Service & Process Management

First lets understand what a service is

Services are background processes that run independently of user interaction, providing specific functionality or performing system tasks. They are managed by the init system, typically systemd, and can be started, stopped, and managed using commands like `systemctl`.

Services are defined in `.service` files, which contain details like the service's name, description, and the command to run. These files are typically located in `/etc/systemd/system` or `/lib/systemd/system`.

- To start a service

```
sudo systemctl start tor.service
```

- To check the status of the service

```
sudo systemctl status tor.service
```

- To restart the service

```
sudo systemctl restart tor.service
```

- To stop the service.

```
sudo systemctl stop tor.service
```

In addition to starting and stopping services on demand, you can also configure services to start automatically when the system boots up. This is known as enabling a service.

To enable a service:

```
sudo systemctl enable tor.service
```

To disable a service:

```
sudo systemctl disable tor.service
```

Process Management

Processes are instances of running programs that execute in memory. Each process has its own memory space, and they can communicate with each other using inter-process communication (IPC) mechanisms. Basically, Processes can be classified into two types:

- **Foreground processes**: These processes run in the foreground and interact with the user directly. Examples include command-line tools like `ls` and `grep`.
- **Background processes**: These processes run in the background and do not interact with the user directly. Examples include system services like `sshd` and `httpd`.
- To view the process, we can us the ps command.

```
ps
```

But this does not give enough information as we have not used any flags. Now if we use the -aux flag.

```
ps aux
```

It now displays a detailed list of all processes, including the process ID (PID), CPU usage, and memory usage.

- We can also use HTOP for this which is much for graphical.

Killing the process:

So, if we want to kill a process like we do in Windows using Task Manager. We can also do it in linux with the kill command. For that, we only need a PIP or Process name.

- To Terminates the process with the specified PID.

```
kill PID
```

- We can also Force the process to terminate immediately, without allowing it to clean up.

```
kill -9 PID
```

- If we want to kill all the process with a certain name we can kill them using killall command.

```
killall <process_name>
```

Backgrounding a process

We can use the Ampersand (&) symbol to make a process run in background.

```
sleep 100&
```

Now the sleep command will run in background for 100 seconds as we have added ampersand behind it and by that time, we can use our terminal like before.

- To check for all the background jobs, we can use the jobs command.

```
jobs
```

- If we want to send a background job to foreground, we can use its Job ID to do so.

```
fg <job ID>
```

- Same can be happen in reverse if we want a foreground job to go in background again.

```
bg <job_ID>
```
