



VIT-AP
UNIVERSITY

SECURE CODING LAB

SLOT: L23+L24

NAME: CH. KARTHIKEYA VARMA

REGD.NO: 19BCD7138

PROF: SIBI CHAKKRAVARTHY

Print Hello World

```
#!/usr/bin/env python3

print("hello world")
```

Make file executable

```
chmod +x ./hello.py
```

Run at 04:00 of first Monday of every month:

```
0 4 1 * *
```

Defragmentation

My system uses ext4 filesystem which doesn't require defragmentation however here are my log rotation and backup scripts that I use in production:

log rotation:

```
#!/bin/sh

find /var/log/nginx -type f -regex ".*[1-9][1-9].gz$" -delete
```

Run with cron every week at 03:00

```
0 3 * * 1
```

Backup

```
#!/bin/sh

backup_files="/home /var/spool/mail /etc /root /boot /opt /srv"

dest="/mnt/backup"

day=$(date +%A)
hostname=$(hostname -s)
archive_file="$hostname-$day.tgz"

echo "Backing up $backup_files to $dest/$archive_file"
date
echo

tar czf $dest/$archive_file $backup_files

echo
echo "Backup finished"
date
```

Run with cron every day at 01:00

```
0 1 * * *
```

File permissions

In UNIX systems, file permissions are scoped under three perspectives:

- current logged in user/ user executing the command
- groups
- everyone else on the system

These rules, however, don't apply to root. root enjoys full privileges on all files in their system.

Analysing file permissions:

```
→ /tmp stat file
File: file
Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 31h/49d Inode: 56385        Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/aravinh)   Gid: ( 1000/aravinh)
Access: 2021-03-08 09:17:58.376874893 +0530
Modify: 2021-03-08 09:17:58.376874893 +0530
Change: 2021-03-08 09:17:58.376874893 +0530
Birth: -
```

File `file` has permissions `644` or `-rw-r--r--`.

- `r` is for readable
- `w` is for writable
- `x` is for executable

First `-` position indicates whether the said file is a directory or not. In our case, it wasn't.

The next three positions are the privileges available to the current user `aravinh`. User `aravinh` can read, write but not execute.

Analysing file permissions:

```
→ /tmp stat file
File: file
Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 31h/49d Inode: 56385        Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/aravinth)   Gid: ( 1000/aravinth)
Access: 2021-03-08 09:17:58.376874893 +0530
Modify: 2021-03-08 09:17:58.376874893 +0530
Change: 2021-03-08 09:17:58.376874893 +0530
Birth: -
```

File `file` has permissions `644` or `-rw-r--r--`.

- `r` is for readable
- `w` is for writable
- `x` is for executable

First `-` position indicates weather the said file is a directory or not. In our case, it wasn't.

The next three positions are the privileges available to the current user `aravinth`. User `aravinth` can read, write but not execute.

The middle three positions are the privileges available to the group `aravinth`. They can read but not write and execute.

The last three positions are the privileges available to the rest of the users in the system. They can read but not write and execute.

Modifying file permissions:

There's a shorthand available for modifying file permissions. Each of the abilities(`r` , `w` and `x`) are assigned a unique number as follows:

- `r` is 4
- `w` is 2
- `x` is 1

And to choose privileges that a scope can have is calculated by picking relevant values adding them. Maximum privileges that can be had is `rwX` which is 7 and minimum is 0.

```
chmod xxx filename
```

where xxx is a three-digit number representing the three scopes discussed above.

Making a file readable only to the owner

```
chmod 400 file
```

Making a file readable only to the owner and group

```
chmod 440 file
```

Making a file readable executable only to the owner and readable only to the group

```
chmod 540 file
```

Hiding a file

In UNIX systems, hiding a file can be done by simple prefixing the filename with a period(`.`)

```
move file .file
```

hidden files can be seen with:

```
ls -a
```


Power configuration

Monitoring

`powertop` is a Swiss army knife for monitoring power usage. It produces outputs in multiple formats. When not specified, it defaults to using a pager.

I'm using HTML for the purpose of this experiment

```
sudo powertop -r -t=2
```



System Information

PowerTOP 2.13 ran at Mon Mar 8 09:36:34 2021

Version

Kernel Version Linux version 5.11.2-arch1-1

System Name CFLCitgo_CFSV1.05

CPU 12 Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz

OS Information Arch Linux

Summary CPU Idle CPU Frequency Software Info Device Info Tuning ACPI All

Target: 1 units System: 3318.6 wakeups CPU: 72.3% usage GPU: 0 ops/s GFX: 0 wakeups VFS: 0 ops/s

Top 10 Power Consumers

Usage	Events/s	Category	Description	PW Estimate
0.4%	1505.6	Timer	tick_sched_timer	5.95 W
4.8%	615.8	Work	handle_update	2.51 W
57.9%	273.7	Process	[PID 1602097] powertop -f -t=2	1.98 W
1.1%	136.9	Process	[PID 1563563] kps/teams-for-linux/teams-for-linux -type=renderer --no-sandbox --enable-features=SharedArrayBuffer --service-p436 mW	
2.0%	102.6	Process	[PID 1563566] kps/teams-for-linux/teams-for-linux -type=renderer --no-sandbox --enable-features=SharedArrayBuffer --service-p436 mW	
0.7%	68.4	Process	[PID 1563570] kps/teams-for-linux/teams-for-linux	281 mW
0.3%	68.4	Process	[PID 1563562] kps/teams-for-linux/teams-for-linux -type=renderer --no-sandbox --enable-features=SharedArrayBuffer --service-p274 mW	
0.4%	68.4	Process	[PID 571903] telegram-desktop --	277 mW
0.3%	68.4	Process	[PID 1563565] kps/teams-for-linux/teams-for-linux	275 mW
0.3%	68.4	Process	[PID 1563562] kps/teams-for-linux/teams-for-linux -type=renderer --no-sandbox --enable-features=SharedArrayBuffer --service-p274 mW	

Wake status of the devices

Description	Script
Wake-on-lan status for device wlp0s200	echo "enabled" > "/sys/class/net/wlp0s200/device/power/wakeup"
Wake-on-lan status for device enp0s0	echo "enabled" > "/sys/class/net/enp0s0/device/power/wakeup"
Wake status for USB device 1-14	echo "enabled" > "/sys/bus/usb/devices/1-14/power/wakeup"
Wake status for USB device usb3	echo "enabled" > "/sys/bus/usb/devices/usb3/power/wakeup"
Wake status for USB device usb1	echo "enabled" > "/sys/bus/usb/devices/usb1/power/wakeup"
Wake status for USB device usb4	echo "enabled" > "/sys/bus/usb/devices/usb4/power/wakeup"
Wake status for USB device usb2	echo "enabled" > "/sys/bus/usb/devices/usb2/power/wakeup"
Wake status for USB device 1-4	echo "enabled" > "/sys/bus/usb/devices/1-4/power/wakeup"
Wake status for USB device 1-4.2	echo "enabled" > "/sys/bus/usb/devices/1-4.2/power/wakeup"

Last wake:

```
→ lab4 git:(master) X journalctl --list-boots
-8 9aaf595ec9924ca793266ed9f75e7460 Wed 2021-02-24 19:52:02 IST-Wed 2021-02-24 21:24:14 IST
-7 7efe1a161df04b1c9da2836f9e564956 Wed 2021-02-24 21:24:25 IST-Wed 2021-02-24 21:29:23 IST
-6 e920683c587349549bca8a3edc869a00 Thu 2021-02-25 09:26:34 IST-Thu 2021-02-25 19:59:47 IST
-5 860d6c51273a41f597dbcbe579c2081d Thu 2021-02-25 19:59:59 IST-Thu 2021-02-25 20:20:18 IST
-4 a0fb8d68060b405a965a0ad3df8601b4 Thu 2021-02-25 20:31:02 IST-Sat 2021-02-27 14:12:18 IST
-3 f8f71b9951eb44de93b62ec42b585d3d Sat 2021-02-27 14:12:28 IST-Mon 2021-03-01 04:27:20 IST
-2 c5d77071e6d4491ea8b29d0f2ae40f90 Mon 2021-03-01 04:27:35 IST-Wed 2021-03-03 15:39:29 IST
-1 8be869311a8b4c36b04c8020cb0ccaa2 Wed 2021-03-03 15:39:40 IST-Wed 2021-03-03 15:41:07 IST
0 015f47e7ca4d48c49ce1d90161e0f660 Wed 2021-03-03 15:41:18 IST-Mon 2021-03-08 09:38:53 IST
```

Hibernate:

```
systemctl hibernate
```

Screenlock

I use a custom built desktop environment so I'm already using a script for locking my screen

```
#!/bin/bash

LOCKSCREEN="/home/aravinth/Pictures/Wallpapers/lockscreen.jpg"

i3lock --nofork -i $LOCKSCREEN --ignore-empty-password -f -k \
-t --pass-media-keys --pass-screen-keys \
--greetertext="Welcome, Batman." --greeterpos="980:900"
```