Linux para Ingeniería:

Understanding Linux Processes

Luis Garreta
luis.garreta@javerianacali.edu.co

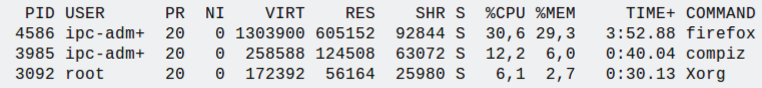
Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana – Cali

20 de abril de 2018

Processes

Overview of a Process

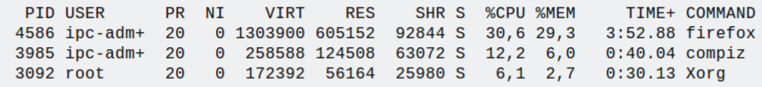
- A process is an instance of a computer program that is currently being executed.
- Associated with a process is a variety of attributes:
  - ✓ ownership,
  - ✓ nice value,
  - ✓ priority, and
  - ✓ SELinux context, to name a few



```
  PID USER       PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 4586 ipc-adm+   20   0 1303900 605152  92844 S  30,6 29,3   3:52.88 firefox
 3985 ipc-adm+   20   0  258588 124508  63072 S  12,2  6,0   0:40.04 compiz
 3092 root       20   0  172392  56164  25980 S   6,1  2,7   0:30.13 Xorg
```

- These atributes extend or limit its ability to access resources on the computer.

Changing the Priority of a process: *nice and renice*



```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 4586 ipc-adm+  20   0 1303900 605152  92844 S  30,6 29,3   3:52.88 firefox
 3985 ipc-adm+  20   0  258588 124508  63072 S  12,2  6,0   0:40.04 compiz
 3092 root      20   0  172392  56164  25980 S   6,1  2,7   0:30.13 Xorg
```

- PR – Priority The scheduling priority of the task. If you see 'rt' in this field, it means the task is running under 'real time' scheduling priority.
- NI – Nice Value The nice value of the task. A negative nice value means higher priority, whereas a positive nice value means lower priority

1. You can launch a programe with your required priority using:

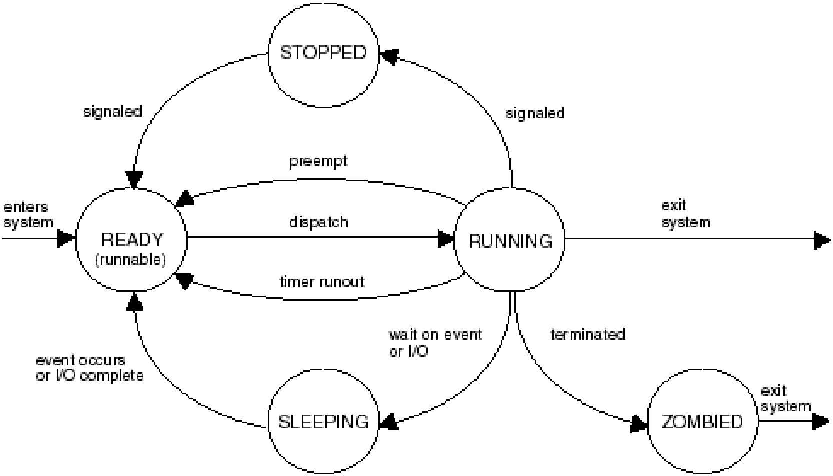
```
#nice -n nice_value program_name
nice -n 10 apt-get upgrade
```

2. You can also change the priority of an already running process using

```
#renice -n nice_value -p process_id
renice 10 -p 21827
```

Life of a Process: States

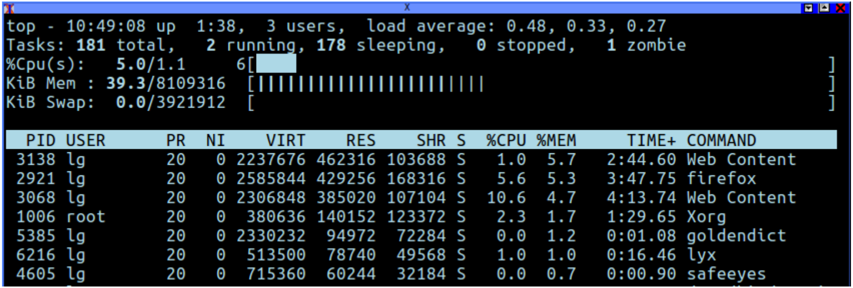
During the life of a process, it can go through different states



Display Linux processes commands: **ps, top, htop**

```
# ps - report a snapshot of the current processes.
ps -ef
```

```
# top - top - display Linux processes
top
```



```
top - 10:49:08 up  1:38,  3 users,  load average: 0.48, 0.33, 0.27
Tasks: 181 total,   2 running, 178 sleeping,   0 stopped,   1 zombie
%Cpu(s):   5.0/1.1     6[                                                              ]
KiB Mem :  39.3/8109316  [||||||||||||||||||||||                                        ]
KiB Swap:   0.0/3921912  [                                                              ]

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 3138 lg        20   0 2237676 462316 103688 S   1.0  5.7   2:44.60 Web Content
 2921 lg        20   0 2585844 429256 168316 S   5.6  5.3   3:47.75 firefox
 3068 lg        20   0 2306848 385020 107104 S  10.6  4.7   4:13.74 Web Content
 1006 root      20   0  380636 140152 123372 S   2.3  1.7   1:29.65 Xorg
 5385 lg        20   0 2330232  94972  72284 S   0.0  1.2   0:01.08 goldendict
 6216 lg        20   0  513500  78740  49568 S   1.0  1.0   0:16.46 lyx
 4605 lg        20   0  715360  60244  32184 S   0.0  0.7   0:00.90 safeeyes
```

UNDERSTANDING PROCESS TYPES

- There are different types of processes in a Linux system.
- These types include:
  - ✓ user processes,
  - ✓ daemon processes, and
  - ✓ kernel processes

User Processes

- Most processes in the system are user processes.
- A user process is one that is initiated by a regular user account and runs in user space.
- Unless it is run in a way that gives the process special permissions, an ordinary user process has no special access to the processor or to files on the system that don't belong to the user who launched the process.

Daemon Process

- A daemon process is an application that is designed to run in the background,
- Typically managing some kind of ongoing service.
- A daemon process might listen for an incoming request for access to a service.
  - ✓ For example, the httpd daemon listens for requests to view web pages.
- Or a daemon might be intended to initiate activities itself over time.
  - ✓ For example, the crond daemon is designed to launch cron jobs at preset times.

Daemon Process

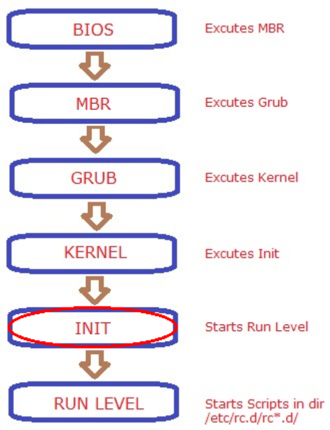
- Although daemon processes are typically managed as services by the root user, daemon processes often run as non-root users by a user account that is dedicated to the service.
- By running daemons under different user accounts, a system is better protected in the event of an attack.
  - ✓ For example, if an attacker were to take over the httpd daemon (web server), which runs as the Apache user, it would give the attacker no special access to files owned by other users (including root) or other daemon processes.
- Systems often start daemons at boot time and have them run continuously until the system is shut down
- Daemons can also be started or stopped on demand, set to run at particular system run levels, and, in some cases, signaled to reload configuration information on the fly.

Kernel Processes

- Kernel processes execute only in kernel space.
- They are similar to daemon processes.
  - ✓ But, kernel processes have full access to kernel data structures,
  - ✓ They are more powerful than daemon processes that run in user space.
- Kernel processes also are not as flexible as daemon processes.
  - ✓ You can change the behavior of a daemon process by changing configuration files and reloading the service.
  - ✓ Changing kernel processes, however, may require recompiling the kernel.

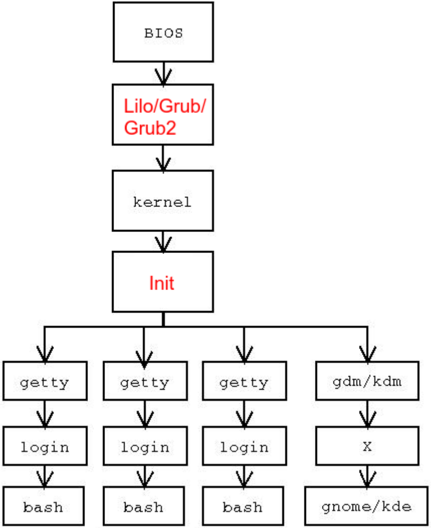
Controlling Linux Services: The init Process

The Init Process



The Init Process

- Commonly known as the mother of all processes.
- It will always have a Process ID of "1" (pid of 1)
- It is the first process started by the kernel on your system.
- init is short for initialization
- It is used to start all other processes and your default runlevel.
- Overtime many distributions have adopted different systems to managgin services:
  - ✓ the System V
  - ✓ the "Systemd" or
  - ✓ the "Upstart".



SysV - System V

- If your Linux distribution uses the "SysV" standards, then init will examine a file called "/etc/inittab.
- This file contains your default runlevel that the system should start under.
  - ✓ For example "id:3:initdefault:" would automatically start all the scripts in the runlevel 3 directory structures.
- Run Levels:
  - ✓ Run Level 1 - Single User Mode
  - ✓ Run Level 2 - Same as runlevel 3, but without NFS
  - ✓ Run Level 3 - Multi-user and networking mode
  - ✓ Run Level 4 - Unused
  - ✓ Run Level 5 - Same as runlevel 3, but with graphical desktop (X window System)

Running ”System V” init scripts

- Quite often as an administrator you will need to stop, start, restart or reload a particular service/daemon.
- To do this we use a command called ”service”.
- This command allows you to execute ”System V” scripts that are normally located within the ”/etc/init.d” directory.
- As well as being able to start and stop a service/daemon, we can also view the current status.

Service command examples

- Examples using the sshd daemon: ssh server
- SSH server is a program that allows logging from remote machines

Status Check:

Here we are requesting the current status of the ”sshd” daemon

```
# service sshd status
openssh-daemon (pid  1599) is running...
```

Status Check on all Processes:

- Runs all of your init scripts in alphabetical order with the status option

```
# service --status-all
abrt-ccpp hook is installed
abrtd (pid  1708) is running...
abrt-dump-oops (pid 1716) is running...
acpid (pid  1487) is running...
atd (pid  1740) is running...
auditd (pid  1262) is running...
automount (pid  1571) is running...
avahi-daemon (pid  1375) is running...
Usage: /etc/init.d/bluetooth {start|stop}
certmonger (pid  1752) is running...
cpuspeed is stopped
crond (pid  1724) is running...
cupsd (pid  1476) is running...
dnsmasq is stopped
firstboot is not scheduled to run
hald (pid  1496) is running...
```

Stop

- Here we requested that the ”sshd” daemon should stop.
- The status option was also issued to check the new status.

```
# service sshd stop
Stopping sshd:
                                                          [  OK  ]

[root@centos etc]# service sshd status
openssh-daemon is stopped
```

Start

- This time we are requesting the ”sshd” should be started.

```
# service sshd start
Starting sshd:
                                                [   OK   ]
```

Restart

- This time we are going to ”bounce” the ”sshd” daemon (stop and then immediately restart).

```
# service sshd restart
Stopping sshd:
                                                [  OK  ]
Starting sshd:
                                                [  OK  ]
```

Reload

- The ”reload” option is very useful if you have made changes to a configuration file and you want to bring these changes in.

```
# service sshd reload
Reloading sshd:
                                                     [  OK  ]
```

Controlling Linux Services: The Upstart Process

The Upstart Process

- Upstart is an event based replacement for the ”init” daemon.
- Upstart was written by a former employee of the well known company that provides us with Ubuntu (Canonical).
- The idea behind Upstart was to move away from the traditional start process whereby tasks that were started had to complete before the next task could start.
- Upstart is an event driven system that allows it to respond to system events asynchronously.
- Upstart is responsible for starting and stopping of services and tasks at boot and shutdown.
  - ✓ It also actively monitors these services and tasks.
- Various distributions included Upstart as a replacement for System V.
  - ✓ These have included RHEL, CentOS, Fedora.
- However, many of these systems have now moved to ”systemd”

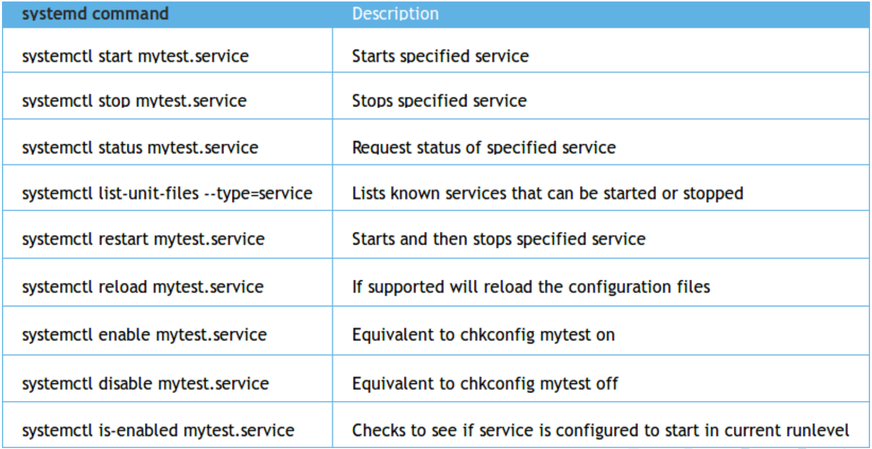
Controlling Linux Services: The systemd Process

The systemd Process

- systemd is another replacement to System V.
- systemd stands for system daemon.
- systemd was designed to allow for better handling of dependencies and have the ability to handle more work in parallel at startup.
- systemd supports snapshotting of your system and the restoring of your systems state
- keeps track of processes stored in what is known as a ”cgroup (control group)” as opposed to the conventional ”PID” method.
- systemd is now been taken up by many popular Linux distributions:
  - ✓ Fedora, Mandriva, Mageia, Arch Linux, Debian, Ubuntu, Redhat

systemd commands for controlling services

- systemd has its own unique set of commands for controlling services.
- The command that is used under systemd is ”systemctl”:



systemd command	Description
systemctl start mytest.service	Starts specified service
systemctl stop mytest.service	Stops specified service
systemctl status mytest.service	Request status of specified service
systemctl list-unit-files --type=service	Lists known services that can be started or stopped
systemctl restart mytest.service	Starts and then stops specified service
systemctl reload mytest.service	If supported will reload the configuration files
systemctl enable mytest.service	Equivalent to chkconfig mytest on
systemctl disable mytest.service	Equivalent to chkconfig mytest off
systemctl is-enabled mytest.service	Checks to see if service is configured to start in current runlevel