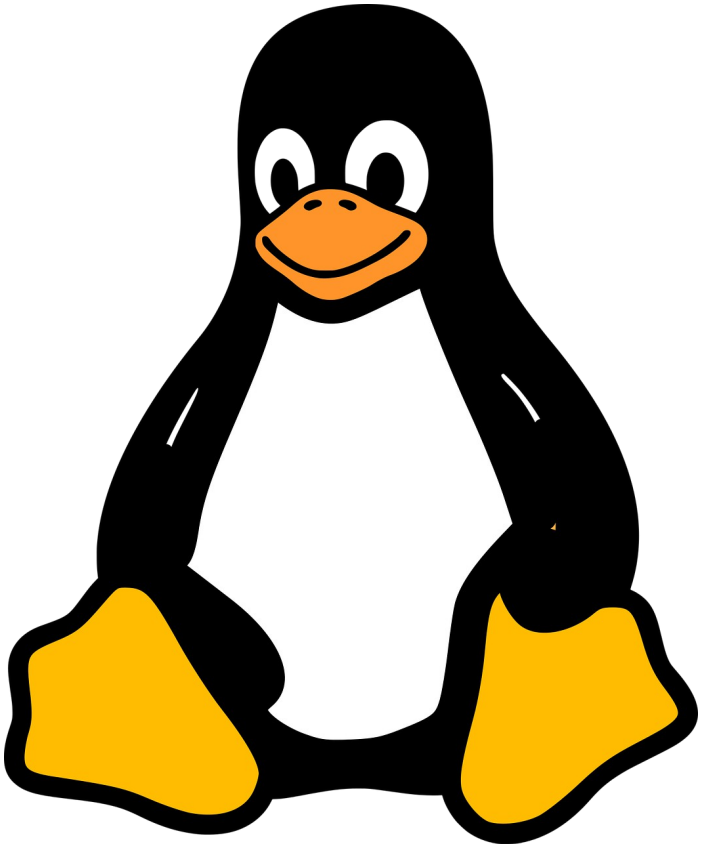
HackerFrogs Afterschool
Linux Basics Pt 5 – Wrap Up

Class:
Linux OS Operations

Workshop Number:
AS-LIN-05

Document Version:
1.75

Special Requirements:
none

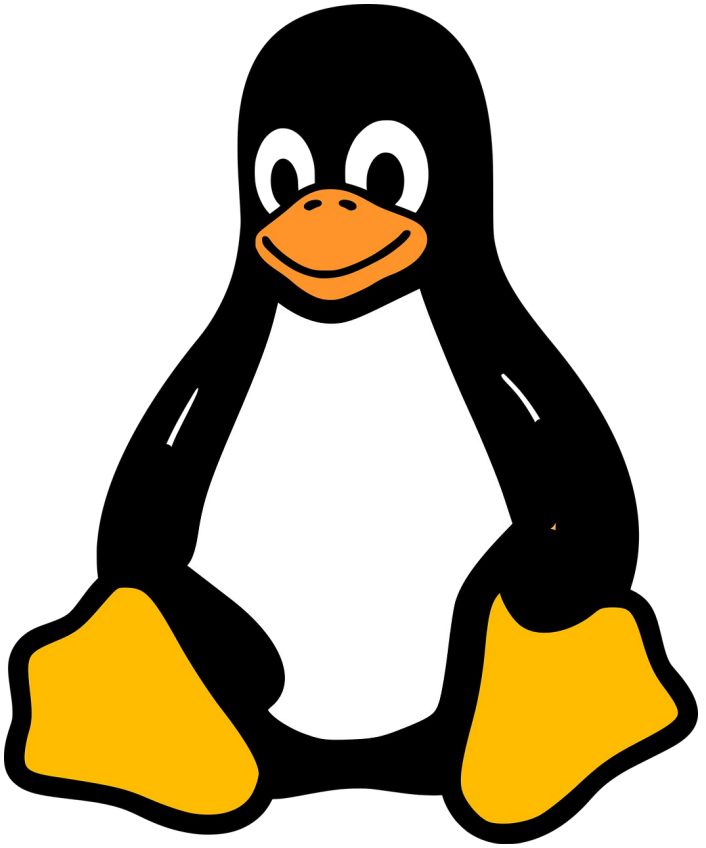


What We Learned In The Previous Workshop

Hey there HackerFrogs!

This is the fifth intro to Linux OS Operations workshop.

In the previous workshop we learned about the following Linux commands:



The Help Flag

```
[root@localhost ~]# cat --help
Usage: cat [OPTION]... [FILE]...
Concatenate FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

  -A, --show-all           equivalent to -vET
  -b, --number-nonblank    number nonempty output lines, overrides -n
```

The **--help** flag can be used as part of any command to return a relatively brief explanation as to how to use the command.

Cp Command

```
[root@localhost ~]# echo 'we copied test.txt to another directory!'> test.txt
[root@localhost ~]# cp test.txt test
[root@localhost ~]# cd test && cat test.txt
we copied test.txt to another directory!
```

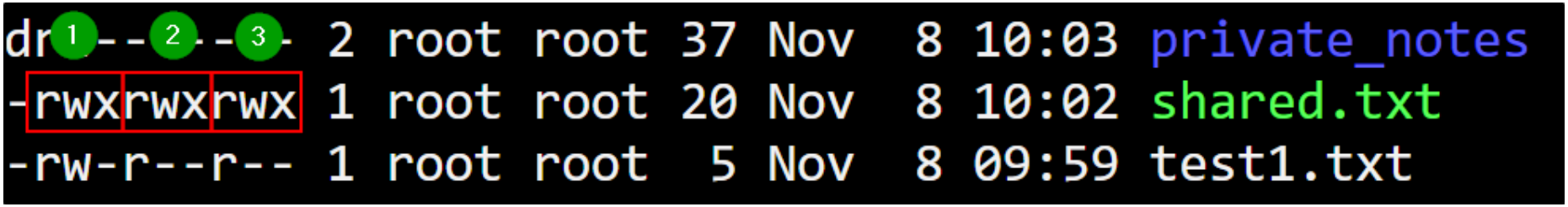
The **cp** command copies files from one directory to another. The first argument is the file to be copied, and the second argument is the directory the file is to be copied to.

Mv Command

```
[root@localhost ~]# ls
test1.txt
[root@localhost ~]# cat test2.txt
cat: test2.txt: No such file or directory
[root@localhost ~]# mv test1.txt test2.txt
[root@localhost ~]# cat test2.txt
test text
```

The **mv** command is used in a similar fashion to the **cp** command, except the original file will not remain in its original directory. This command is also used for modifying the names of files.

Linux File Permissions



Each file in the Linux filesystem has different (r)ead, (w)rite, and e(x)ecute permissions, depending on whether a user is (1), the file owner, (2), part of a certain group, or (3), any other user.

Noteworthy Linux File Directories

/etc where system files (user passwords) are stored

/var where log files and database files are stored

/root typically the folder with the most secure access

/tmp all files here are deleted on a regular basis

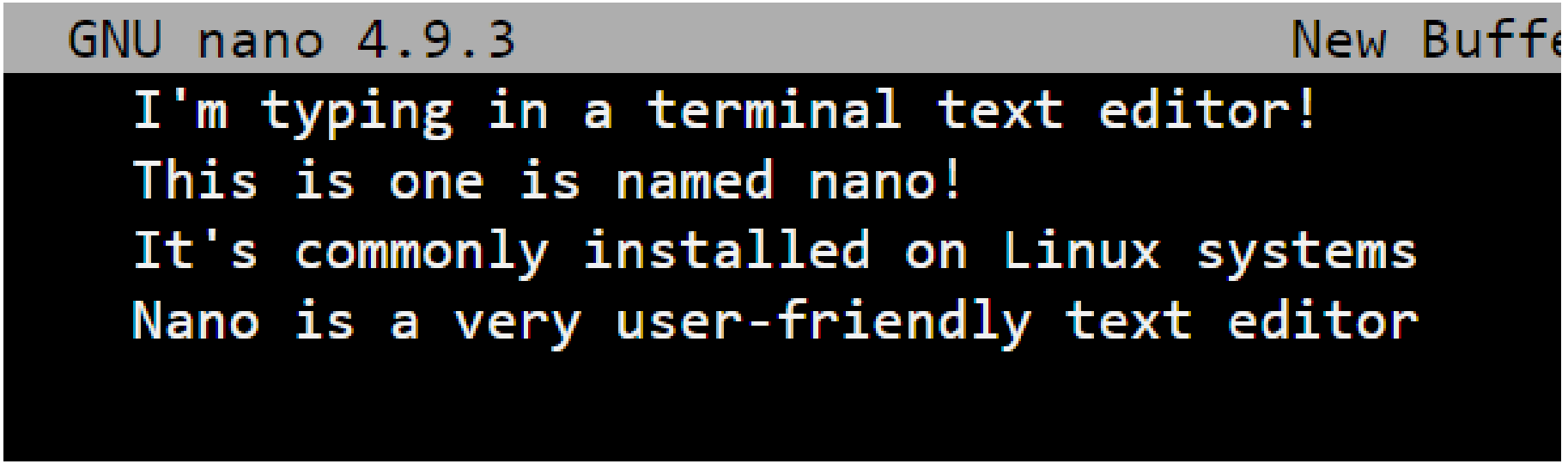
/home where individual user account files are stored

Let's Access a Linux Console

We can access a web-based Linux console for our educational needs at the following URL:

https://bellard.org/jslinux/index.html

Terminal Text Editors

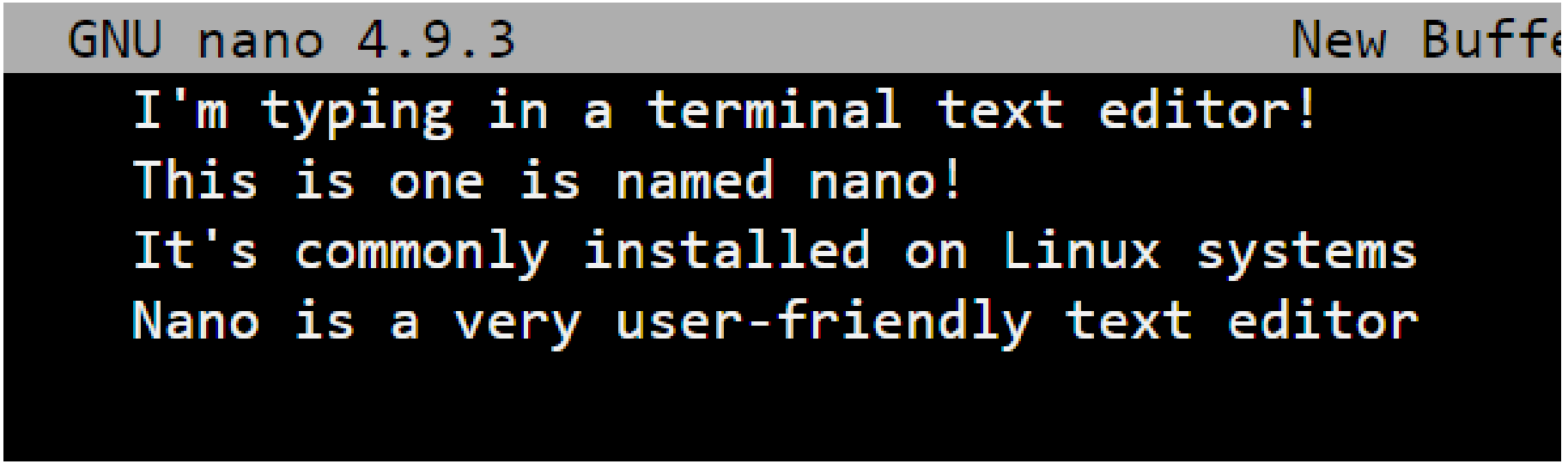


From time to time, we'll need to write new files or modify existing ones. To do so, we'll need to use Linux text editors.

Terminal Text Editors

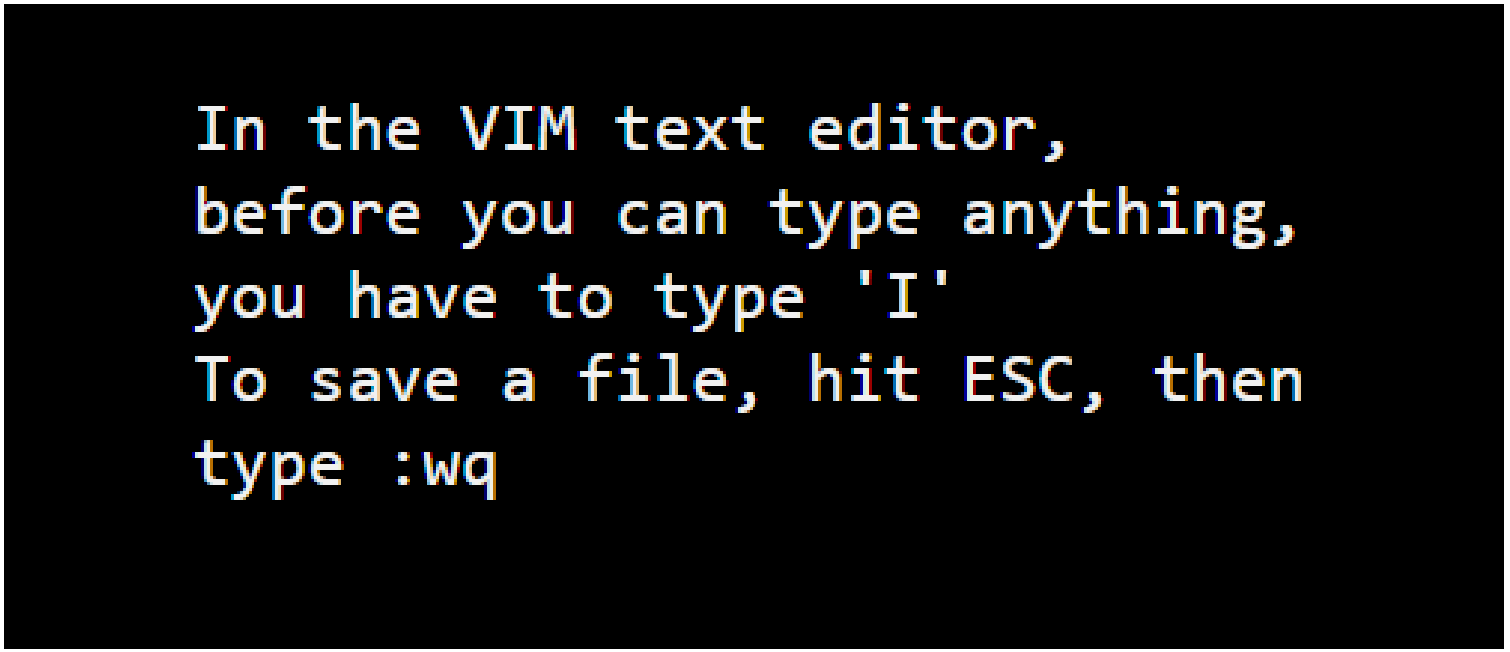
Two commonly installed text editors on Linux systems are Nano and Vim.

Nano Text Editor



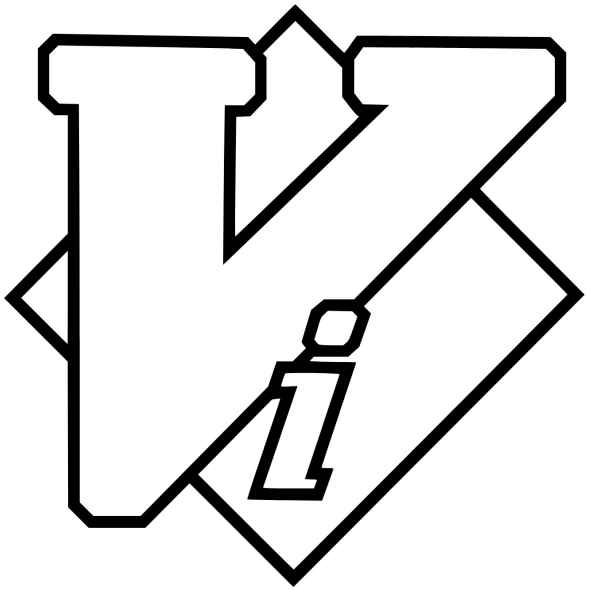
Nano is the more straightforward and user-friendly of the two text editors.

Vim Text Editor



Vim is a more complicated text editor, but it is favored by Linux power users, who can make use of its advanced features.

Vi Text Editor



On older systems, both Vim and Nano may be absent, but Vim's predesessor, Vi, may be present. It's basic functions are very similar to Vim

Downloading Files - Wget

Downloading public files is easy with **wget**, all we need is the link to the file

Managing Processes: PS

```
[root@localhost ~]# ps
  PID TTY          TIME CMD
   47 hvc0     00:00:00 sh
  102 hvc0     00:00:00 ps
```

Ongoing programs running on Linux are called **processes**, and the current processes on the system can be retrieved using the **PS** command.

Managing Processes: Kill

```
[root@localhost ~]# ps
  PID TTY          TIME CMD
   47 hvc0     00:00:01 sh
  224 hvc0     00:00:23 find
  226 hvc0     00:00:03 find
  227 hvc0     00:00:00 ps
[root@localhost ~]# kill 226
```

If we know the PID of a process, we can remove it with the **kill** command.

Checking if a Program Exists – Which Command

```
[root@localhost ~]# which python
/bin/python
[root@localhost ~]#
```

The which command can be used to check where a program is located. If there is no output, it's possible this program is not installed

Installing New Programs

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install tor
```

```
After this operation, 17.8 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

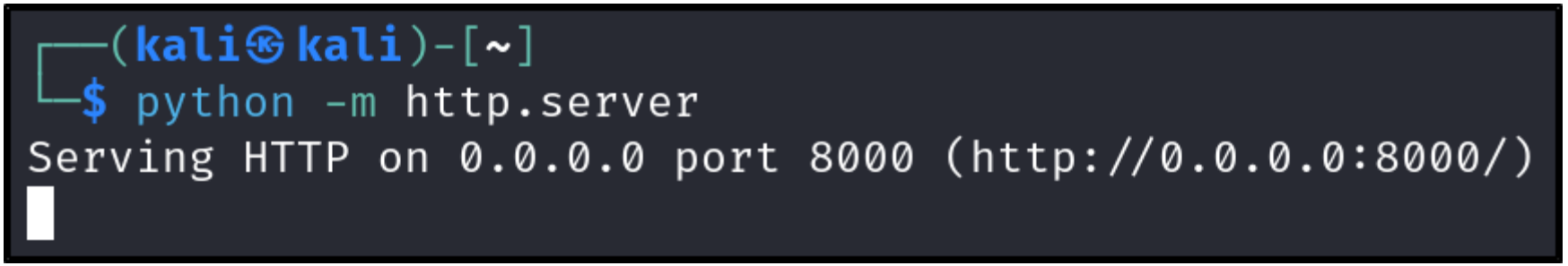
Although different Linux distros use different methods of installing programs, Debian-based distros, like Ubuntu and Kali Linux, use the **apt** program to install new programs.

Networking Commands - Ifconfig

```
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.5.13.77  netmask 255.255.0.0  broadcast 10.5.255.255
        ether 02:ca:3d:74:e2:6e  txqueuelen 1000  (Ethernet)
        RX packets 99  bytes 8640 (8.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2  bytes 684 (684.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The **ifconfig** command is used to return networking information for the system, including its IP address and MAC address.

Creating a Simple HTTP Server



We can use Python to serve files to the internet with the simple http server module. Although it's should be confined to local network use.

Networking Commands - Netstat

```
┌──(kali㉿kali)-[/tmp]
└─$ netstat -tulpn
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
 State       PID/Program name
tcp        0      0 0.0.0.0:8000            0.0.0.0:*
 LISTEN      73956/python
```

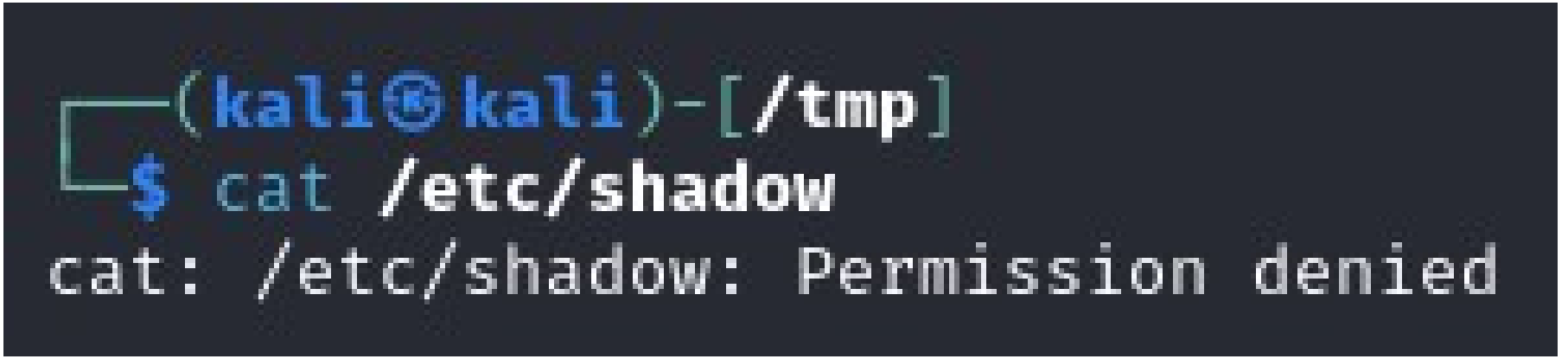
The netstat command is used to retrieve network connections.

Sudo – The Super Command

```
[root@localhost ~]# sudo --help
sudo - execute a command as another user
```

The **sudo** command can be added to any other command, and it allows the command to be executed in the context of the root user.

Sudo – The Super Command



Sudo needs to be used whenever we try to access or modify systems files or directories.

Sudo – The Super Command

```
└─$ sudo -l
Matching Defaults entries for kali on kali:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bi
n\:/sbin\:/bin,
    use_pty

User kali may run the following commands on kali:
    (ALL : ALL) ALL
```

Generally, only system administrator accounts should be allowed to use **sudo** with all commands.

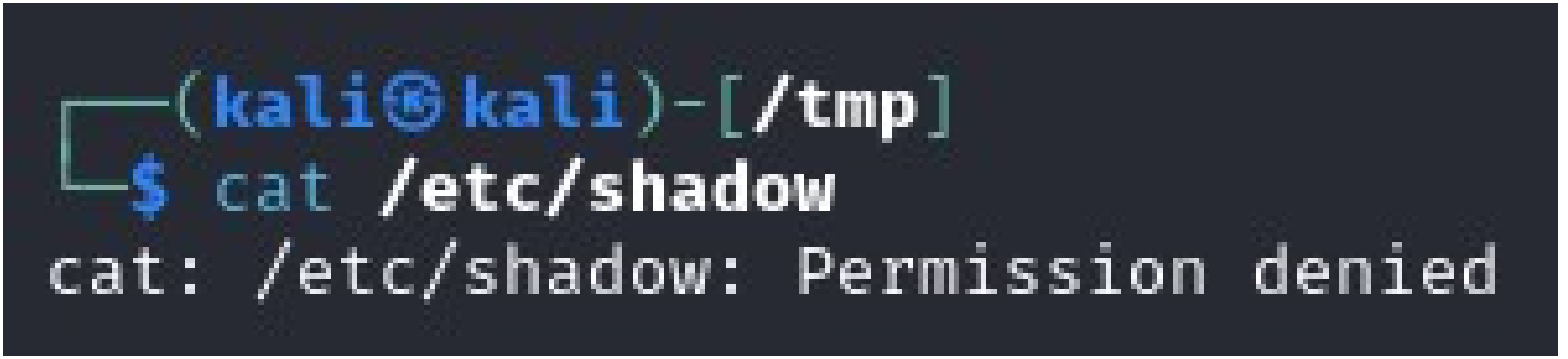
Sudo – The Super Command

```
└─$ sudo -l
Matching Defaults entries for kali on kali:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bi
n\:/sbin\:/bin,
    use_pty

User kali may run the following commands on kali:
    (ALL : ALL) ALL
```

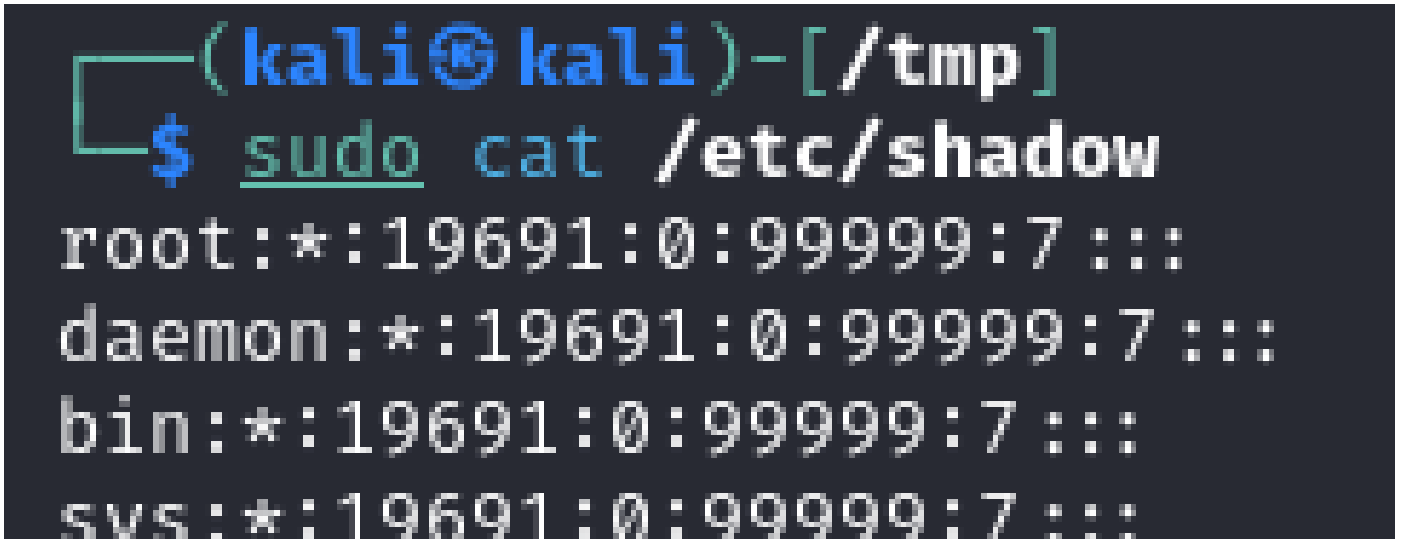
But some systems have users who are able to use **sudo** with a specific command for some reason or another.

Sudo – The Super Command



If you ever try to run a command as a regular user, and you see the permission denied message...

Sudo – The Super Command



You can usually retry that command with sudo to execute it successfully.

Sudo – The Super Command

```
┌──(kali㉿kali)-[/tmp]
└─$ rm -rf /
rm: it is dangerous to operate recursively on '/'
rm: use --no-preserve-root to override this failsafe
```

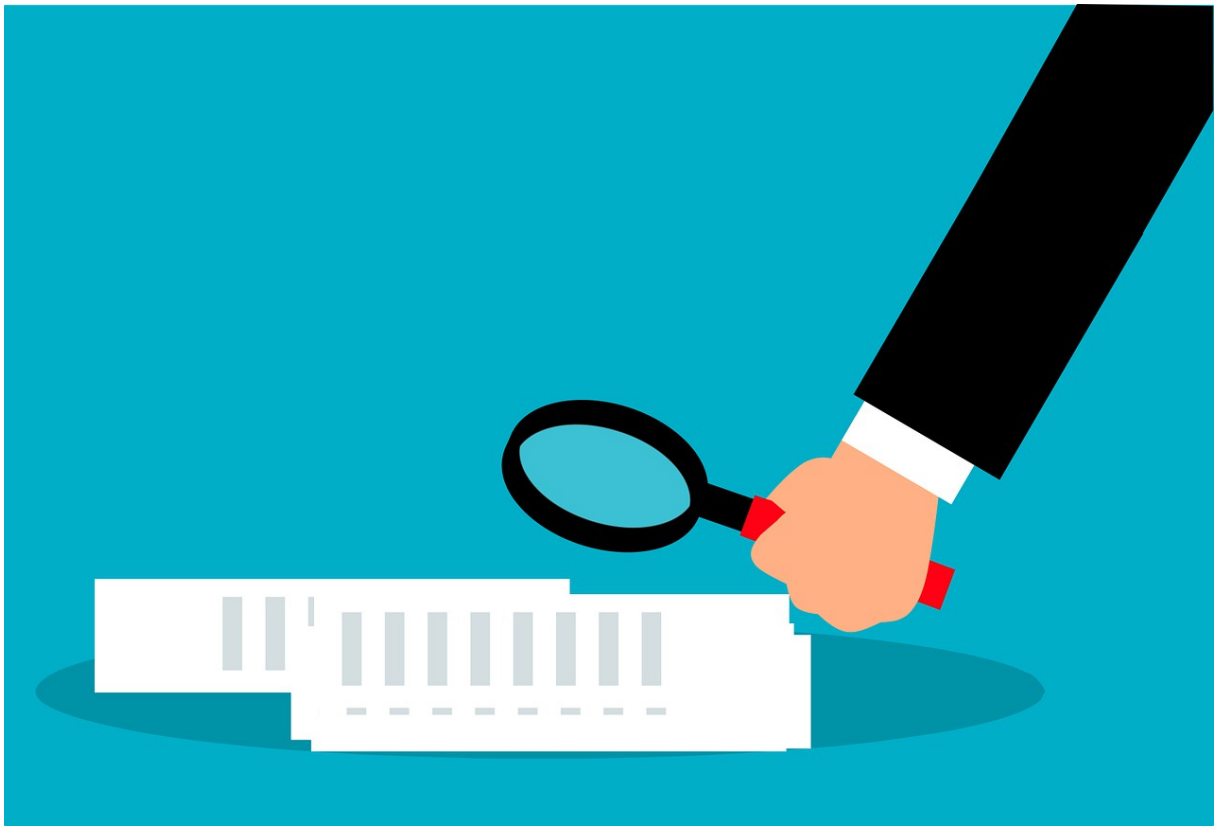
If we need to use sudo, we should always try to understand why we need to use it...

Sudo – The Super Command

```
┌──(kali㉿kali)-[/tmp]
└─$ rm -rf /
rm: it is dangerous to operate recursively on '/'
rm: use --no-preserve-root to override this failsafe
```

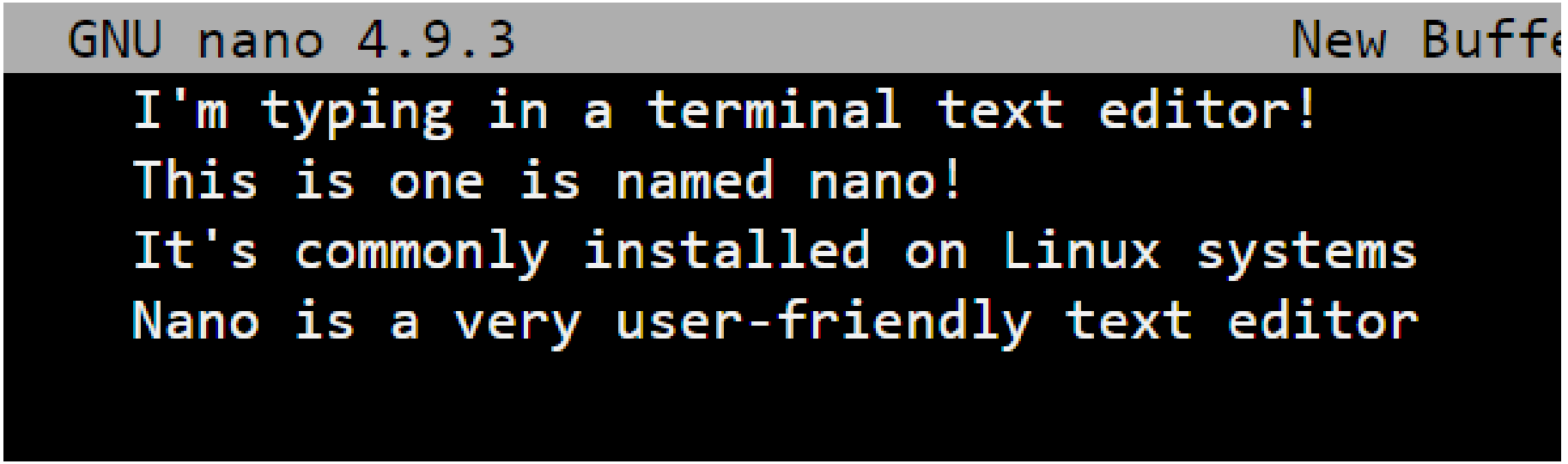
Since it means we're interacting with sensitive files or directories.

Summary



Let's review the Linux commands we learned in this workshop:

Terminal Text Editors



We learned about terminal text editors, which are used for word processing in CLI environments.

Terminal Text Editors

Two commonly installed text editors on Linux systems are Nano and Vim.

Managing Processes: PS

```
[root@localhost ~]# ps
  PID TTY          TIME CMD
   47 hvc0     00:00:00 sh
  102 hvc0     00:00:00 ps
```

We learned how to manage processes on the system with the **ps** command.

Managing Processes: Kill

```
[root@localhost ~]# ps
  PID TTY          TIME CMD
   47 hvc0     00:00:01 sh
  224 hvc0     00:00:23 find
  226 hvc0     00:00:03 find
  227 hvc0     00:00:00 ps
[root@localhost ~]# kill 226
```

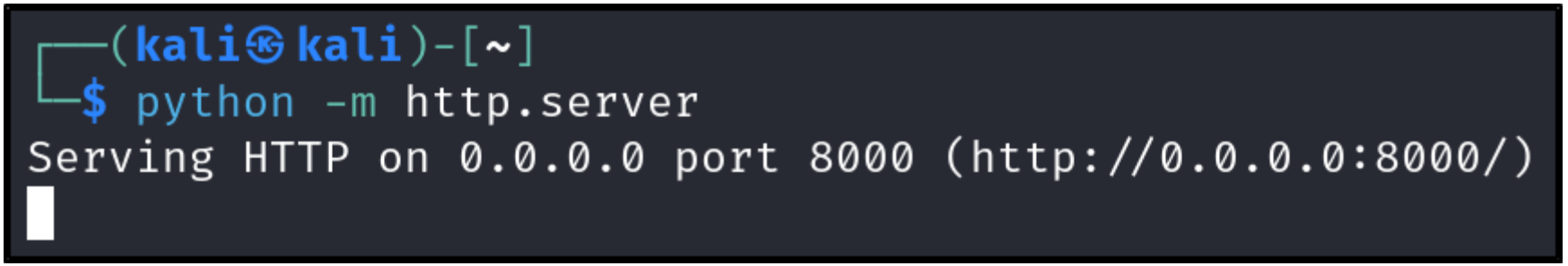
After looking up processes, you can stop them with the **kill** command.

Networking Commands - Ifconfig

```
[root@localhost ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.5.13.77  netmask 255.255.0.0  broadcast 10.5.255.255
        ether 02:ca:3d:74:e2:6e  txqueuelen 1000  (Ethernet)
        RX packets 99  bytes 8640 (8.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2  bytes 684 (684.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

We used the **ifconfig** command to look up our system's networking configuration.

Creating a Simple HTTP Server



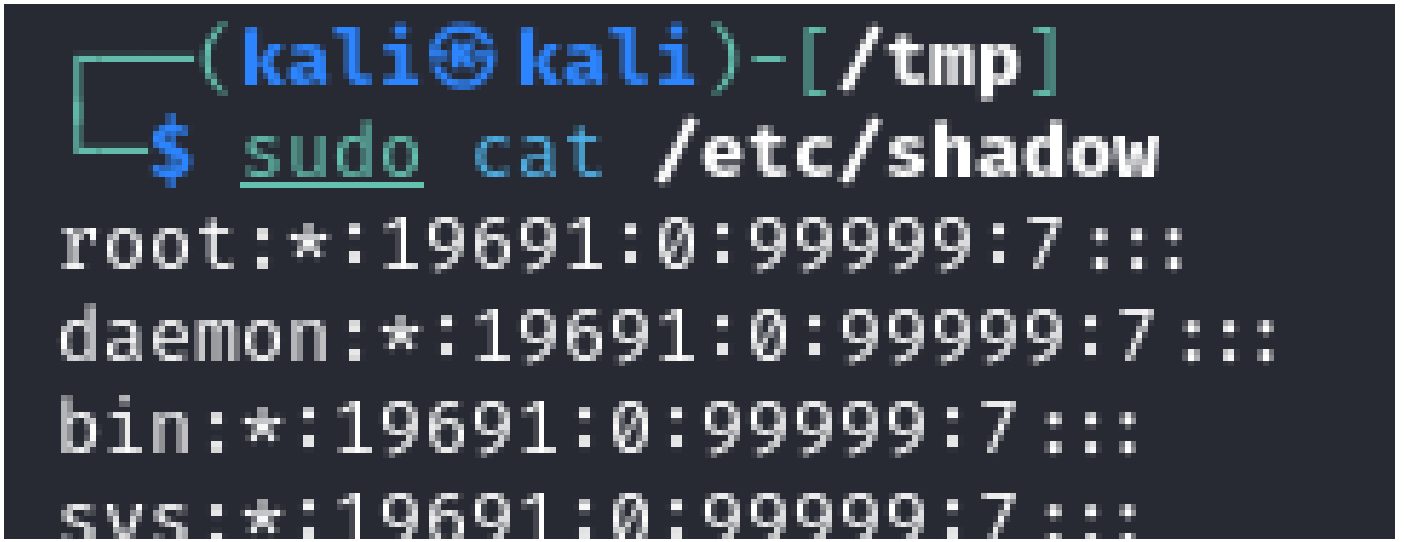
We used Python to create a simple HTTP server to serve files.

Networking Commands - Netstat

```
┌──(kali㉿kali)-[/tmp]
└─$ netstat -tulpn
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
 State       PID/Program name
tcp        0      0 0.0.0.0:8000            0.0.0.0:*
 LISTEN      73956/python
```

We used the **netstat** command to determine our system's active network connections.

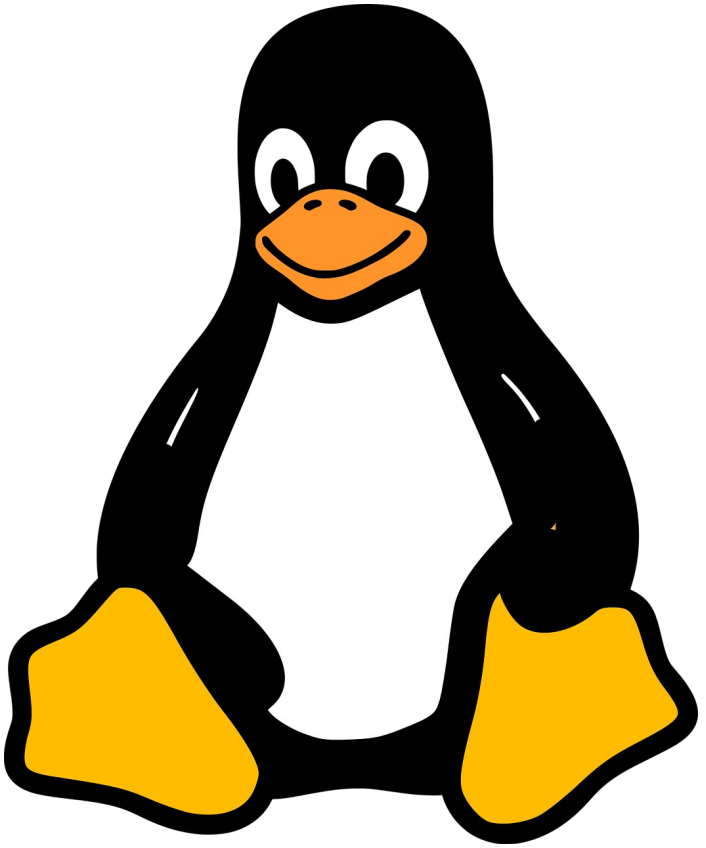
Sudo – The Super Command



And we learned about the **sudo** command, which lets us run commands as the super user (root).

What's Next?

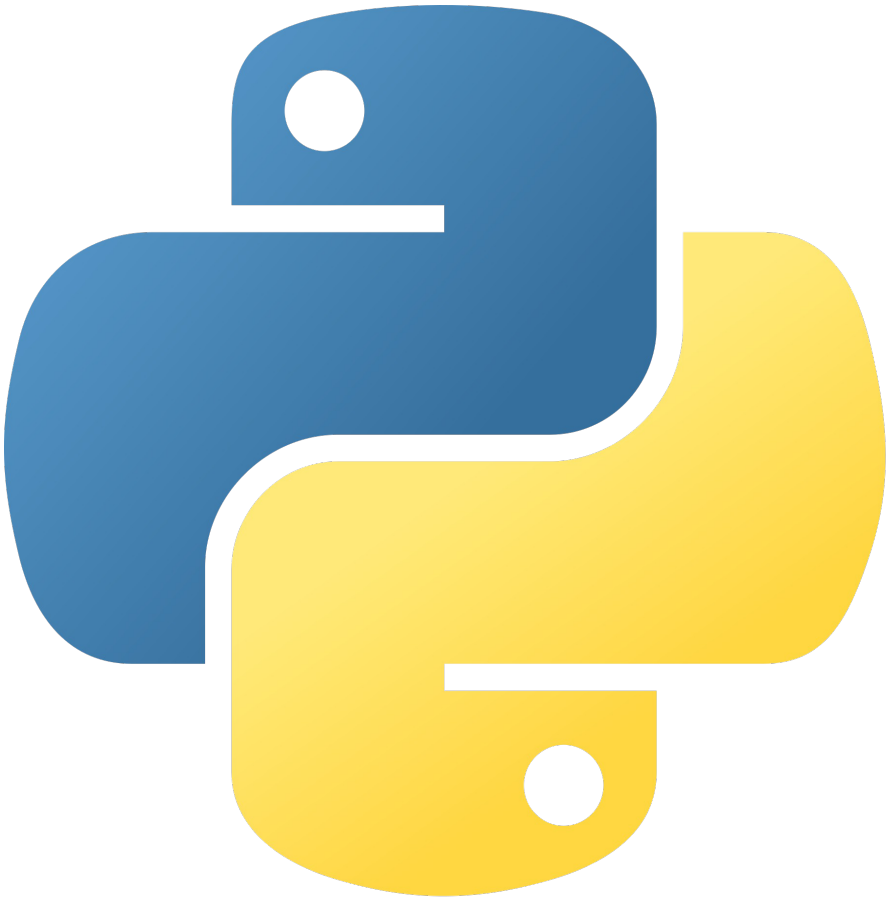
This is the final Linux OS operations workshop.
I hope you've had fun learning about this popular alternative OS to the likes of Windows and macOS.



What's Next?

In the next workshop, we'll be starting a new subject:

Programming



Extra Credit

Looking for more study material on this workshop's topics?

See this video's description for links to supplemental documents and exercises!

Until Next Time, HackerFrogs!