, but their uses are actually quite different
- KILL is unblockable and terminates a process at the kernel level. A process can never actually receive this signal.
- INT is sent by the terminal driver when you type <Control-C>. It's a request to terminate the current operation. Simple programs should quit (if they catch the signal) or simply allow themselves to be killed, which is the default if the signal is not caught. Programs that have an interactive command line (such as a shell) should stop what they're doing, clean up, and wait for user input again.
- TERM is a request to terminate execution completely. It's expected that the receiving process will clean up its state and exit.

SIGNALS

- HUP has two common interpretations. First, it's understood as a reset request by many daemons. If a daemon is capable of rereading its configuration file and adjusting to changes without restarting, a HUP can generally be used to trigger this behaviour.
- QUIT is similar to TERM, except that it defaults to producing a core dump if not caught. A few programs cannibalize this signal and interpret it to mean something else.

Linux System Administration-I
CSE-4043

Chapter 5:Controlling Processes

NIBEDITA JAGADEV

Department of CSE

Asst. Professor

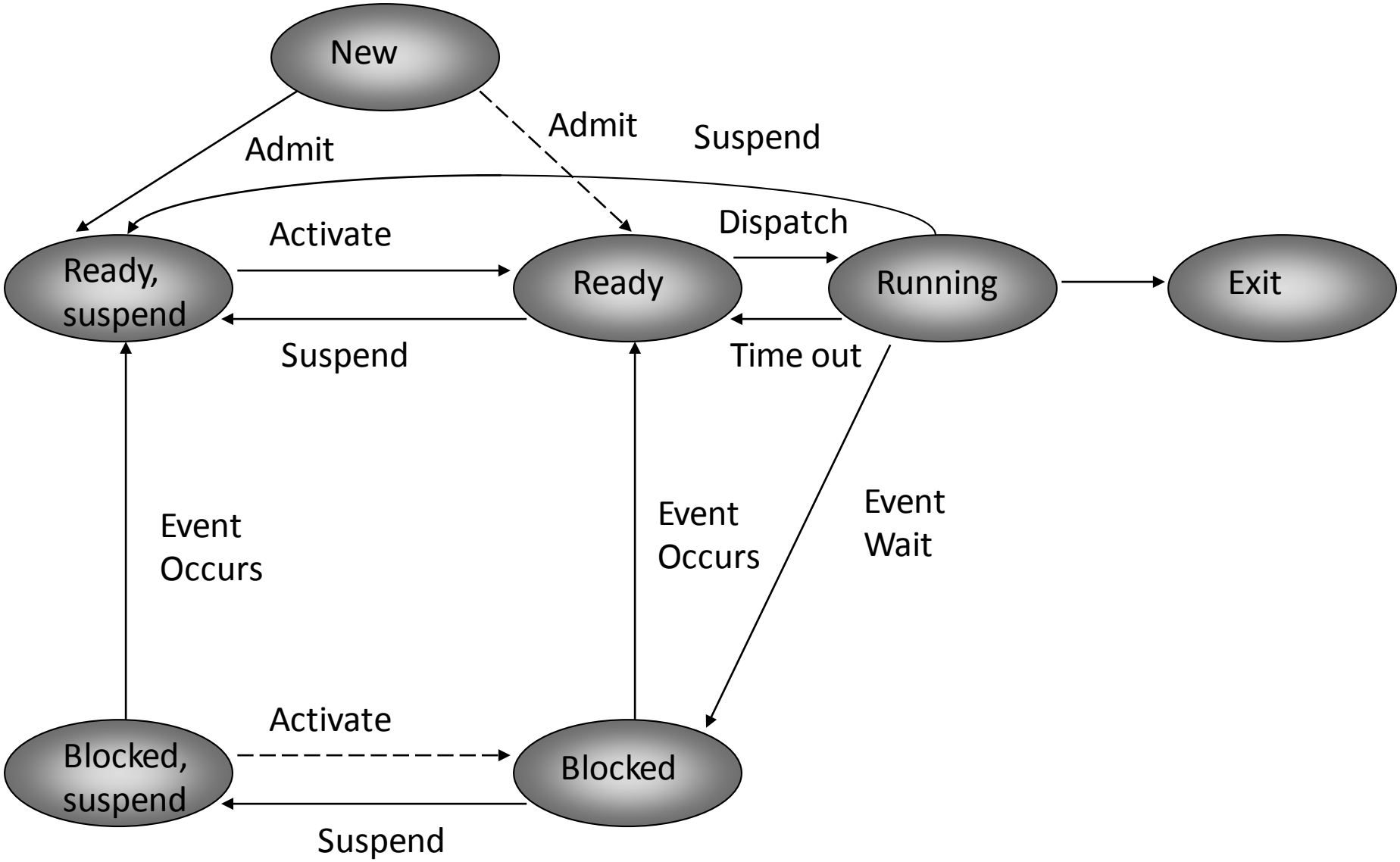
SOA Deemed to be University, Bhubaneswar, Odisha , India

nibeditajagadev@soa.ac.in

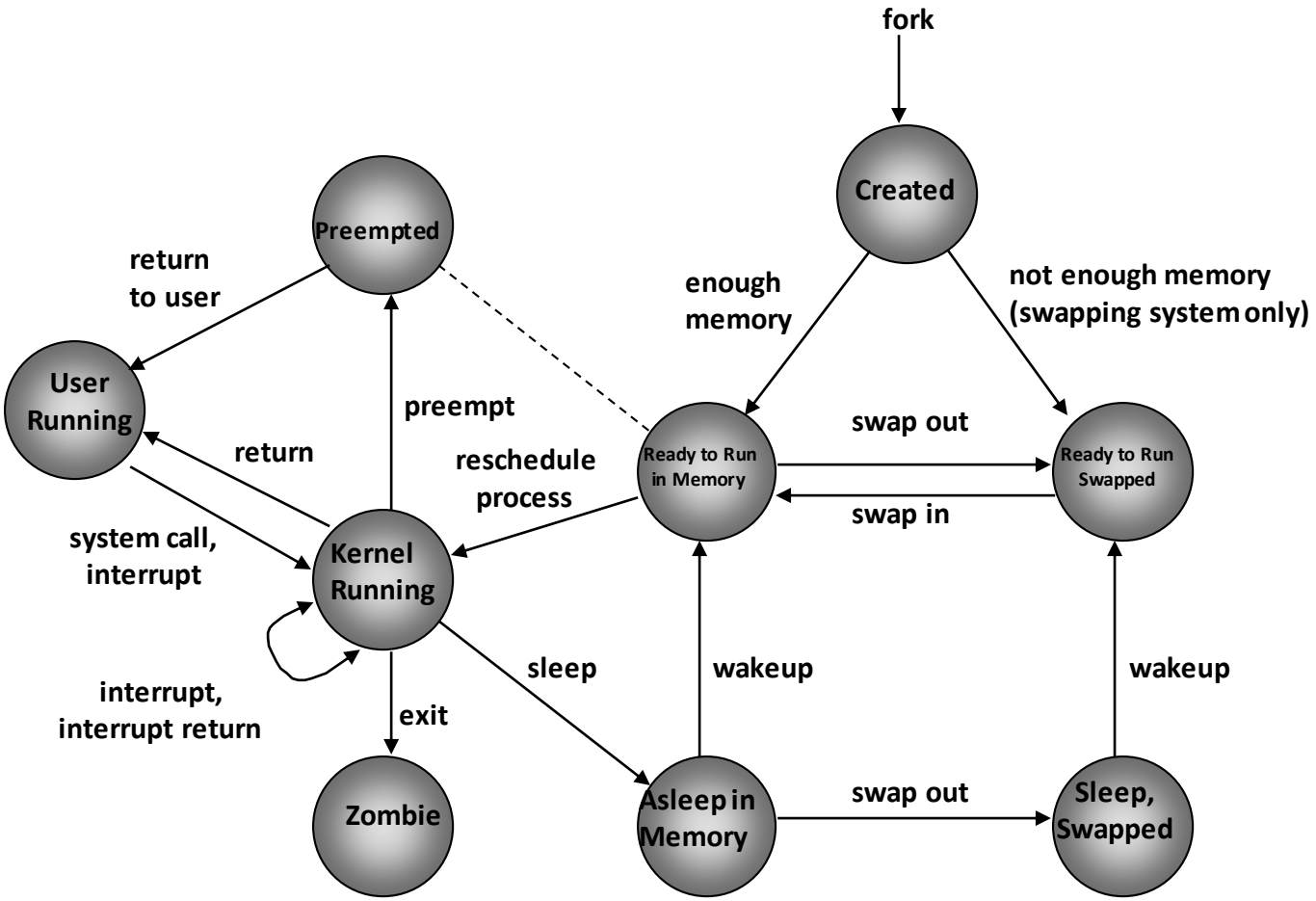
Contents

- PROCESS STATE TRANSITION DIAGRAM
- UNIX PROCESS STATE TRANSITION DIAGRAM
- PROCESS CONTROL BLOCK
- UNIX PROCESS CONTROL TABLE
- PROCESSOR STATE INFORMATION
- NICE AND RENICE
- PS COMMAND

Process State Transition Diagram with Two Suspend States - Seven-State Process Model



UNIX Process State Transition Diagram (1)



UNIX Process State Transition Diagram (2)

- User running: Executing in user mode.
- Kernel running: Executing in kernel model.
- Ready to run, in memory: Ready to run as soon as the kernel schedules it.

UNIX Process State Transition Diagram (3)

- Asleep in memory: unable to execute until an event occurs; process in main memory.
- Ready to run, swapped: process is ready to run, but the the swapper must swap the process into main memory before the kernel can schedule it to execute.
- Sleeping, swapped: The process is awaiting an event and has been swapped to secondary storage.

UNIX Process State Transition Diagram (4)

- Preempted: process is returning from kernel to user mode, but the kernel preempts it and does a process switch to schedule another process.
- Created: process is newly created and not yet ready to run.
- Zombie: process no longer exists, but it leaves a record for its parent process to collect.

Process Control Block

- Process Control Block

  The collection of attributes is refereed to as process control block.

- Unique numeric identifier
  - may be an index into the primary process table

UNIX Process Control Table

- Process Identifiers

  ID of this process and ID of parent process.
- User Identifiers

  real user ID, effective user ID
- Pointers

  To U area and process memory (text, data, stack)
- Process Size, Priority, Signal, Timers, ...

Processor State Information

- Contents of processor registers
  - User-visible registers (data/address register).
  - Control and status registers (program counter, instruction register.
  - Stack pointers (points to the top of the stack).
- Program status word (PSW)
  - contains status information - indicate the mode of execution (user or kernel mode) ??????

NICE AND RENICE(INFLUENCE SCHEDULING PRIORITY)

- The “niceness” of a process is a numeric hint to the kernel about how the process should be treated in relation to other processes contending for the CPU
- A high nice value means a low priority for your process: you are going to be nice. A low or negative value means high priority: you are not very nice.

PS(MONITOR PROCESSES)

- **ps** is the system administrator’s main tool for monitoring processes
- **ps** can show the PID, UID, priority, and control terminal of processes

Thank You