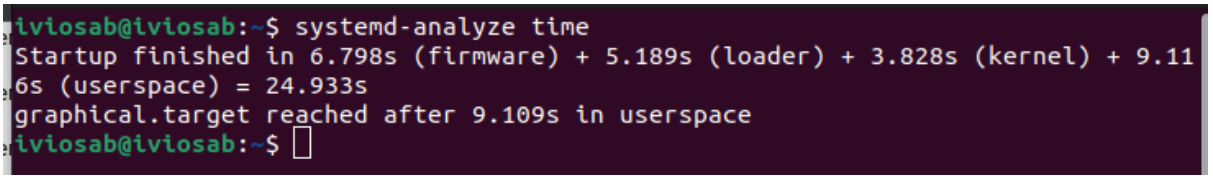
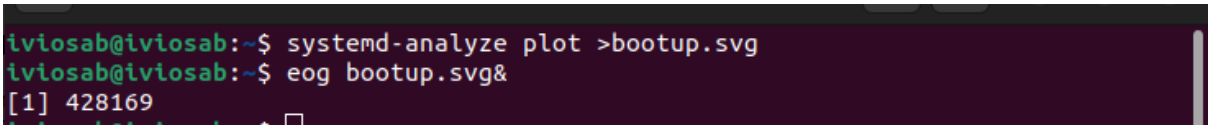
Name: Mosab Fathy Ramadan Mohamed
Group: B20-SD-01
Lab 9: Systemd

1. Show the following boot-up performance statistics on your system:
    - Time spent in the kernel space before the user space was reached.
    - Show an SVG image that contains services that have been started, and how long it took for them to initialize.

Answer:
systemd-analyze time: This command prints the time spent in the kernel before userspace has been reached, the time spent in the initrd before normal system userspace has been reached, and the time normal system userspace took to initialize. Note that these measurements simply measure the time passed up to the point where all system services have been spawned, but not necessarily until they fully finished initialization or the disk is idle.



systemd-analyze plot: This command prints an SVG graphic detailing which system services have been started at what time, highlighting the time they spent on initialization.



Note: since the svg image is so large I'll attach it to the submission instead of showing it in the report.

2. Take the systemd unit `graphical.target` as your starting point, start tracing backwards using only the `Requires` variable. At what systemd unit do you reach a dead end where there is no more `Requires` variable?
    - Provide brief explanation for each of the systemd units you encounter while performing this trace.
    - The unit at this dead end `Wants` some systemd units. Why does it want these units?

    Show screenshots of every step as you trace.

Answer:

.target - a unit configuration file which is used for grouping units via dependencies and as standardized synchronization points during start-up.

```
iviosab@iviosab:~$ sudo cat /usr/lib/systemd/system/graphical.target | grep 'Requires'
Requires=multi-user.target
```

A special target unit for setting up a graphical login screen. This pulls in multi-user.target.

```
iviosab@iviosab:~$ sudo cat /usr/lib/systemd/system/multi-user.target | grep 'Requires'
Requires=basic.target
```

A special target unit for setting up a multi-user system (non-graphical). This is pulled in by graphical.target.

```
iviosab@iviosab:~$ sudo cat /usr/lib/systemd/system/basic.target | grep 'Requires'
Requires=sysinit.target
RequiresMountsFor=/var /var/tmp
```

A special target unit covering basic boot-up.

```
iviosab@iviosab:~$ sudo cat /usr/lib/systemd/system/sysinit.target | grep 'Requires'
```

Sysinit.target does not require anything

systemd automatically adds dependencies of the types Requires= and After= for this target unit to all services (except for those with DefaultDependencies=no).

```
iviosab@iviosab:~$ sudo cat /usr/lib/systemd/system/sysinit.target | grep 'Wants'
Wants=local-fs.target swap.target
iviosab@iviosab:~$
```

Sysinit.target wants

- Local-fs.target: automatically adds dependencies of type Before= to all mount units that refer to local mount points for this target unit. In addition, it adds dependencies of type Wants= to this target unit for those mounts listed in /etc/fstab that have the auto mount option set.
- Swap.target: Similar to local-fs.target, but for swap partitions and swap files.

This target pulls in the services required for system initialization. System services pulled in by this target should declare DefaultDependencies=no and specify all their dependencies manually, including access to anything more than a read only root filesystem.

3. Create a simple web server in bash that shows the following: system uptime, inode usage, current memory, disk space usage statistics, and the last 15 lines of `/var/log/syslog`.
    - The required information should be queried from the server everytime a user opens or refreshes the page.
    - You do not need to save the results anywhere. Users only need live updates when the server is visited.
    - The results should be displayed on a single page in an orderly manner that is easy to read.
    - Create a systemd service on your system to run this script (web server). Show how you can start your new service, and configure it to run after system reboot.
    - Your systemd service should restart the web server if the web server crashes or is killed.
    - This service is allowed to use a maximum of 15% of the CPU and 256MB memory.

    > Show all steps taken, and all unit files created in your report.
    > At the end of this task, you must have at least one bash script, one service file, and one slice file all working together to achieve the objectives.

Answer:

```
#!/bin/bash
while true;
do echo -e "HTTP/1.1 200 OK\n\nUptime(since): $(uptime -s)\nInode Usage: $(df -i
 | grep '/dev/nvme0n1p6' | awk '{print $5}')\nMemory Usage: $(free | grep "Mem"
| awk '{print ($3/$2)*100 "%"}')\nDisk Usage: $(df / | awk '{print $5}'| tail -n
 1)\nLast 15 lines of /var/log/syslog:\n$(tail -15 /var/log/syslog)" \
  | nc -l -k -p 8080 -q 1;
done
~
~
```

```
iviosab@iviosab:~$ sudo vim /lib/systemd/system/myscript.service
```

```
[Unit]
Description=My custom web service to show system resources

[Service]
ExecStart=/usr/bin/script.sh
Slice=myslice.slice
Restart=on-failure
RestartSec=5s

[Install]
WantedBy=multi-user.target
~
```

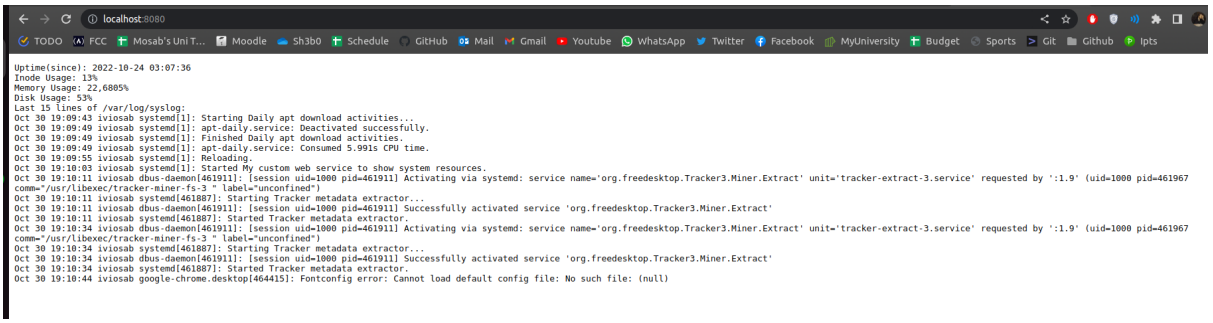
```
iviosab@iviosab:~$ sudo vim /etc/systemd/system/myslice.slice
```

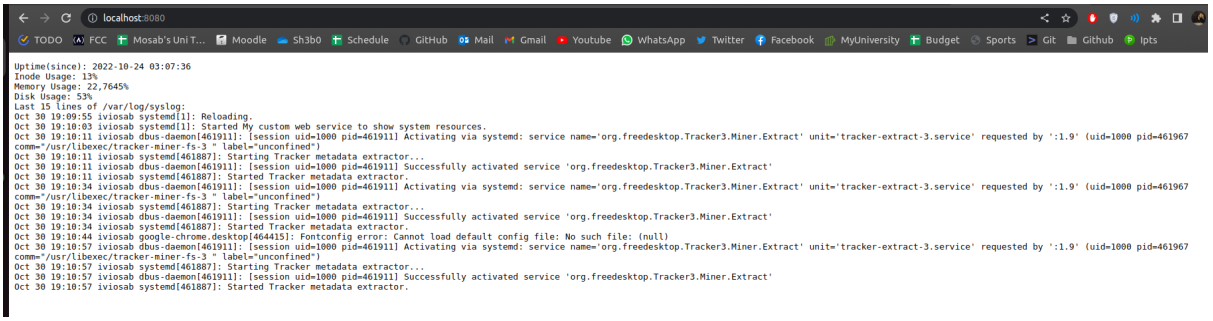
```
[Unit]
Description=Custom systemd slice for exercise 4 in SNA Lab 9
Before=slices.target

[Slice]
MemoryAccounting=true
CPUAccounting=true
MemoryMax=256M
CPUQuota=15%
~
```

```
iviosab@iviosab:~$ sudo systemctl daemon-reload
iviosab@iviosab:~$ sudo systemctl enable myscript.service
iviosab@iviosab:~$ sudo systemctl restart myscript.service
iviosab@iviosab:~$ sudo systemctl status myscript.service
● myscript.service - My custom web service to show system resources
     Loaded: loaded (/lib/systemd/system/myscript.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2022-10-30 19:10:03 MSK; 20s ago
   Main PID: 469227 (script.sh)
      Tasks: 2 (limit: 19006)
     Memory: 592.0K (available: 255.3M)
        CPU: 10ms
     CGroup: /myslice.slice/myscript.service
             ├─469227 /bin/bash /usr/bin/script.sh
             └─469229 nc -l -k -p 8080 -q 1

окт 30 19:10:03 iviosab systemd[1]: Started My custom web service to show system resources.
iviosab@iviosab:~$
```





4. Create a systemd service that will update your package sources list from the repository.
    - The service should update the package source list five minutes after booting, and then every day after that.
    - The schedule of the execution should be done with only systemd.

Answer:

```
iviosab@iviosab:~$ sudo vim /etc/systemd/system/updater.timer
```

```
[Unit]
Description=Run apt update periodically

[Timer]
OnBootSec=5min
OnUnitActiveSec=1d

[Install]
WantedBy=timers.target
~
```

```
iviosab@iviosab:~$ sudo vim /etc/systemd/system/updater.service
```

```
[Unit]
Description=Service for updating packages

[Service]
ExecStart=/bin/bash apt update

[Install]
WantedBy=multi-user.target
~
```

5. Create a custom target in `/etc/systemd/system/<your_target>.target`.
    - Add a description of the target file.
    - Create a directory `/etc/systemd/system/<your_target>.wants/`
    - Create sylinks to additional services you wish to enable in this new directory. It should be a symlink to services from `/usr/lib/systemd/system/` that you wish to enable.

Answer:

```
iviosab@iviosab:/$ sudo vim /etc/systemd/system/foo.target
```

```
[Unit]
Description=Foobar boot target
Requires=multi-user.target
Wants=uuidd.service
Conflicts=rescue.service rescue.target
After=multi-user.target rescue.service rescue.target
AllowIsolate=yes
```

```
iviosab@iviosab:/$ sudo mkdir /etc/systemd/system/foo.target.wants
```

This is the directory where you will link the services you create/require for your target. It is equivalent to the Wants= option in the unit file.

```
iviosab@iviosab:/$ sudo ln -s /usr/lib/systemd/system/uuidd.service /etc/systemd
/system/foo.target.wants/uuidd.service
```

I used the same service i used in my unit folder because i'm not doing this on a VM and i'm relatively scared to do user random stuff(even though this service is also a random service)