Linux Operating System and Applications

Process Management

Program & Process

- A **program** is an executable file on the system (e.g., `/sbin/init`, `/sbin/shutdown`)
- A **process** is a running instance of a program
- Multiple processes of the same program can run at once (e.g., multiple Word windows)
- The Linux kernel supports multitasking: running many processes simultaneously by sharing CPU time

Process Priorities

- Each process has a **priority** that affects how much CPU time it receives
- Priority is determined by the **nice value**
- **Lower nice value = higher priority (runs sooner):** `-20` highest, `+19` lowest
- Priority can be set when the process starts (`nice`) or while running (`renice`)

```
nice -n -5 ./backup.sh
renice -10 -p 2034
```

Types of process in Linux

- **Daemon**: Background process offering system services (e.g., `cron`, `sshd`)
- **Parent**: Process that creates (spawns) others using `fork()`
- **Child**: Spawned by a parent (e.g., shell scripts, worker processes)
- **Orphan**: Parent exited, but child keeps running (adopted by `init` or `systemd`)
- **Zombie**: Finished execution but not cleaned by its parent

Daemon Processes

- Run in the background without a terminal
- Responsible for system functions
- Typically shown with `?` in the TTY column when using `ps`
- Wait for specific events (signals, input, timeouts) and handle them in the background

```
systemctl status sshd
ps aux | grep cron
```

Zombie & Orphan Processes

- **Zombie**: Process has exited, but still has an entry in the process table to report status to parent
- **Orphan**: Parent process is gone; adopted by `init` (`PID=1`)
- Use `ps -el | grep Z` to find zombies
- System should automatically clean orphans, but many zombies indicate a programming flaw

Process attributes

- **PID**: Unique process ID
- **PPID**: Parent process ID
- **UID / GID**: User and group ownership (affects permissions)
- **CMD**: Command used to launch the process
- Use `ps -eo pid,ppid,uid,gid,cmd` for detailed view

Foreground & Background Processes

- **Foreground**: Runs with user interaction (e.g., terminal commands)

```
ls -l /etc
```

- **Background**: Detached from terminal, continues after logout

```
./backup.sh &

nohup ./longtask.sh &
```

View Running Processes

- `ps`: Show snapshot of running processes
- `pstree`: Visualize process hierarchy
- `top`: Live monitoring of CPU, memory, and processes

Using ps

- Basic usage

```
ps aux
ps -ef
ps -U john
ps -eo pid,ppid,%cpu,%mem,cmd --sort=-%cpu
```

- Combine with grep to filter

```
ps aux | grep apache2
```

ps Output Explanation

Column	Meaning
UID	User ID
PID	Process ID
PPID	Parent Process ID
%CPU	CPU usage
%MEM	Memory usage
CMD	Command used to start process

Using pstree

- Visual tree of parent and child processes
- Helps understand service structure (e.g., web server with multiple workers)

```
pstree
pstree -p  # show PIDs
```

Monitoring with top

Displays processes and system stats in real time

Press `M` to sort by memory, `P` to sort by CPU

Options:

```
top -n 3 -d 2  # refresh every 2s, run 3 times
```

Useful for finding processes consuming high resources

top

```
top - 23:08:08 up 12 min,  2 users,  load average: 0.75, 0.68, 0.42
Tasks: 107 total,   2 running, 105 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  1.3%sy,  0.0%ni, 98.0%id,  0.3%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    255392k total,   249748k used,     5644k free,    13176k buffers
Swap:   650624k total,        0k used,   650624k free,   139668k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2486 root      15   0 31768 7380 3276 S  1.3  2.9   0:11.22 Xorg
 1843 root      15   0  2380  860  712 S  0.3  0.3   0:01.72 vmware-guestd
 2328 root      15   0  3140  868  752 S  0.3  0.3   0:00.15 hald-addon-stor
 2731 root      15   0  8224 2528 1924 S  0.3  1.0   0:01.57 vmware-user
    1 root      15   0  2140  628  540 S  0.0  0.2   0:01.06 init
    2 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.99 ksoftirqd/0
    4 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    5 root      10  -5     0    0    0 S  0.0  0.0   0:00.07 events/0
    6 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 khelper
    7 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 kthread
   46 root      10  -5     0    0    0 S  0.0  0.0   0:00.03 kblockd/0
   47 root      20  -5     0    0    0 S  0.0  0.0   0:00.00 kacpid
  111 root      20  -5     0    0    0 S  0.0  0.0   0:00.00 cqueue/0
  112 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 ksuspend_usbd
  115 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 khubd
  117 root      10  -5     0    0    0 S  0.0  0.0   0:00.00 kseriod
```

Shell

top

Trường	Giải Thích
USER hoặc UID	Tên của tiến trình
PID	ID (định danh) của tiến trình
%CPU	% CPU sử dụng của tiến trình
%MEM	% bộ nhớ tiến trình sử dụng

top

SIZE	Kích thước bộ nhớ ảo tiến trình sử dụng
RSS	Kích thước của bộ nhớ thực sử dụng bởi tiến trình
TTY	Vùng làm việc của tiến trình
STAT	Trạng thái của tiến trình
START	Thời gian hay ngày bắt đầu của tiến trình
TIME	Tổng thời gian sử dụng CPU

top

COMMAND	Câu lệnh được thực hiện
PRI	Mức ưu tiên của tiến trình
PPID	ID của tiến trình cha
WCHAN	Tên của hàm nhân khi tiến trình ngủ được lấy từ file /boot/System.map

FLAGS	Số cờ được kết hợp với tiến trình

Managing Processes with kill

- Terminate or control processes by sending signals

```
kill -9 1234  # force kill
kill -15 1234 # default termination
```

- List of signals

```
kill -l
```

- Common signals

Signal	Number	Action
SIGINT	2	Interrupt (Ctrl+C)
SIGTERM	15	Graceful termination
SIGKILL	9	Immediate termination
SIGTSTP	20	Pause (Ctrl+Z)
SIGCONT	18	Resume paused process

nice & renice Commands

- **nice**: Start-time priority
  - Set initial priority of a process

    ```
    nice -n 10 ./report.sh
    ```

  - Default nice value is 0
  - Only root can set negative (higher priority) values
- **renice**: Runtime priority
  - Change priority of running processes

    ```
    renice -5 -p 2310
    renice 10 -u john
    ```

  - Helps adjust performance of long-running services or background tasks

Background Execution with &

- Add & at the end of a command to run it in the background

```
ping google.com > ping.log &
```

- Use `jobs` to list background tasks
- Use `fg` to bring one to foreground

Job Control with Ctrl+C, fg, bg, Ctrl+Z

- `Ctrl+C:` stop a running process
- `Ctrl+Z`: pause foreground process
  - `bg`: resume in background
  - `fg`: bring paused job back to foreground

```
fg %1
bg %2
```

Managing Daemons with System Scripts

- If SysV init scripts exist:

```
/etc/init.d/httpd start

/etc/init.d/cron stop
```

Managing Services with systemctl

- Modern distros use `systemd`

```
systemctl start nginx

systemctl stop nginx

systemctl restart nginx

systemctl status nginx
```

Autostart systemctl

- Enable or disable service autostart

```
systemctl enable nginx

systemctl disable nginx
```

Summary

- Linux processes are flexible and powerful
- Tools: `ps`, `top`, `kill`, `nice`, `renice`, `jobs`, `systemctl`
- Proper process and daemon management is essential for system stability and performance

Q&A