novobi

AUTOMATE
OPTIMIZE
TRANSFORM

DevOps Summer Internship 2023 Week 1

www.novobi.com

WEEK 1
SYSTEM ENGINEERING

Agenda

- Linux OS
- Command Line Proficiency
- System Administration
- Shell Scripting and Automation
- Networking and Security
- Vagrant

LINUX OS

Linux OS - Basic concept

- Open-source, based on UNIX kernel
- Many distributions
- Command Line Interface (CLI)
- Multi-user, multi-programming
- Filesystem hierarchy
- Security
- Everything is a file

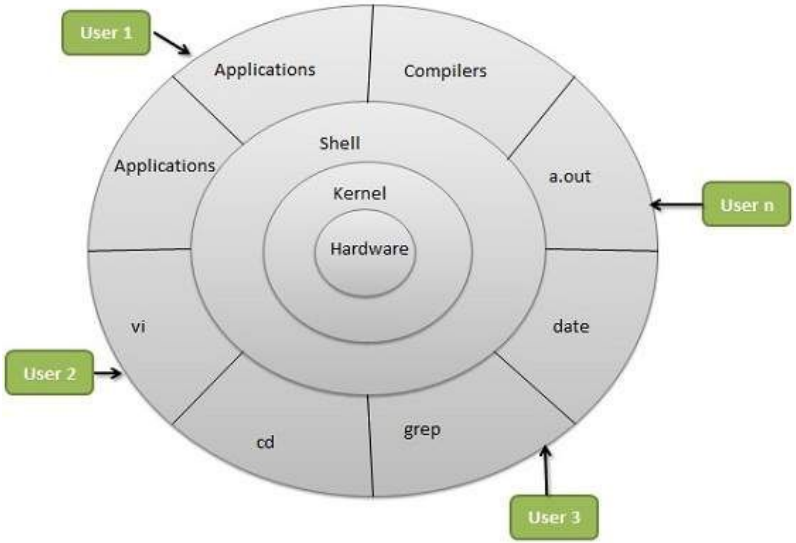
Linux OS - File

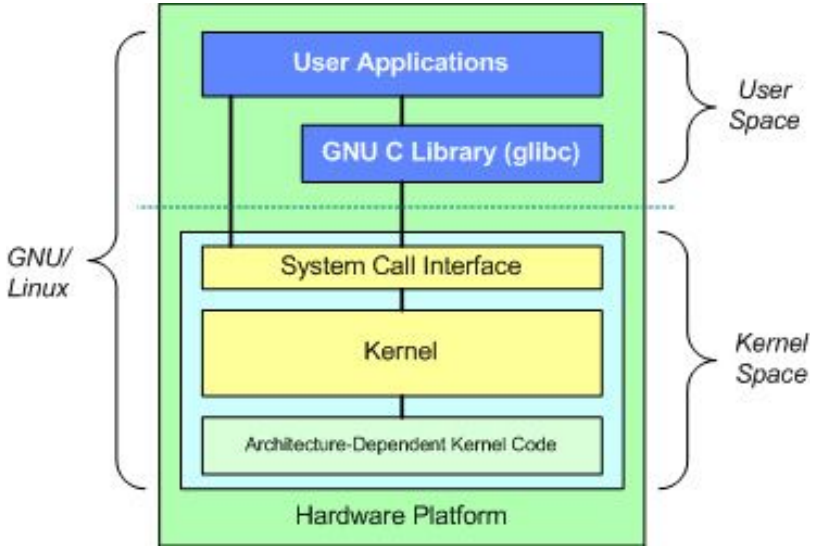
6 types of file:

- Regular file
- Directory
- Link
- Special file
- Socket
- Pipe

Linux OS - Architecture

- Popular architecture: Monolithic





Linux OS - Distribution

- There are many and many distributions for Linux OS
- Can divide 3 types:
  - **RPM-based:** Fedora, RHL, RHEL, CentOS,...
  - **Debian-based:** Ubuntu,...
  - **Pacman-based:** Arch Linux,...
- Different distributions has different packet manager

COMMAND LINE PROFICIENCY

Command Line Proficiency

- Text-based user interface used to interact with operating systems by commands.
- Known as Terminal, shell, console.

Navigating Files

- pwd
- cd
- ls
- touch
- file
- cat
- history

Navigating Files

- cp
- mv
- mkdir
- rm
- find
- alias

Process

- Programs running on machine.
- Managed by kernel
- Each process has an ID (PID)

```
$ ps au

USER               PID  %CPU %MEM      VSZ    RSS   TT  STAT STARTED      TIME COMMAND
root             14890   0.0  0.0 34130140   1048 s000  R+   11:24AM   0:00.02 ps a
thuscomputer     14868   0.0  0.0 34168200   1488 s000  S    11:23AM   0:00.03 -bas
root             14867   0.0  0.0 34161288   3280 s000  Ss   11:23AM   0:00.04 logi
```

Process

Signals

- Notify to a process that something has happened.
- Defined by numbers and symbolic names (SIGxxx)
- Process control, event notification and resource management.

Process

Tracking Process

- Get a real-time view of the system utilization by processes.

```
18:06:26 up 6 days,  4:07,  2 users,  load average: 0.92, 0.62, 0.59
Tasks: 389 total,   1 running, 387 sleeping,   0 stopped,   1 zombie
%Cpu(s):  1.8 us,  0.4 sy,  0.0 ni, 97.6 id,  0.1 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:  32870888 total, 27467976 used,  5402912 free,   518808 buffers
KiB Swap: 33480700 total,    39892 used, 33440808 free. 19454152 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND

 6675 patty    20   0 1731472 520960  30876 S   8.3  1.6 160:24.79 chrome
 6926 patty    20   0  935888 163456  25576 S   4.3  0.5   5:28.13 chrome
```

Monitoring

- CPU monitoring
- I/O monitoring
- Memory monitoring
- Continuous monitoring

SYSTEM ADMINISTRATION

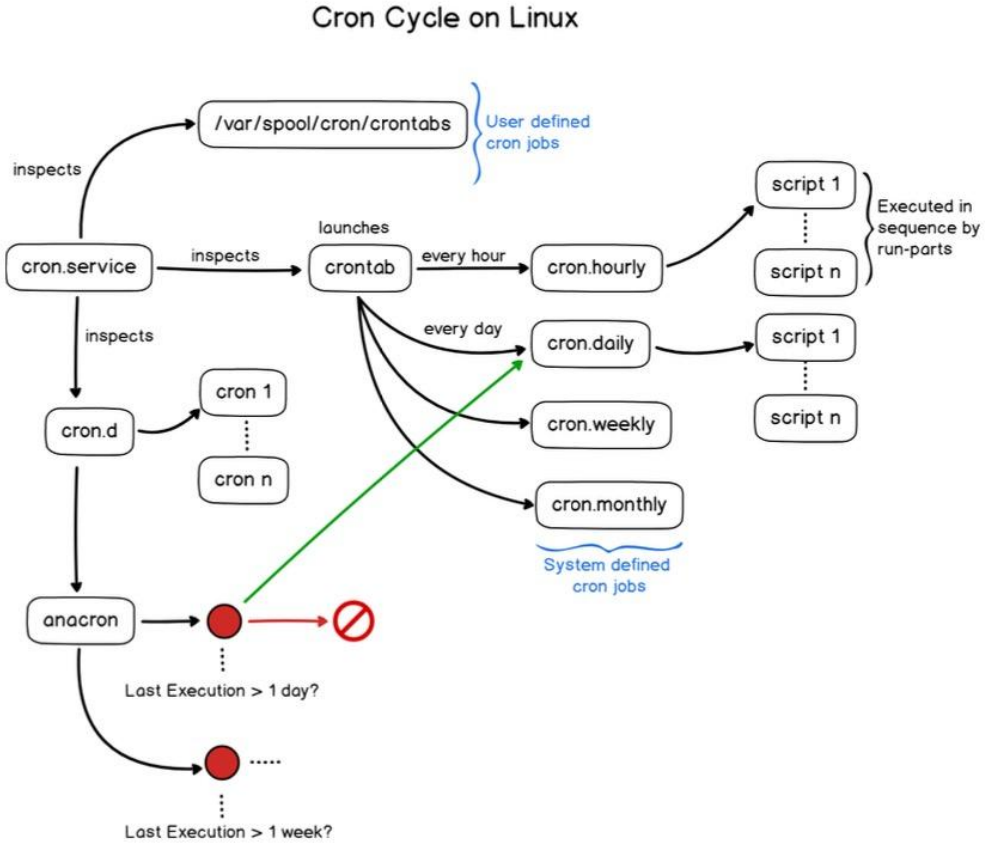
System Administration

- Managing, configuring and maintaining Linux OS
- Ensure its optimal performance, reliability and security.

Cron Jobs

- Cron is a scheduling daemon that executes tasks at specified intervals.
- Crontab is a text file that specifies the schedule of cron jobs.

Cron Jobs


Cron Cycle on Linux
/var/spool/cron/crontabs
User defined cron jobs
inspects
launches
cron.service
inspects
crontab
every hour
cron.hourly
script 1
script n
Executed in sequence by run-parts
inspects
every day
cron.daily
script 1
script n
cron 1
cron.d
cron n
cron.weekly
cron.monthly
System defined cron jobs
anacron
Last Execution > 1 day?
Last Execution > 1 week?

Logging

- Collect, store and analyze log files generated by OS.
- Managing and maintaining the system.

Logging

- System logging
- General logging
- Kernel logging
- Authentication logging

Services

- Long-running process performs tasks in the background - daemon.
- System functionality and centralized management.
- Maintaining system’s performance.

System V

Path file: /etc/inittab

- Starts and stops processes sequentially with scripts
- Pros: Easy to solve dependencies.
- Cons: Underperformance.

System V

State of machine

- 0 - shut down
- 1 - single user mode
- 2 - multi-user mode - no networking
- 3 - multi-user mode - networking
- 4 - unused
- 5 - multi-user mode - networking - GUI
- 6 - reboot

Upstart

Manages system services and responds to events.

- Load the job configurations from /etc/init
- Event -> run jobs.
- Continue until it completes all the jobs.

systemd

Path file: /usr/lib/systemd

Default init system recently - flexible and robust.

- Load configuration files /usr/lib/systemd/system
- Determines its boot goal
- Dependencies of the boot target and activates them

systemd

- Service units - starting and stopping, end in .service
- Mount units - mount filesystems, end in .mount
- Target units - group together other units, end in .target

.service files

Provide information about how systemd should manage a particular service.

- [Unit]: contains general information about the unit.
- [Service]: includes information about the service itself.
- [Install]: how the unit should be enabled or disabled.

.target files

- Define the system state or run levels by specifying dependencies between units.
- Synchronization point in the boot process and groups other units together.

SHELL SCRIPTING & AUTOMATION

Shell Scripting & Automation

- Text file contains commands executed by a shell.
- Automate commands for various tasks.

Shebang (optional)

- Appear at the first line #! + path to the shell interpreter.
- Tell the system which interpreter used to execute.

Variable

- variable_name=value (note: no space between)
- To get the value, add $ before.

```
1 #!/bin/bash
2 # This is comment - A simple variable example - hello.sh
3 greeting=Hello
4 name=Tux
5 echo $greeting $name
```

Arithmetic operations

- var=$((expression)) (note: no space between)

```
1 #!/bin/bash
2 # file sum.sh
3
4 var=$((1+2))
5 echo $var
```

Passing argument

```
#!/bin/bash
# file name.sh

echo $0
echo Your name is $1
```

```
$ ./name.sh student
./name.sh
Your name is student
```

Read from keyboard

```
#!/bin/bash
# file input.sh

read -p "Enter a number: " a
read -p "Enter a number: " b

var=$((a+b))
echo $var
```

```
$ ./input.sh

Enter a number: 1
Enter a number: 2
3
```

Comparison

Use these operators to compare 2 statements:

Operators	Description
`=`	`$a -eq $b`
`>=`	`$a -ge $b`
`>`	`$a -gt $b`
`<=`	`$a -le $b`
`<`	`$a -lt $b`
`!=`	`$a -ne $b`

If structure

```
1 if [ conditions ]
2 then
3   commands
4 fi
```

For loop

```
1 #!/bin/bash
2 for i in {1..5}
3 do
4   echo $i
5 done
```

While loop

```
#!/bin/bash
i=1
while [[ $i -le 5 ]] ; do
  echo "$i"
  (( i += 1 ))
done
```

Save results from a command

- variable_name=’<command>’
- variable_name=$(command)

Functions

```
functionName(){
  first command
  second command
  ...
}
```

NETWORKING & SECURITY

Network

- Some of network configuration files:

	Configuration file
Host	/etc/host
DNS	/etc/resolv.conf
Name service switch configuration file	/etc/nsswitch.conf

- Using **netstat** to monitor network status

Network

Work with Network by command:

- Check your machine IP
- Get hostname information
- Show, add, delete with routing table
- Check ARP cache
- Request for an IP with DHCP server
- ....

Security - User

Three kinds of user:

- Root
- Regular
- Service

Security - User

Some action with user

- Add, remove user
- Change between user
- User info in **/etc/passwd**, password in **/etc/shadow**

Security - User authentication

Some plugin authentication model (PAM):

- LDAP
- SASL
- NIS, NIS+
- SSL
- ....

We can find out their configuration file in **/etc/pam.d**

Security - Group

Group is a set of many user:

- A user can be belong to many group
- Each of group has defined by ID
- Add, delete user from group
- Group info can be found at **/etc/group**

Security - Group

Group is a set of many user:

- A user can be belong to many group
- Each of group has defined by ID
- Add, delete user from group
- Group info can be found at **/etc/group**

Security - File permissions

3 actions with a file:

- Read: ability to read/see the contents of file
- Write: ability to change the contents of file
- Execute: ability to run the file
- Each action corresponds to a bit (4-r,2-w,1-x)

Security - File permissions

3 types of user with file: owner, group, other

- Each type has different file permissions
- Can change file permissions
- Can change the owner of file

Security - Firewall management

Firewall management:

- Can be installed as a service on machine
- Work in zone concept
- With each zone, agree/disagree connection

Security - Firewall management

Firewall management:

- Open/close a port
- Forwarding a port
- Change zone
- Accept IP to connect to a port

VAGRANT

Vagrant - What and Why

- OS installation with specific settings
- Easy installation for multi VM
- Easy to control (by Vagrantfile)

Vagrant - How to use

How many ways for user to use Vagrant

- Default by Vagrant Cloud
- Customize Vagrantfile

Vagrant - How to use

Default by using Vagrant Cloud - The simplest way

- Go to the directory you want
- Just run 2 commands

```
1  $ vagrant init + <distro>
2  $ vagrant up
```

<distro> can be find out here

Vagrant - How to use

Customize your Vagrantfile - You have what you want

- Vagrantfile can be written in some programming languages (Ruby in default)
- Structure of Vagrantfile:

```
Vagrant.configure("2") do |config|
  config.vm.box = "<vm_url>"
  <some of customize here>
end
```

Vagrant - How to use

What we can customize?

- Using **network** to configuration network
- Using **provider** to customize CPU, RAM, GUI for VM
- Using **synced_folder** to interact with host filesystem
- Using **provision** to add shell script

Vagrant - How to use

Some others common commands for using Vagrant

Command	Meaning
vagrant halt <name>	shut down the virtual machine
vagrant ssh <name>	log in to the virtual machine
vagrant destroy <name>	uninstall the virtual machine
vagrant reload <name>	reload the virtual machine
vagrant suspend <name>	suspend the guest machine
vagrant resume <name>	resume to the guest machine after suspend
vagrant status <name>	get the status of guest machine
vagrant port	get the mapped port from virtual machine to host
vagrant global-status	get the status of all vagrant machine

DEMO

Demo

- Shell script programming
- Vagrant

THANK YOU

Q&A