Student: Pham Dinh Quy

Student ID: 313540040

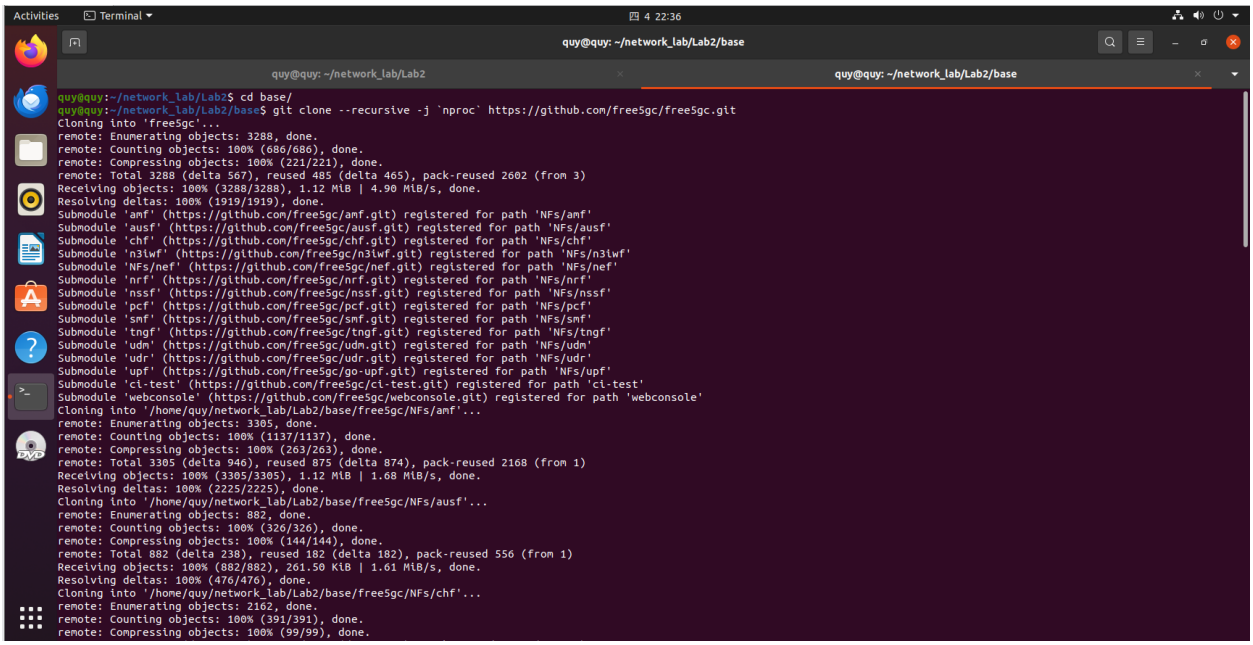
Lab2 Report

A. Building the system

1. Clone project from github repository

git clone https://github.com/NYCU-CSCS20047-PoCaWN/Lab2.git

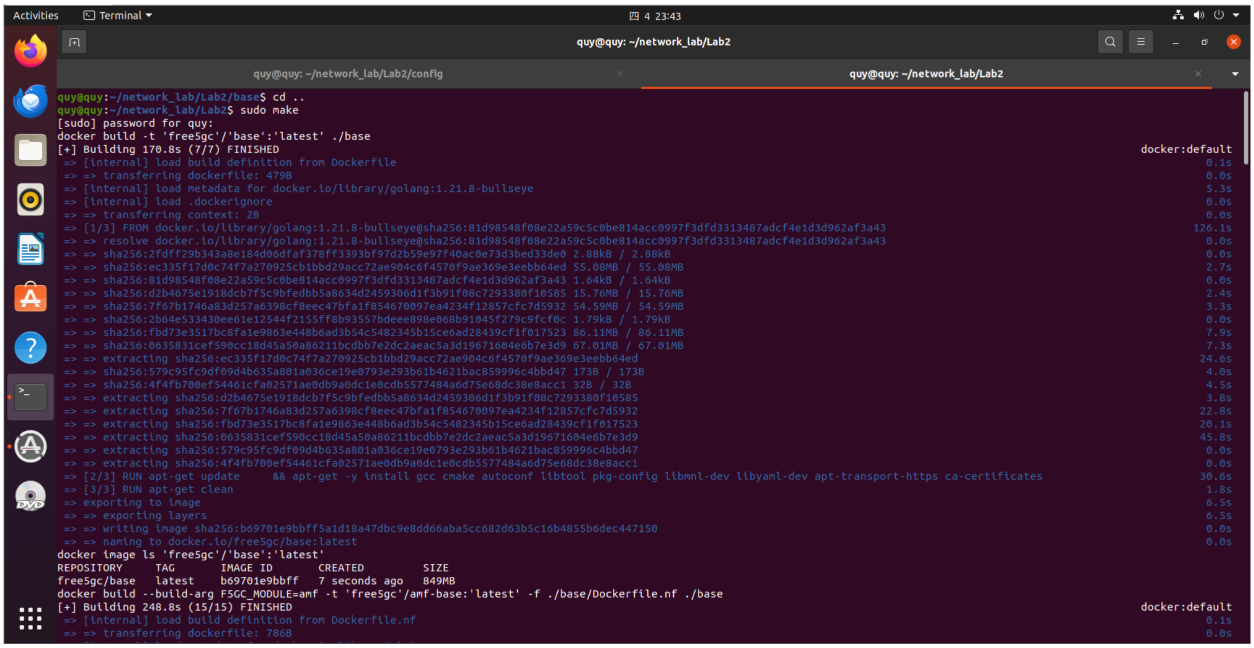
git clone --recursive -j `nproc` https://github.com/free5gc/free5gc.git



2. Build image from local

Using <sudo make>:

This command automatically compile and build programs according to instructions in Makefile. The Makefile includes commands that build a base Docker image for Free5GC, with steps such as installing dependencies, copying the Free5GC source code, and compiling the required modules.., preparing for deploy the docker image in the next step.



```
quy@quy:~/network_lab/Lab2/base$ cd ..
quy@quy:~/network_lab/Lab2$ sudo make
[sudo] password for quy:
docker build -t 'free5gc'/'base':'latest' ./base
[+] Building 170.8s (7/7) FINISHED                                                                                     docker:default
 => [internal] load build definition from Dockerfile                                                                           0.1s
 => => transferring dockerfile: 479B                                                                                           0.0s
 => [internal] load metadata for docker.io/library/golang:1.21.8-bullseye                                                      5.3s
 => [internal] load .dockerignore                                                                                              0.0s
 => => transferring context: 2B                                                                                                0.0s
 => [1/3] FROM docker.io/library/golang:1.21.8-bullseye@sha256:81d98548f08e22a59c5c0be814acc0997f3dfd3313487adcf4e1d3d962af3a43  126.1s
 => => resolve docker.io/library/golang:1.21.8-bullseye@sha256:81d98548f08e22a59c5c0be814acc0997f3dfd3313487adcf4e1d3d962af3a43  0.0s
 => => sha256:2fdff29b343a8e184d06dfaf378ff3393bf97d2b59e97f40ac0e73d3bed33de0 2.88kB / 2.88kB                                 0.0s
 => => sha256:ec335f17d0c74f7a270925cb1bbd29acc72ae904c6f4570f9ae369e3eebb64ed 55.08MB / 55.08MB                               2.7s
 => => sha256:81d98548f08e22a59c5c0be814acc0997f3dfd3313487adcf4e1d3d962af3a43 1.64kB / 1.64kB                                 0.0s
 => => sha256:d2b4675e1918dcb7f5c9bfedbb5a8634d2459306d1f3b91f08c7293380f10585 15.76MB / 15.76MB                               2.4s
 => => sha256:7f67b1746a83d257a6398cf8eec47bfa1f854670097ea4234f12857cfc7d5932 54.59MB / 54.59MB                               3.3s
 => => sha256:2b64e533430ee61e12544f2155ff8b93557bdeee898e068b91045f279c9fcf0c 1.79kB / 1.79kB                                 0.0s
 => => sha256:fbd73e3517bc8fa1e9863e448b6ad3b54c5482345b15ce6ad28439cf1f017523 86.11MB / 86.11MB                               7.9s
 => => sha256:0635831cef590cc18d45a50a86211bcdbb7e2dc2aeac5a3d19671604e6b7e3d9 67.01MB / 67.01MB                               7.3s
 => => extracting sha256:ec335f17d0c74f7a270925cb1bbd29acc72ae904c6f4570f9ae369e3eebb64ed                                     24.6s
 => => sha256:579c95fc9df09d4b635a801a036ce19e0793e293b61b4621bac859996c4bbd47 173B / 173B                                     4.0s
 => => sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1 32B / 32B                                       4.5s
 => => extracting sha256:d2b4675e1918dcb7f5c9bfedbb5a8634d2459306d1f3b91f08c7293380f10585                                      3.8s
 => => extracting sha256:7f67b1746a83d257a6398cf8eec47bfa1f854670097ea4234f12857cfc7d5932                                     22.8s
 => => extracting sha256:fbd73e3517bc8fa1e9863e448b6ad3b54c5482345b15ce6ad28439cf1f017523                                     20.1s
 => => extracting sha256:0635831cef590cc18d45a50a86211bcdbb7e2dc2aeac5a3d19671604e6b7e3d9                                     45.8s
 => => extracting sha256:579c95fc9df09d4b635a801a036ce19e0793e293b61b4621bac859996c4bbd47                                      0.0s
 => => extracting sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1                                      0.0s
 => [2/3] RUN apt-get update     && apt-get -y install gcc cmake autoconf libtool pkg-config libmnl-dev libyaml-dev apt-transport-https ca-certificates  30.6s
 => [3/3] RUN apt-get clean                                                                                                    1.8s
 => exporting to image                                                                                                         6.5s
 => => exporting layers                                                                                                        6.5s
 => => writing image sha256:b69701e9bbff5a1d18a47dbc9e8dd66aba5cc682d63b5c16b4855b6dec447150                                   0.0s
 => => naming to docker.io/free5gc/base:latest                                                                                 0.0s
docker image ls 'free5gc'/'base':'latest'
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
free5gc/base   latest    b69701e9bbff   7 seconds ago   849MB
docker build --build-arg F5GC_MODULE=amf -t 'free5gc'/amf-base:'latest' -f ./base/Dockerfile.nf ./base
[+] Building 248.8s (15/15) FINISHED                                                                                   docker:default
 => [internal] load build definition from Dockerfile.nf                                                                        0.1s
 => => transferring dockerfile: 786B                                                                                           0.0s
```

```
docker build -t 'free5gc'/webconsole-base:'latest' -f ./base/Dockerfile.nf.webconsole ./base
[+] Building 535.1s (17/17) FINISHED
 => [internal] load build definition from Dockerfile.nf.webconsole
 => => transferring dockerfile: 1.07kB
 => [internal] load metadata for docker.io/free5gc/base:latest
 => [internal] load metadata for docker.io/library/alpine:3.15
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [stage-1 1/7] FROM docker.io/library/alpine:3.15@sha256:19b4bcc4f60e99dd5ebdca0cbce22c503bbcff197549d7e19dab4f22254dc864
 => [my-base 1/4] FROM docker.io/free5gc/base:latest
 => CACHED [stage-1 2/7] WORKDIR /free5gc
 => [stage-1 3/7] RUN mkdir -p cert/ public
 => [internal] load build context
 => => transferring context: 153.11kB
 => CACHED [my-base 2/4] COPY free5gc/ /go/src/free5gc/
 => [my-base 3/4] RUN curl -fsSL https://deb.nodesource.com/setup_20.x | bash -     && apt update     && apt install nodejs -y
 => [my-base 4/4] RUN cd /go/src/free5gc     && apt-get update     && apt-get -y install sudo     && make webconsole
 => [stage-1 4/7] COPY --from=my-base /go/src/free5gc/webconsole/bin/webconsole ./webui
 => [stage-1 5/7] COPY --from=my-base /go/src/free5gc/webconsole/public ./public
 => [stage-1 6/7] COPY --from=my-base /go/src/free5gc/config/* ./config/
 => [stage-1 7/7] COPY --from=my-base /go/src/free5gc/cert/* ./cert/
 => exporting to image
 => => exporting layers
 => => writing image sha256:c61a7adf027919a1ddeeaefb2076162f4be43bf7cd93b924e90b15e231df958d
 => => naming to docker.io/free5gc/webconsole-base:latest
quy@quy:~/network_lab/Lab2$ 
```

Using docker build:

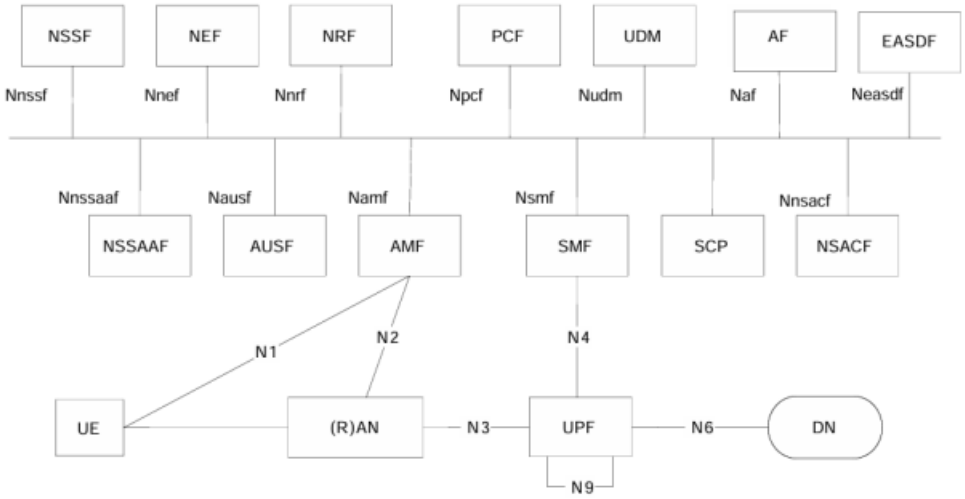
The docker compose build command will build all Docker images from the services defined in docker-compose file such as amf, nrf, webui, ueransim,… and network layer such as N2, N4, N6,…The docker compose file will also point to the Docker file located in corresponding directory to get the data.

```
quy@quy:~/network_lab/Lab2$ sudo docker compose -f 313540040.yaml build
[sudo] password for quy:
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 2079.8s (147/147) FINISHED                                                                                    docker:default
 => [free5gc-n6gw internal] load build definition from Dockerfile                                                                    0.1s
 => => transferring dockerfile: 383B                                                                                                 0.1s
 => [free5gc-nrf internal] load build definition from Dockerfile                                                                     0.1s
 => => transferring dockerfile: 642B                                                                                                 0.0s
 => [free5gc-upf internal] load build definition from Dockerfile                                                                     0.1s
 => => transferring dockerfile: 936B                                                                                                 0.0s
 => [free5gc-smf internal] load metadata for docker.io/bitnami/minideb:bullseye                                                     6.9s
 => [free5gc-upf internal] load metadata for docker.io/free5gc/upf-base:latest                                                      0.0s
 => [free5gc-upf internal] load metadata for docker.io/free5gc/base:latest                                                          0.0s
 => [free5gc-nrf internal] load metadata for docker.io/free5gc/nrf-base:latest                                                      0.0s
 => [free5gc-chf internal] load metadata for docker.io/library/alpine:3.15                                                          5.6s
 => [free5gc-nrf internal] load .dockerignore                                                                                        0.0s
 => => transferring context: 2B                                                                                                      0.0s
 => [free5gc-chf stage-1 1/6] FROM docker.io/library/alpine:3.15@sha256:19b4bcc4f60e99dd5ebdca0cbce22c503bbcff197549d7e19dab4f22254dc864  0.0s
 => [free5gc-nrf builder 1/1] FROM docker.io/free5gc/nrf-base:latest                                                                 0.4s
 => CACHED [free5gc-chf stage-1 2/6] WORKDIR /free5gc                                                                                0.0s
 => [free5gc-nrf stage-1 3/6] RUN mkdir -p config/ log/ cert/                                                                        3.5s
 => [free5gc-n6gw internal] load .dockerignore                                                                                       0.2s
 => => transferring context: 2B                                                                                                      0.0s
 => [free5gc-upf internal] load .dockerignore                                                                                        0.4s
 => => transferring context: 2B                                                                                                      0.1s
 => CACHED [ueransim stage-1 1/7] FROM docker.io/bitnami/minideb:bullseye@sha256:d2e21765766b393fc43f250e632fb8b07722ed94126d94f91b37687d4e4dbfe8  45.4s
 => => resolve docker.io/bitnami/minideb:bullseye@sha256:d2e21765766b393fc43f250e632fb8b07722ed94126d94f91b37687d4e4dbfe8                0.2s
 => => sha256:d2e21765766b393fc43f250e632fb8b07722ed94126d94f91b37687d4e4dbfe8 741B / 741B                                           0.0s
 => => sha256:4619e708f8bc7e6c7571cf41cf64b24fa1fadd7cfa73742ea3a3ff97f61e0df8 528B / 528B                                           0.0s
 => => sha256:93af0c23b703296f7ba0725f47c66423a8b5b1e0da491fc638ee23a31f88a0df 910B / 910B                                           0.0s
 => => sha256:17be88885e9d6aba819673d2186d56a8384ee935c7e846ae628a51baeac05264 30.96MB / 30.96MB                                     2.9s
 => => extracting sha256:17be88885e9d6aba819673d2186d56a8384ee935c7e846ae628a51baeac05264                                           38.0s
 => CACHED [free5gc-upf my-base 1/2] FROM docker.io/free5gc/base:latest                                                              0.1s
 => [free5gc-upf builder 1/1] FROM docker.io/free5gc/upf-base:latest                                                                 0.8s
 => [free5gc-upf my-base 2/2] RUN git clone https://github.com/free5gc/go-gtp5gnl.git && mkdir "go-gtp5gnl/bin" &&     cd "go-gtp5gnl/cmd/gogtp5g-tunnel" &&  go build -o "/  161.8s
 => [free5gc-nrf stage-1 4/6] COPY --from=builder /free5gc/nrf ./                                                                    0.5s
 => [free5gc-nrf stage-1 5/6] COPY --from=builder /free5gc/cert/nrf.pem ./cert/                                                      0.4s
 => [free5gc-nrf stage-1 6/6] COPY --from=builder /free5gc/cert/nrf.key ./cert/                                                      0.3s
 => [free5gc-nrf] exporting to image                                                                                                 0.7s
 => => exporting layers                                                                                                              0.6s
 => => writing image sha256:89a1a57c7f3da323fba69887b3d658fd54f5d2b195592d5bcc447ec8a0d5c413                                         0.0s
```

```
 => => extracting sha256:8c51bf5a7bd72d9b6eb9e7417b838d37e0786ad44f0833863bae467f7239a059                                            0.0s
 => => extracting sha256:71c2d6da5d28e6d960493c4e643061760eae300186bbdb6d6498670d08e5658d                                            0.0s
 => [ueransim stage-1  2/10] RUN apt-get update     && apt-get install libsctp-dev lksctp-tools iproute2 iputils-ping procps psmisc -y     && apt-get clean  60.5s
 => [free5gc-smf stage-1 3/7] WORKDIR /free5gc                                                                                       0.2s
 => [free5gc-smf stage-1 4/7] RUN mkdir -p config/ log/ cert/                                                                        5.4s
 => [free5gc-smf stage-1 5/7] COPY --from=builder /free5gc/smf ./                                                                    1.1s
 => [ueransim stage-1  3/10] RUN apt-get update && apt-get install -y vim strace net-tools iputils-ping curl netcat tcpdump         44.7s
 => [free5gc-smf stage-1 6/7] COPY --from=builder /free5gc/cert/smf.pem ./cert/                                                      0.3s
 => [free5gc-smf stage-1 7/7] COPY --from=builder /free5gc/cert/smf.key ./cert/                                                      0.5s
 => [free5gc-smf] exporting to image                                                                                                 3.9s
 => => exporting layers                                                                                                              3.6s
 => => writing image sha256:fbe8961f1052f838f602ad61cdc0a40e48efc149cba9f3ff9d559c36edd02b14                                         0.0s
 => => naming to docker.io/library/lab2-free5gc-smf                                                                                  0.1s
 => [free5gc-smf] resolving provenance for metadata file                                                                             0.1s
 => [ueransim stage-1  4/10] WORKDIR /ueransim                                                                                       0.1s
 => [ueransim stage-1  5/10] RUN mkdir -p config/ binder/                                                                            3.0s
 => [ueransim builder 2/2] RUN apt-get update     && apt-get install libsctp-dev lksctp-tools iproute2 -y     && wget https://github.com/Kitware/CMake/releases/download/v3  1660.2s
 => [ueransim stage-1  6/10] COPY --from=builder /UERANSIM/build/nr-gnb .                                                            0.3s
 => [ueransim stage-1  7/10] COPY --from=builder /UERANSIM/build/nr-ue .                                                             0.3s
 => [ueransim stage-1  8/10] COPY --from=builder /UERANSIM/build/nr-cli .                                                            0.1s
 => [ueransim stage-1  9/10] COPY --from=builder /UERANSIM/build/nr-binder binder/                                                   0.2s
 => [ueransim stage-1 10/10] COPY --from=builder /UERANSIM/build/libdevbnd.so binder/                                                0.1s
 => [ueransim] exporting to image                                                                                                   10.6s
 => => exporting layers                                                                                                             10.5s
 => => writing image sha256:ed64d92560748ba18a8ca326f58e9577849f1f87b12fd22dc6c633908b386c04                                         0.0s
 => => naming to docker.io/library/lab2-ueransim                                                                                     0.0s
 => [ueransim] resolving provenance for metadata file                                                                                0.1s
[+] Building 13/13
 ✔ free5gc-amf    Built                                                                                                              0.0s
 ✔ free5gc-ausf   Built                                                                                                              0.0s
 ✔ free5gc-chf    Built                                                                                                              0.0s
 ✔ free5gc-n6gw   Built                                                                                                              0.0s
 ✔ free5gc-nrf    Built                                                                                                              0.0s
 ✔ free5gc-nssf   Built                                                                                                              0.0s
 ✔ free5gc-pcf    Built                                                                                                              0.0s
 ✔ free5gc-smf    Built                                                                                                              0.0s
 ✔ free5gc-udm    Built                                                                                                              0.0s
 ✔ free5gc-udr    Built                                                                                                              0.0s
 ✔ free5gc-upf    Built                                                                                                              0.0s
 ✔ free5gc-webui  Built                                                                                                              0.0s
 ✔ ueransim       Built                                                                                                              0.0s
quy@quy:~/network_lab/Lab2$
```

3. Run container

Before I run up the container, I need to modify the config file in ./config directory. Here is the 5G core network architecture:



5G core network architecture

Because we need to configure Docker networks for 5G interfaces (N1/N2, N3, N4, N6), we should configure the such files: amfcfg.yaml, smfcfg.yaml, upfcfg.yaml, and gnbcfg.yaml:

- In amfcfg.yaml, ngapIpList is the list of IP addresses that AMF will listen on to establish NGAP connections (N2 interface) from gNB, In 313540040.yaml, amf container is using alias amf.n2.org, so I will set this value to FDQN of ngapIpList in amfcfg.yaml (line8).

```
configuration:
  amfName: AMF # the name of this AMF
  ngapIpList:  # the IP list of N2 interfaces on this AMF
    - amf.n2.org
  ngapPort: 38412 # the SCTP port listened by NGAP
  sbi: # Service-based interface information
```

- In smfcfg.yaml, we have to modify the SMF and UPF properties. UPF connect to SMF through N4 interface. I set SMF nodeID is smf.free5gc.org, listenAddr, externalAddr is amf.n4.org (line 52-54). The UPF nodeID and UPF N4 interface IP is upf.n4.org (line 66-67). The N3 interface endpoint (line 75) will be upf N3 static ip defined in 313540040.yaml: 10.100.3.100.

```
  pfcp: # the IP address of N4 interface on this SMF (PFCP)
    # addr config is deprecated in smf config v1.0.3, please use the following config
    nodeID: smf.free5gc.org # the Node ID of this SMF
    listenAddr: smf.n4.org # the IP/FQDN of N4 interface on this SMF (PFCP)
    externalAddr: smf.n4.org # the IP/FQDN of N4 interface on this SMF (PFCP)
  userplaneInformation: # list of userplane information
    upNodes: # information of userplane node (AN or UPF)
      gNB1: # the name of the node
        type: AN # the type of the node (AN or UPF)
      UPF: # the name of the node
        type: UPF # the type of the node (AN or UPF)
        nodeID: upf.n4.org # the Node ID of this UPF
        addr: upf.n4.org # the IP/FQDN of N4 interface on this UPF (PFCP)
        sNssaiUpfInfos: # S-NSSAI information list for this UPF
```

```
        interfaces: # Interface list for this UPF
          - interfaceType: N3 # the type of the interface (N3 or N9)
            endpoints: # the IP address of this N3/N9 interface on this UPF
              - 10.100.3.100
            networkInstances: # Data Network Name (DNN)
              - internet
    links: # the topology graph of userplane, A and B represent the two nodes of each link
```

- In the upfcfg.yaml, I need to set the N4 interface of UPF in line 6-7, so I change it to upf.n4.org. The N3 address will be upf N3 static ip defined in 313540040.yaml: 10.100.3.100 (line 14)

```
pfcp:
  addr: upf.n4.org # IP addr for listening
  nodeID: upf.n4.org # External IP or FQDN can be reached
  retransTimeout: 1s # retransmission timeout
  maxRetrans: 3 # the max number of retransmission

gtpu:
  forwarder: gtp5g
  # The IP list of the N3/N9 interfaces on this UPF
  # If there are multiple connection, set addr to 0.0.0.0 or list all the addresses
  ifList:
    - addr: 10.100.3.100
      type: N3
```

- In gnbcfg.yaml, I set the gNB's local IP address for N2 and N3 Interface is gnb.n2.org and gnb.n3.org (line 9-10) and amf address is amf.n2.org.

```
linkIp: 127.0.0.1 # gNB's local IP address for Radio Link Simulation (Usually same with local IP)
ngapIp: gnb.n2.org #local IP address for N2 Interface (Usually same with local IP)
gtpIp: gnb.n3.org # gNB's local IP address for N3 Interface (Usually same with local IP)

# List of AMF address information
amfConfigs:
  - address: amf.n2.org
    port: 38412
```

After modifying the config file, I run up the container by <docker compose up>:

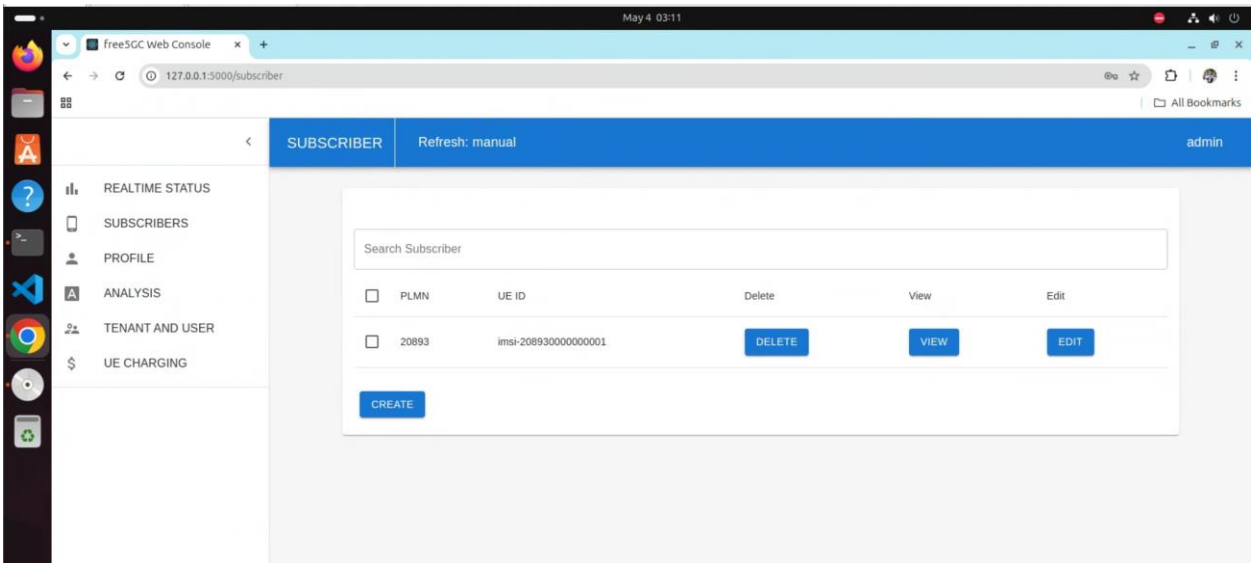
This command will run up all the containers defined in 313540040.yaml file, image is take from the previous step.

```
quydinh@quydin:~/computer_network/Lab2$ sudo docker compose -f 313540040.yaml up -d
[sudo] password for quydinh:
[+] Running 14/14
 ✔ Container n6gw      Started
 ✔ Container upf       Started
 ✔ Container mongodb   Started
 ✔ Container nrf       Started
 ✔ Container pcf       Started
 ✔ Container smf       Started
 ✔ Container webui     Started
 ✔ Container chf       Started
 ✔ Container nssf      Started
 ✔ Container udm       Started
 ✔ Container udr       Started
 ✔ Container ausf      Started
 ✔ Container amf       Started
 ✔ Container ueransim  Started
```

```
quydinh@quydin:~/computer_network/Lab2$ sudo docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED         STATUS         PORTS
                                            NAMES
380d000093fc   lab2-ueransim            "./nr-gnb -c ./confi…"   7 seconds ago   Up 6 seconds
                                            ueransim
e5f860acb3bd   lab2-free5gc-ausf        "./ausf -c ./config/…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            ausf
360860d66ae2   lab2-free5gc-webui       "./webui -c ./config…"   7 seconds ago   Up 6 seconds   0.0.0.0:2121-2122->2121-2122/tcp, [::]:2121-2122->2121-2122/tcp, 0.
0.0.0:5000->5000/tcp, [::]:5000->5000/tcp   webui
59f0ffbeeb66   lab2-free5gc-udr         "./udr -c ./config/u…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            udr
19bb014dd1eb   lab2-free5gc-pcf         "./pcf -c ./config/p…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            pcf
9b73bc2dc127   lab2-free5gc-smf         "./smf -c ./config/s…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            smf
f721f5a061d4   lab2-free5gc-nssf        "./nssf -c ./config/…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            nssf
a671bcb43872   lab2-free5gc-udm         "./udm -c ./config/u…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            udm
f3e918534134   lab2-free5gc-amf         "./amf -c ./config/a…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            amf
f5392169225c   lab2-free5gc-nrf         "./nrf -c ./config/n…"   7 seconds ago   Up 6 seconds   8000/tcp
                                            nrf
19b6b61c68bb   mongo                    "docker-entrypoint.s…"   7 seconds ago   Up 7 seconds   27017/tcp
                                            mongodb
84cb38f1567f   lab2-free5gc-upf         "bash -c './upf-ipta…"   7 seconds ago   Up 7 seconds
                                            upf
69b32b7e392d   lab2-free5gc-n6gw        "bash -c './n6gw-ipt…"   7 seconds ago   Up 7 seconds
                                            n6gw
11e48d3ad7f7   portainer/portainer-ce   "/portainer"             6 days ago      Up 22 hours    0.0.0.0:8000->8000/tcp, [::]:8000->8000/tcp, 9443/tcp, 0.0.0.0:80->
9000/tcp, [::]:80->9000/tcp                 portainer
quydinh@quydin:~/computer_network/Lab2$
```

4. Test by exec to ueramsim container

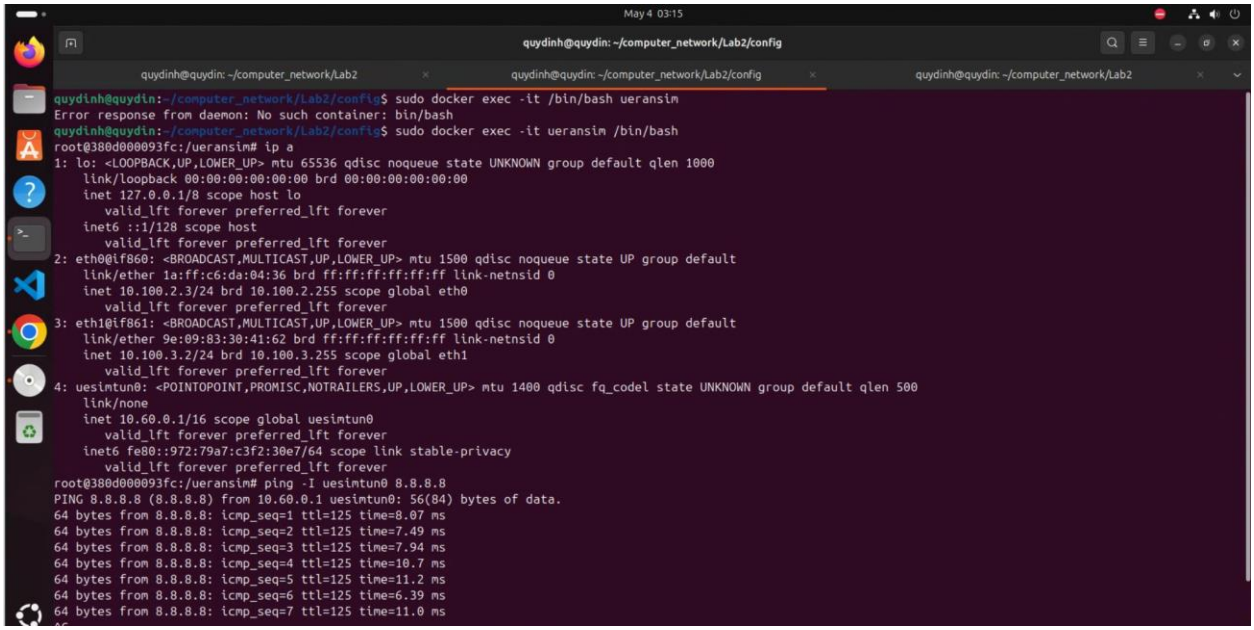
Go into WebUI and create Subscriber:



Execting to ueramsim container and run "**./nr-ue -c config/uecfg.yaml**" to register ueransim to the 5g core network.

```
^Croot@380d000093fc:/ueransim# quydinh@quydin:~/computer_network/Lab2$ sudo docker exec -it ueransim /bin/bash
root@380d000093fc:/ueransim# ./nr-ue -c config/uecfg.yaml
UERANSIM v3.2.7
[2025-05-03 19:10:56.839] [nas] [info] UE switches to state [MM-DEREGISTERED/PLMN-SEARCH]
[2025-05-03 19:10:56.839] [rrc] [debug] New signal detected for cell[1], total [1] cells in coverage
[2025-05-03 19:10:56.839] [nas] [info] Selected plmn[208/93]
[2025-05-03 19:10:56.839] [rrc] [info] Selected cell plmn[208/93] tac[1] category[SUITABLE]
[2025-05-03 19:10:56.839] [nas] [info] UE switches to state [MM-DEREGISTERED/PS]
[2025-05-03 19:10:56.839] [nas] [info] UE switches to state [MM-DEREGISTERED/NORMAL-SERVICE]
[2025-05-03 19:10:56.839] [nas] [debug] Initial registration required due to [MM-DEREG-NORMAL-SERVICE]
[2025-05-03 19:10:56.840] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2025-05-03 19:10:56.840] [nas] [debug] Sending Initial Registration
[2025-05-03 19:10:56.841] [nas] [info] UE switches to state [MM-REGISTER-INITIATED]
[2025-05-03 19:10:56.841] [rrc] [debug] Sending RRC Setup Request
[2025-05-03 19:10:56.842] [rrc] [info] RRC connection established
[2025-05-03 19:10:56.842] [rrc] [info] UE switches to state [RRC-CONNECTED]
[2025-05-03 19:10:56.842] [nas] [info] UE switches to state [CM-CONNECTED]
[2025-05-03 19:10:56.876] [nas] [debug] Authentication Request received
[2025-05-03 19:10:56.876] [nas] [debug] Received SQN [000000000023]
[2025-05-03 19:10:56.876] [nas] [debug] SQN-MS [000000000000]
[2025-05-03 19:10:56.895] [nas] [debug] Security Mode Command received
[2025-05-03 19:10:56.895] [nas] [debug] Selected integrity[2] ciphering[0]
[2025-05-03 19:10:56.964] [nas] [debug] Registration accept received
[2025-05-03 19:10:56.964] [nas] [info] UE switches to state [MM-REGISTERED/NORMAL-SERVICE]
[2025-05-03 19:10:56.964] [nas] [debug] Sending Registration Complete
[2025-05-03 19:10:56.964] [nas] [info] Initial Registration is successful
[2025-05-03 19:10:56.964] [nas] [debug] Sending PDU Session Establishment Request
[2025-05-03 19:10:56.969] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2025-05-03 19:10:57.170] [nas] [debug] Configuration Update Command received
[2025-05-03 19:10:57.272] [nas] [debug] PDU Session Establishment Accept received
[2025-05-03 19:10:57.272] [nas] [info] PDU Session establishment is successful PSI[1]
[2025-05-03 19:10:57.292] [app] [info] Connection setup for PDU session[1] is successful, TUN interface[uesimtun0, 10.60.0.1] is up.
```

After that I try to ping 8.8.8.8 through uesimtun0 interface. The successful ping mean that ue successfully register in to local core 5G network and can reach out to external networks.



B. Question and Answer

1. What are N1, N2, N3, N4, and N6 in the 5G core network?

The N1, N2, N3, N4, N6 interfaces define how different components communicate with each other:

- N1 is interface for signaling and registration (initial access, authentication, etc.) from UE (mobile phone, IoT device,…) to the AMF (Access and Mobility Management Function)
- N2 interface connect between gNB and AMF, using for UE registration, mobility management, handovers.
- N3 interface is connection between gNB and UPF, it handle the actual data traffic (internet, app data) from UE flows.
- N4 interface connect SMF (Session Management Function) to UPF, it tells the UPF how to route packets (e.g., rules, QoS) and controls user plane forwarding behavior
- N6 connect UPF to external network, it handle actual outgoing/incoming data to/from external networks like your UE's internet traffic after it passes through UPF

2. What is bridge network in docker?

A bridge network is the default network driver Docker uses. It acts as a virtual LAN where containers can talk to each other within the same host, it also provide isolation from containers that aren't connected to that bridge network .Docker assigns each container an IP within that network. Containers can also communicate via container name (DNS resolution).

In my lab, there are also bridge networks namely lab2_sbinet:

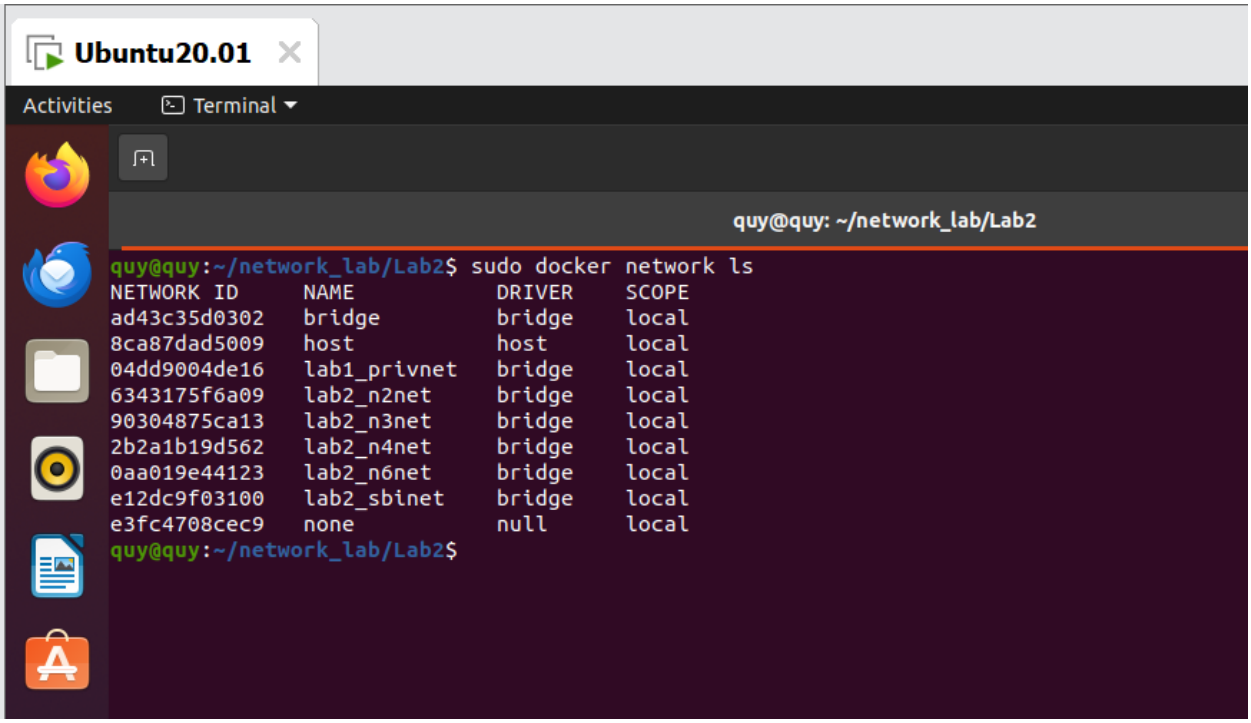
- SBI network (10.100.163.0/24) is defined by user (like me or any administrator) and connect 5G core components such as AMF, SMF, UPF, WebUI,... ensure containers can ping, communicate via TCP/UDP, and route according to the simulated packet flow in the 5G network

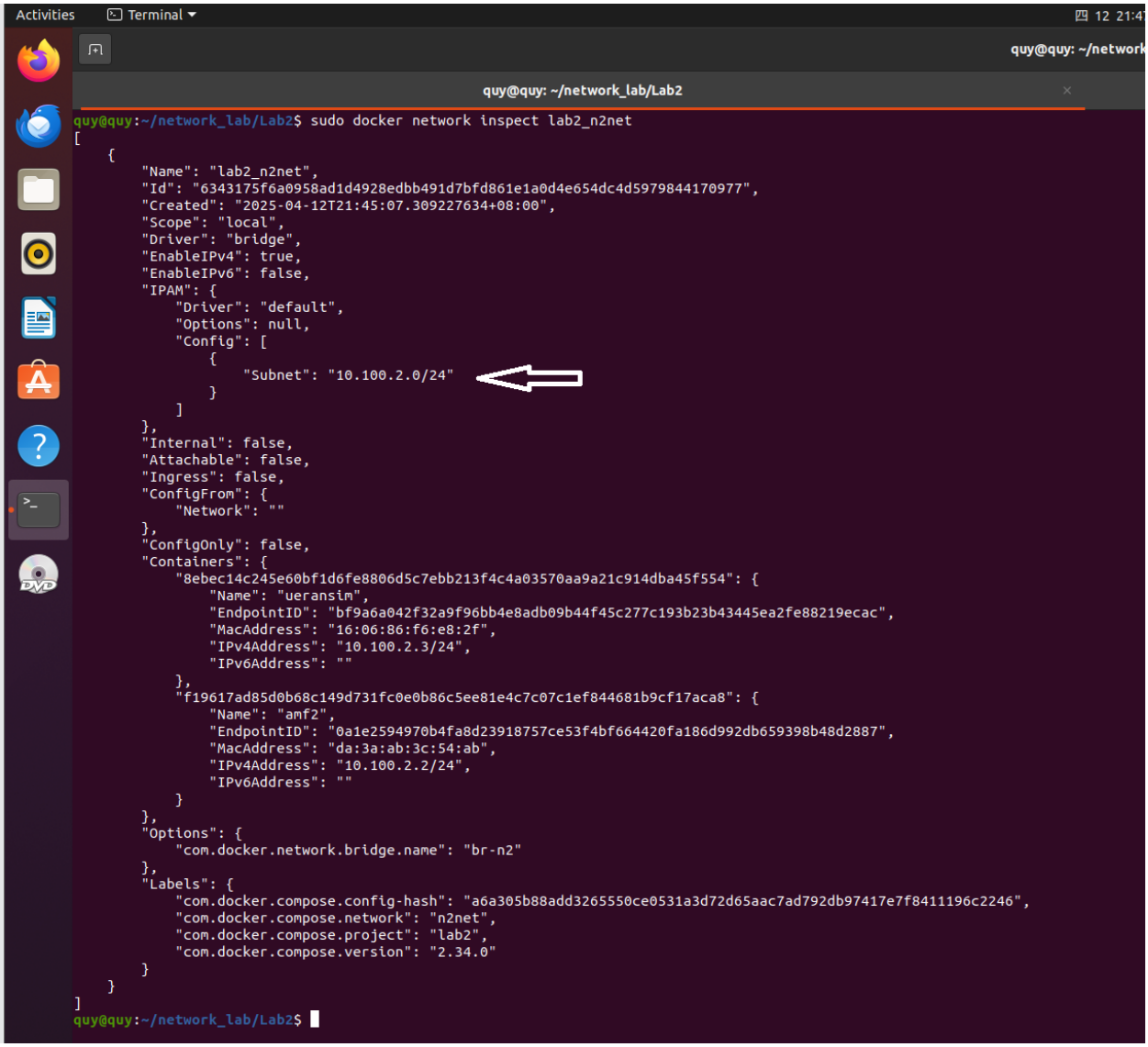
3. What is the name of each network you are using and the subnet for each?

To view all the networks, I run: sudo docker network ls

To view the network detail, I run: sudo docker network inspect <network name>

I use 5 network in this lab: lab2_n2net, lab2_n3net, lab2_n4net, lab2_n6net, lab2_sbinet, the rest are networks created from previous labs.





N2 network

- Name: lab2_n2net
- Subnet: "Subnet": "10.100.2.0/24"

N3 network

- Name: lab2_n3net
- Subnet: 10.100.3.0/24

N4 network

- Name: lab2_n4net
- Subnet: 10.100.4.0/24

N6 network

- Name: lab2_n6net
- Subnet: 10.100.6.0/24

SBI network

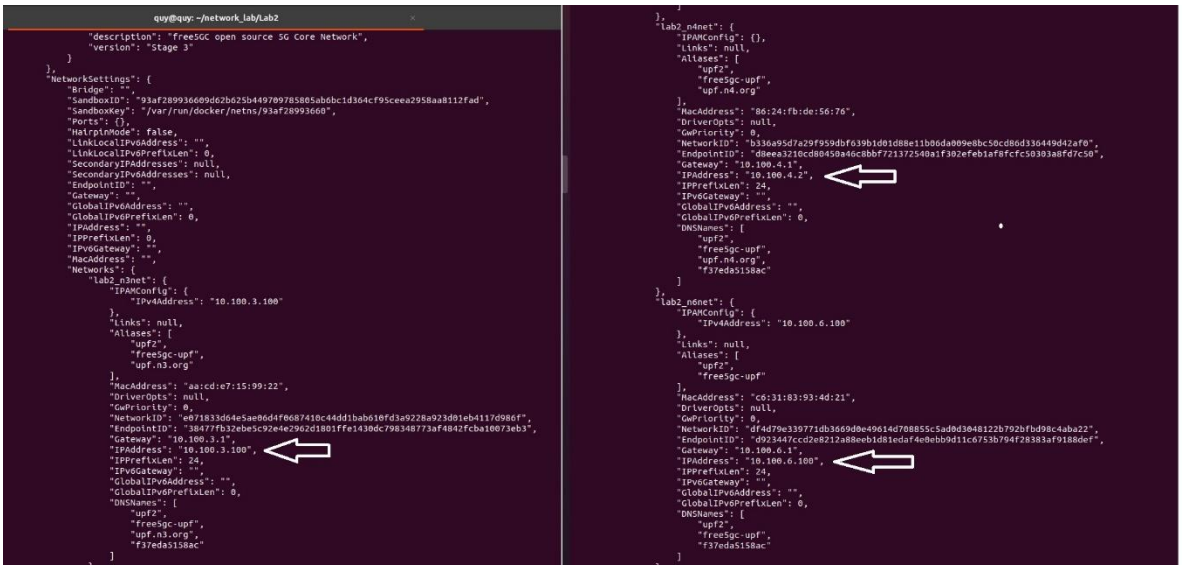
- Name: lab2_sbinetwork
- Subnet: 10.100.163.0/24

All the subnet of networks is working properly like I defined in the 313540040.yaml file, from line 305 to 345

4. What the IP Address for your UPF? (Also provides screenshot and steps for how you get your answers.)

To see the UPF IP_address, I run command:

docker inspect <upf_container_name>



N3 interface : 10.100.3.100

N4 interface: 10.100.4.2

N6 interface: 10.100.6.100

The ipv4 addresses are like we defined in the 313540040.yaml file, from line 14 to 23.

5. What protocol are used in N2, N3, N4 interfaces? (Describe each)

N2 Interface

Connects: gNB ↔ AMF (Access and Mobility Management Function)

Protocols:

1. NGAP (NG Application Protocol):
    - Runs over SCTP/IP.
    - Responsible for signaling between gNB and AMF.
    - Supports UE context management, NAS transport, paging, PDU session setup, etc.
2. SCTP (Stream Control Transmission Protocol):
    - Reliable transport layer protocol.
    - Allows multi-streaming and multi-homing between gNB and AMF.

N3 Interface

Connects: gNB ↔ UPF (User Plane Function)

Protocols:

1. GTP-U (GPRS Tunneling Protocol – User Plane):
    - Encapsulates user IP packets inside a tunnel.
    - Enables tunneling over IP for data traffic.
    - Runs over UDP/IP (typically port 2152).
2. UDP/IP: Transport protocol for GTP-U.

N4 Interface

Connects: SMF (Session Management Function) ↔ UPF

Protocols:

1. PFCP (Packet Forwarding Control Protocol):
    - Controls and configures rules on UPF (e.g., forwarding rules, buffer settings, usage reporting).
    - Defines session establishment, modification, deletion.
    - Runs over UDP/IP (typically port 8805).
2. UDP/IP: Transport protocol for PFCP messages.