

Mid-term report.
5G Framed Routing
Adding management of sub-networks on WebGUI

Leonardo Cordeiro Gonçalves - 2020228071
Gonçalo Tavares Bastos - 2020228071
Supervisores Altice: Engs. Francisco Fontes e Miguel Freitas
Supervisor UC: Prof. Maria Medeiros
Sponsor: Altice Labs

October 2022 - April 2023

1 Summary

Where we provide an overview of the progress made so far. Our project aims to validate the 5G framed route feature and manage sub-networks on WebGUI. Since the inception of the project, our team has been working diligently towards achieving the set goals and objectives.

In this mid-term report, we will provide a summary of the work done so far, including the challenges we have encountered, the milestones achieved, and the next steps we plan to take. We will also discuss the strategies employed to ensure that the project stays on track.

To date, most of the worktime we have was delivered for validating the 5G Framed Route using open5GS and UERANSIM, task that is already done. We also have been studying the API's northbound (ex: CAPIF) and we confirm that the Open5Gs UE provision API is aligned with current standards. Overall

In the upcoming weeks, we will try to improve the management web-interface of the 5G core in order to add support for static framed route provisioning, and also figure out ways to improve possible validations for overlapping routes.

We hope that this mid-term report provides valuable insight into the progress made so far, and we welcome any feedback or suggestions that will help us improve our project.

2 Project Status

The project has made significant progress since its inception. To date we achieved two main objectives:

- Validate framed routing functionality on open5GS, that is a open source 5G core implementation.
- Investigate 3GPP standardized northbounds API's in order to validate wether Open5GS UE provision API is aligned with current standards.

We are currently in the process of figure out a way to improve the management web-interface of the 5G core by adding support for static framed routes, and also to improve possible validations of overlapping routes.

2.1 Framed Route Validation

In order to test Framed Routes, we built a mobile 5GC simulated network, WITH Open5GS and UERANSIM.

The following minimum configuration was set as a condition:

- Two UEs have the same DNN and connect to the same DN.
- Two UEs have different Framed Routes. On the UPF VM, make sure to be able to ping the Framed Routes via the IP address (Tunnel GW/uesimtun0) assigned to each UE.

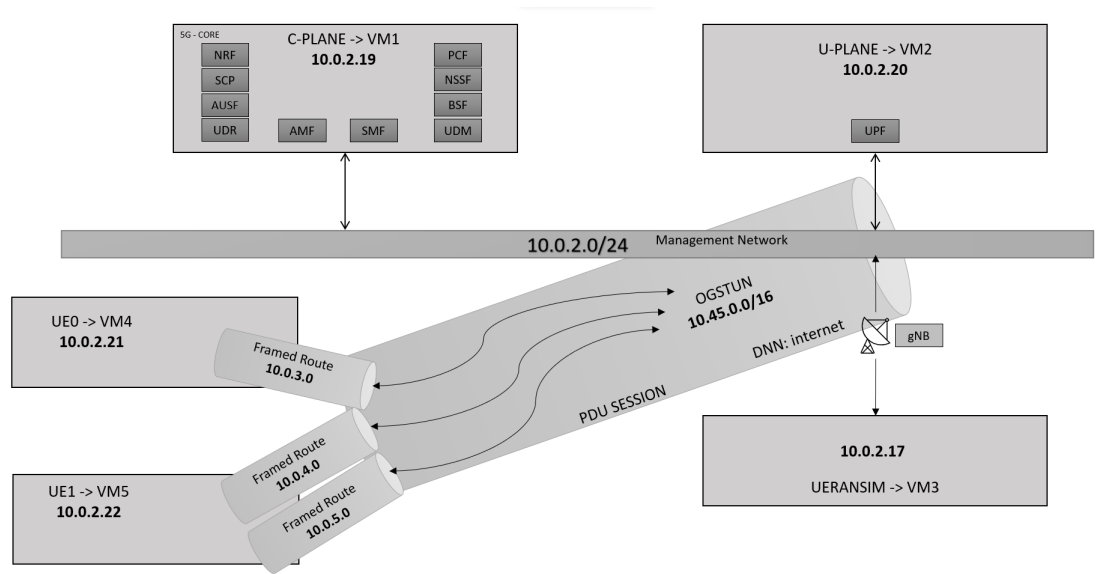


Figure 1: Our Network Illustration

The VM's are as follow:

VM #	SW & Role	IP address	OS	Memory (Min)	HDD (Min)
VM1	Open5GS 5GC C-Plane	10.0.2.19	Ubuntu 20.04	2GB	50GB
VM2	Open5GS 5GC U-Plane	10.0.2.20	Ubuntu 20.04	2GB	50GB
VM3	UERANSIM RAN (gNodeB)	10.0.2.17	Ubuntu 20.04	2GB	50GB
VM4	UERANSIM UE0	10.0.2.21	Ubuntu 20.04	2GB	50GB
VM5	UERANSIM UE1	10.0.2.22	Ubuntu 20.04	2GB	50GB

Figure 2: VM configurations

2.1.1 Core Network Configuration

To perform the validation of Framed Routing we need to change some core configuration files of Open5GS, in order to connect different components of 5G Network.

First of all, we set up C-Plane virtual machine (VM1), which represents the core of 5G Network with the following functions:

- NRF
- SCP
- AMF
- SMF
- AUSF
- UDM
- UDR
- PCF
- NSSF
- BSF

Changes in AMF (Access and Mobility Management Function): the AMF works in conjunction with other core network functions, such as the User Plane Function (UPF) and the Session Management Function (SMF), to ensure efficient communication between user devices and the network. It performs tasks such as authentication, authorisation, and accounting, as well as routing.

```

471 #
472 amf:
473   sbt:
474     - addr: 127.0.0.5
475       port: 7777
476   ngap:
477     - addr: 10.0.2.19
478   metrics:
479     - addr: 127.0.0.5
480       port: 9090
481   guami:
482     - plmn_id:
483         mcc: 001
484         mnc: 01
485         amf_id:
486           region: 2
487           set: 1
488   tai:
489     - plmn_id:
490         mcc: 001
491         mnc: 01
492       tac: 1
493   plmn_support:
494     - plmn_id:
495         mcc: 001
496         mnc: 01
497       s_nssai:
498         - sst: 1
499   security:
500     integrity_order : [ NIA2, NIA1, NIA0 ]
501     ciphering_order : [ NEA0, NEA1, NEA2 ]
502   network_name:
503     full: Open5GS
504   amf_name: open5gs-amf0
505
506 #

```

Figure 3: amf.yaml file configuration

NGAP is found on the N2 reference point between gNB and the AMF in order to support both UE and non UE associated services. The NGAP protocol handles the establishment, modification, and termination of PDU sessions by exchanging messages with the network functions involved in the session in order to transfer packets data between the user equipment and the 5G core network.

Changes in SMF (Session Management Function): where we change GTPU (GPRS Tunneling Protocol User Plane) and PFCP (Packet Forwarding Control Protocol) IP's. The SMF uses the GTP-U protocol to manage the data sessions, GTP-U tunnels are used to transport the data between the user equipment and the UPF, the SMF sets up these tunnels and ensures that the necessary resources are allocated to support the data transfer. PFCP is used by the SMF to control and manage the user plane functions, the protocol enables the SMF to communicate with the UPF for session establishment, modification, and termination.

To conclude the core network configuration, we need to set up UPF, that runs on U-Plane (VM2), in order to establish connections with C-Plane functions and also with gNodeB on RAN machine (VM3).

```

601 smf:
602
603   sbi:
604     - addr: 127.0.0.4
605       port: 7777
606   pfcf:
607     - addr: 10.0.2.19
608   gtpc:
609     - addr: 127.0.0.4
610   gtpu:
611     - addr: 10.0.2.19
612   metrics:
613     - addr: 127.0.0.4
614       port: 9090
615   subnet:
616     - addr: 10.45.0.1/16
617       dnn: internet
618   dns:
619     - 8.8.8.8
620     - 8.8.4.4
621   mtu: 1400
622   ctf:
623     enabled: auto
624   freeDiameter: /etc/freeDiameter/smf.conf
625 #

```

Figure 4: smf.yaml file configuration

```

197 upf:
198   pfcf:
199     - addr: 10.0.2.20
200   gtpu:
201     - addr: 10.0.2.20
202   subnet:
203     - addr: internet
204     - dev: ogstun
205   metrics:
206     - addr: 127.0.0.7
207       port: 9090
208

```

Figure 5: upf.yaml file configuration

```

805 upf:
806   pfcf:
807     - addr: 10.0.2.20
808     dnn: internet
809

```

Figure 6: upf.yaml file configuration

2.1.2 UERANSIM Configuration

- RAN (gNodeB) - Configuration (VM3): The gNB connects to the 5G core network through the NG-RAN (Next Generation Radio Access Network)

interface, which enables communication between the gNB and the core network functions.

```

1 mcc: '001' # Mobile Country Code value
2 mnc: '01' # Mobile Network Code value (2 or 3 digits)
3
4 nci: '0x000000010' # NR Cell Identity (36-bit)
5 idLength: 32 # NR gNB ID length in bits [22..32]
6 tac: 1 # Tracking Area Code
7
8 linkIp: 10.0.2.17 # gNB's local IP address for Radio Link Simulation (Usually same with
local IP)
9 ngapIp: 10.0.2.17 # gNB's local IP address for N2 Interface (Usually same with local IP)
10 helpIp: 10.0.2.17 # gNB's local IP address for N3 Interface (Usually same with local IP)
11
12 # List of AMF address information
13 amfConfigs:
14 - address: 10.0.2.19
15 port: 38412
16
17 # List of supported S-NSSAIs by this gNB
18 slices:
19 - sst: 1
20
21 # Indicates whether or not SCTP stream number errors should be ignored.
22 ignoreStreamIds: true

```

Figure 7: UERANSIM - open5gs-gnb.yaml file configuration

The gNB sends signaling messages to the AMF over the NG-C interface to initiate and manage the connection between the UE and the core network. These messages include information such as UE identification, location, and requested services. The AMF uses this information to authenticate the UE, allocate network resources, and establish the necessary network functions, such as the SMF and UPF, to support the requested services.

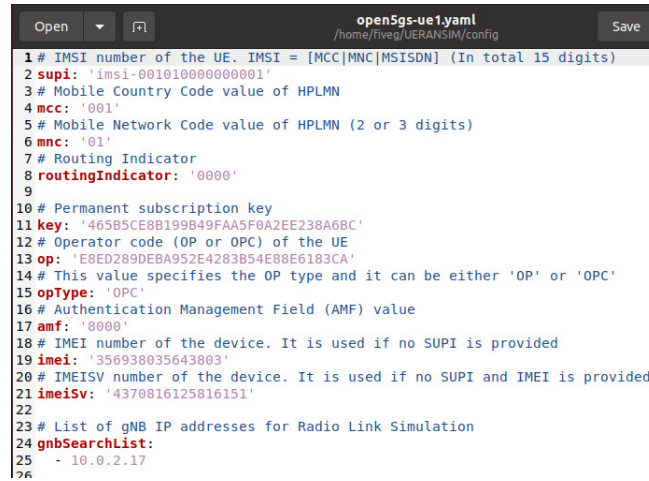
- UE0 and UE1 Configuration(VM4 and VM5): where we need to set up gNBSearchList to connect the user equipment with gNB(VM3).

```

4 mcc: '001'
5 # Mobile Network Code value of HPLMN (2 or 3 digits)
6 mnc: '01'
7 # Routing Indicator
8 routingIndicator: '0000'
9
10 # Permanent subscription key
11 key: '465B5CE8B199B49FAA5F0A2EE238A6BC'
12 # Operator code (OP or OPC) of the UE
13 op: 'E8ED289DEBA952E4283B54E88E6183CA'
14 # This value specifies the OP type and it can be either 'OP' or 'OPC'
15 opType: 'OPC'
16 # Authentication Management Field (AMF) value
17 amf: '8000'
18 # IMEI number of the device. It is used if no SUPI is provided
19 imei: '356938035643803'
20 # IMEISV number of the device. It is used if no SUPI and IMEI is provided
21 imeiSv: '4370816125816151'
22
23 # List of gNB IP addresses for Radio Link Simulation
24 gnbSearchList:
25 - 10.0.2.17
26

```

Figure 8: UERANSIM - open5gs-ue0.yaml file configuration



```
1 # IMSI number of the UE. IMSI = [MCC|MNC|MSISDN] (In total 15 digits)
2 supi: 'imsi-001010000000001'
3 # Mobile Country Code value of HPLMN
4 mcc: '001'
5 # Mobile Network Code value of HPLMN (2 or 3 digits)
6 mnc: '01'
7 # Routing Indicator
8 routingIndicator: '0000'
9
10 # Permanent subscription key
11 key: '465B5CE8B199B49FAA5F0A2EE238A6BC'
12 # Operator code (OP or OPC) of the UE
13 op: 'E8ED289DEBA952E4283B54E88E6183CA'
14 # This value specifies the OP type and it can be either 'OP' or 'OPC'
15 opType: 'OPC'
16 # Authentication Management Field (AMF) value
17 amf: '8000'
18 # IMEI number of the device. It is used if no SUPI is provided
19 imei: '356938035643803'
20 # IMEISV number of the device. It is used if no SUPI and IMEI is provided
21 imeiSV: '4370816125816151'
22
23 # List of gNB IP addresses for Radio Link Simulation
24 gnbSearchList:
25   - 10.0.2.17
26
```

Figure 9: UERANSIM - open5gs-ue1.yaml file configuration

2.1.3 WebGUI Subscriptions

One of the main points of our project is to add and manage users on open5GS WebGUI, WenGUI is a web-based user interface for the administration and configuration of the Open5GS core network components. By using WebGUI network administrators can manage subscribers via IMSI, KEY and OPC. With these parameters, we can set up the appropriate PDU sessions, ensuring access to DN for each subscriber via the appropriate interfaces. So we need to set the IMSI correspondent to our users. We have two ways to do this: the first one is directly on WebGUI interface, and the other one is by mongoDB Compass application that allows to change directly the subscribers information on database. We prefer to add the subscribers information directly on mongoDB Compass.

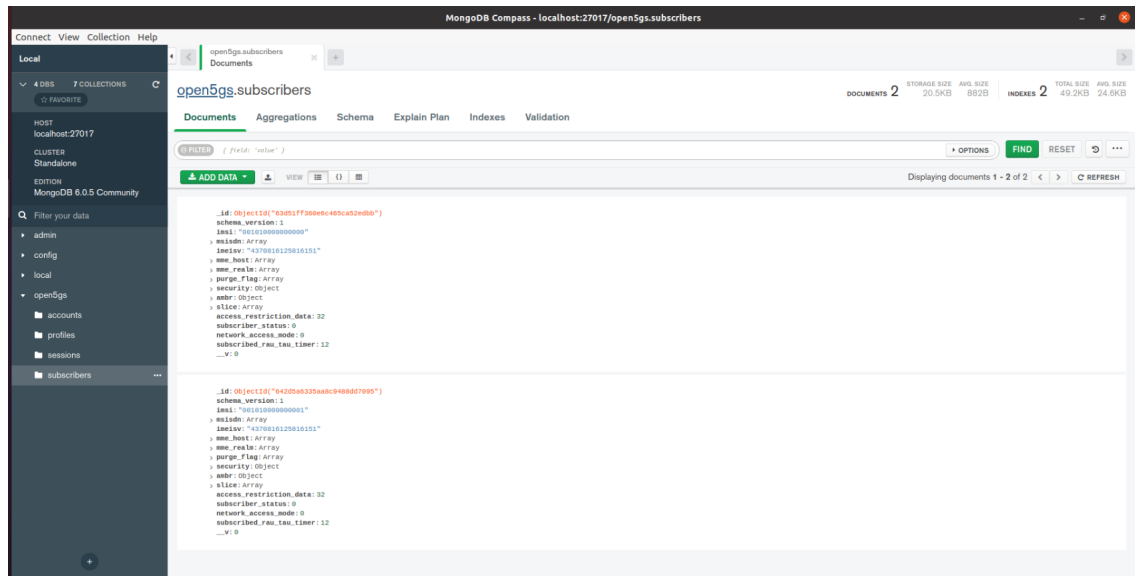


Figure 10: MongoDB Compass Interface

We add the following on subscribers JSON file with the essential parameters to be added the Framed Routes to users subscribers.

Once we add the subscribers file on mongoDB Compass, the users will automatically appears on WebGUI interface as we can see on image below:


```

103 "slice": {
104   {
105     "sst": 1,
106     "default_indicator": true,
107     "session": [
108       {
109         "name": "internet",
110         "type": 3,
111         "qos": {
112           "index": 9,
113           "arp": {
114             "priority_level": 8,
115             "pre_emption_capability": 1,
116             "pre_emption_vulnerability": 1
117           }
118         },
119         "ambr": {
120           "downlink": {
121             "value": 1,
122             "unit": 3
123           },
124           "uplink": {
125             "value": 1,
126             "unit": 3
127           }
128         },
129         "_id": {
130           "$oid": "642d5a6335aa8c9488dd7097"
131         },
132         "ipv4_framed_routes": ["10.0.4.0/24", "10.0.5.0/24"],
133         "pcc_rule": []
134       }
135     ],
136     "_id": {
137       "$oid": "642d5a6335aa8c9488dd7096"
138     }
139   }
140 }
141 },
142

```

Figure 11: Subscribers file, UE1 example

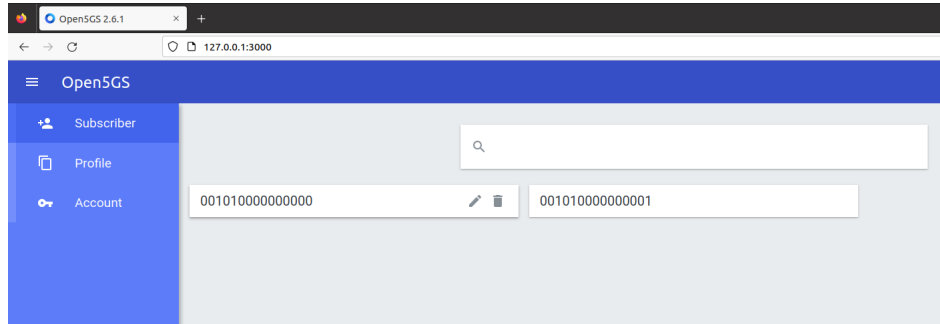


Figure 12: WebGUI Interface

2.1.4 Validation

After all this changes have been made, we are able to proceed to the validation of the Frame Routing.

We need to start and initialise the virtual machines for the following order:

1. Start C-Plane (VM1)
2. Start U-Plane (VM2)

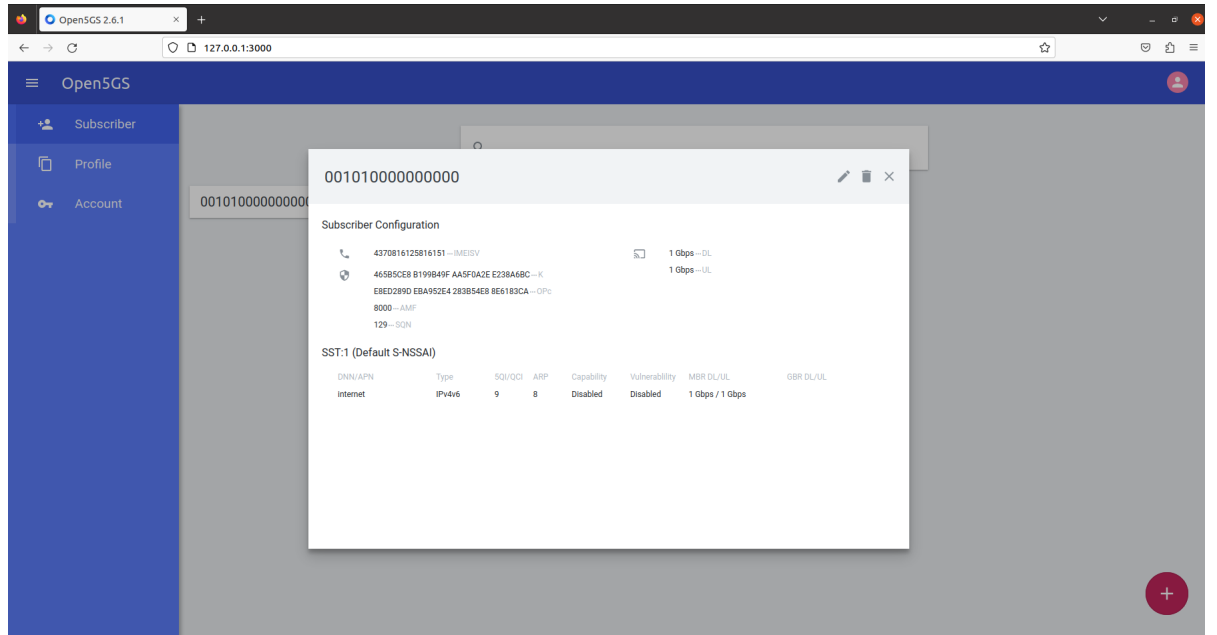


Figure 13: WebGUI Interface - UE subscriber information

3. Start gNB (VM3)
4. Start UE0 (VM4)
5. Start UE1 (VM5)

First of all, we need to start open5gs daemons/services, for that we could use the following procedure:

- `systemctl restart open5gs-mmed;`
- `systemctl restart open5gs-sgwcd;`
- `systemctl restart open5gs-smfd;`
- `systemctl restart open5gs-amfd`
- `systemctl restart open5gs-sgwud`
- `systemctl restart open5gs-hssd`
- `systemctl restart open5gs-pcrfd`
- `systemctl restart open5gs-nrfd`
- `systemctl restart open5gs-scpd`
- `systemctl restart open5gs-ausfd`
- `systemctl restart open5gs-udmd`

- systemctl restart open5gs-pcfd
- systemctl restart open5gs-nssfd
- systemctl restart open5gs-bsfd
- systemctl restart open5gs-udrd
- systemctl restart open5gs-webui

We should check if all daemons are running using *systemctl status open5gs-**.

Next we need to start UPF daemon on U-Plane machine with:

- systemctl restart open5gs-mmed;

Once the services of C-Plane and U-Plane are running without any error or warning, we are able to start gNB on UERANSIM-VM3, that can be started by the following command: *./nr-gnb -c ../config/open5gs-gnb.yaml*.
Case it started correctly, we are supposed to see the following logs:

```
fiveg@fiveg-VirtualBox:~/UERANSIM$ cd build
fiveg@fiveg-VirtualBox:~/UERANSIM/build$ ls -l
total 10848
-rwxrwxr-x 1 fiveg fiveg 15560 apr 15 16:57 libdevbnd.so
-rw-rw-r-- 1 fiveg fiveg 377 apr 15 16:57 nr-binder
-rwxrwxr-x 1 fiveg fiveg 350568 apr 15 16:57 nr-cli
-rwxrwxr-x 1 fiveg fiveg 6185128 apr 15 16:57 nr-gnb
-rwxrwxr-x 1 fiveg fiveg 4545768 apr 15 16:57 nr-ue
fiveg@fiveg-VirtualBox:~/UERANSIM/build$ sudo -s
[sudo] password for fiveg:
root@fiveg-VirtualBox:/home/fiveg/UERANSIM/build# ./nr-gnb -c ../config/open5gs-gnb.yaml
UERANSIM v3.2.6
[2023-04-20 00:23:31.083] [sctp] [info] Trying to establish SCTP connection... (10.0.2.19:38412)
[2023-04-20 00:23:31.156] [sctp] [info] SCTP connection established (10.0.2.19:38412)
[2023-04-20 00:23:31.156] [sctp] [debug] SCTP association setup ascId[3]
[2023-04-20 00:23:31.156] [ngap] [debug] Sending NG Setup Request
[2023-04-20 00:23:31.486] [ngap] [debug] NG Setup Response received
[2023-04-20 00:23:31.486] [ngap] [info] NG Setup procedure is successful
```

Figure 14: Start logs of gNB

We also can confirm the gNB initialisation on AMF logs file, on C-Plane(VM1) side.

```
45 04/20 00:23:31.210: [amf] INFO: gNB-N2 accepted[10.0.2.17]:57633 in ng-path module (./src/amf/ngap-sctp.c:113)
46 04/20 00:23:31.223: [amf] INFO: gNB-N2 accepted[10.0.2.17] in master_sm module (./src/amf/amf-sm.c:733)
47 04/20 00:23:31.272: [amf] INFO: [Added] Number of gNBs is now 1 (./src/amf/context.c:1175)
48 04/20 00:23:31.406: [amf] INFO: gNB-N2[10.0.2.17] max_num_of_ostreams : 10 (./src/amf/amf-sm.c:772)
```

Figure 15: Start logs of gNB

With the image above we can see that the connection established between gNodeB and AMF was done by N2 interface, this is essential because before a service can be accessed, the UE must be connected to the network. That's why N2 is so important, because it handles control-plane signalling.

So now, we are able to connect the first user UE0(VM4). It can be done by the following comand `./nr-ue -c ../config/open5gs-ue0.yaml`. This will register the UE with 5GC and establish a PDU session.

```

root@fiveg-VirtualBox:/home/fiveg/UERANSIM/build# ./nr-ue -c ../config/open5gs-ue0.yaml
UERANSIM v3.2.6
[2023-04-20 00:27:41.044] [nas] [Info] UE switches to state [MM-DEREGISTERED/PLMN-SEARCH]
[2023-04-20 00:27:41.044] [rrc] [debug] New signal detected for cell[1], total [1] cells in coverage
[2023-04-20 00:27:41.045] [nas] [Info] Selected plmn[001/01]
[2023-04-20 00:27:41.045] [rrc] [Info] Selected cell plmn[001/01] tac[1] category[SUITABLE]
[2023-04-20 00:27:41.045] [nas] [Info] UE switches to state [MM-DEREGISTERED/PS]
[2023-04-20 00:27:41.045] [nas] [Info] UE switches to state [MM-DEREGISTERED/NORMAL-SERVICE]
[2023-04-20 00:27:41.045] [nas] [debug] Initial registration required due to [MM-DEREG-NORMAL-SERVICE]
[2023-04-20 00:27:41.046] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-04-20 00:27:41.046] [nas] [debug] Sending Initial Registration
[2023-04-20 00:27:41.047] [rrc] [debug] Sending RRC Setup Request
[2023-04-20 00:27:41.047] [nas] [Info] UE switches to state [MM-REGISTER-INITIATED]
[2023-04-20 00:27:41.047] [rrc] [Info] RRC connection established
[2023-04-20 00:27:41.047] [rrc] [Info] UE switches to state [RRC-CONNECTED]
[2023-04-20 00:27:41.047] [nas] [Info] UE switches to state [CM-CONNECTED]
[2023-04-20 00:27:41.366] [nas] [debug] Authentication Request received
[2023-04-20 00:27:41.366] [nas] [debug] Sending Authentication Failure due to SQN out of range
[2023-04-20 00:27:41.396] [nas] [debug] Authentication Request received
[2023-04-20 00:27:41.401] [nas] [debug] Security Mode Command received
[2023-04-20 00:27:41.401] [nas] [debug] Selected integrity[2] ciphering[0]
[2023-04-20 00:27:41.549] [nas] [debug] Registration accept received
[2023-04-20 00:27:41.549] [nas] [Info] UE switches to state [MM-REGISTERED/NORMAL-SERVICE]
[2023-04-20 00:27:41.549] [nas] [debug] Sending Registration Complete
[2023-04-20 00:27:41.549] [nas] [Info] Initial Registration is successful
[2023-04-20 00:27:41.549] [nas] [debug] Sending PDU Session Establishment Request
[2023-04-20 00:27:41.549] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-04-20 00:27:41.760] [nas] [debug] Configuration Update Command received
[2023-04-20 00:27:42.300] [nas] [debug] PDU Session Establishment Accept received
[2023-04-20 00:27:42.313] [nas] [Info] PDU Session establishment is successful PSI[1]
[2023-04-20 00:27:42.588] [app] [Info] Connection setup for PDU session[1] is successful, TUN interface[uesimtun0, 10.45.0.2] is up.

```

Figure 16: Start logs of UE0

At last, we start the UE1 with the following command `./nr-ue -c ../config/open5gs-ue1.yaml`, and if all going well we have completed the starting procedure.

Now we need to do some Network configurations which consists to configure TUNnel interface (ogstun - 10.45.0.0) and NAPT on U-Plane(VM2), and set the respectively Framed Route IP adress (ex:10.0.3.0//10.0.4.0//10.0.5.0) and the routing DN(10.45.0.0) to the uesimtun0 interface on UE0(VM4) and UE1(VM5).

So now, we have all configurations done. To confirm the framed route we have done some PING tests.

```

root@fiveg-VirtualBox:/home/fiveg/UEANSIM/build# ./nr-ue -c ../config/opensgs-ue0.yaml
UEANSIM v3.2.6
[2023-04-20 00:27:41.044] [nas] [Info] UE switches to state [MM-DEREGISTERED/PLMN-SEARCH]
[2023-04-20 00:27:41.044] [rrc] [debug] New signal detected for cell[1], total [1] cells in coverage
[2023-04-20 00:27:41.045] [nas] [Info] Selected plmn[001/01]
[2023-04-20 00:27:41.045] [rrc] [Info] Selected cell plmn[001/01] tac[1] category[SUITABLE]
[2023-04-20 00:27:41.045] [nas] [Info] UE switches to state [MM-DEREGISTERED/PS]
[2023-04-20 00:27:41.045] [nas] [Info] UE switches to state [MM-DEREGISTERED/NORMAL-SERVICE]
[2023-04-20 00:27:41.045] [nas] [debug] Initial registration required due to [MM-DEREG-NORMAL-SERVICE]
[2023-04-20 00:27:41.046] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-04-20 00:27:41.046] [nas] [debug] Sending Initial Registration
[2023-04-20 00:27:41.047] [rrc] [debug] Sending RRC Setup Request
[2023-04-20 00:27:41.047] [nas] [Info] UE switches to state [MM-REGISTER-INITIATED]
[2023-04-20 00:27:41.047] [rrc] [Info] RRC connection established
[2023-04-20 00:27:41.047] [rrc] [Info] UE switches to state [RRC-CONNECTED]
[2023-04-20 00:27:41.047] [nas] [Info] UE switches to state [CM-CONNECTED]
[2023-04-20 00:27:41.366] [nas] [debug] Authentication Request received
[2023-04-20 00:27:41.366] [nas] [debug] Sending Authentication Failure due to SQN out of range
[2023-04-20 00:27:41.396] [nas] [debug] Authentication Request received
[2023-04-20 00:27:41.401] [nas] [debug] Security Mode Command received
[2023-04-20 00:27:41.401] [nas] [debug] Selected integrity[2] ciphering[0]
[2023-04-20 00:27:41.549] [nas] [debug] Registration accept received
[2023-04-20 00:27:41.549] [nas] [Info] UE switches to state [MM-REGISTERED/NORMAL-SERVICE]
[2023-04-20 00:27:41.549] [nas] [debug] Sending Registration Complete
[2023-04-20 00:27:41.549] [nas] [Info] Initial Registration is successful
[2023-04-20 00:27:41.549] [nas] [debug] Sending PDU Session Establishment Request
[2023-04-20 00:27:41.549] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-04-20 00:27:41.760] [nas] [debug] Configuration Update Command received
[2023-04-20 00:27:42.300] [nas] [debug] PDU Session Establishment Accept received
[2023-04-20 00:27:42.313] [nas] [Info] PDU Session establishment is successful PSI[1]
[2023-04-20 00:27:42.588] [app] [Info] Connection setup for PDU session[1] is successful, TUN interface[uesimtun0, 10.45.0.2] is up.

```

Figure 17: Start logs of UE0 on amf log file

```

root@fiveg-VirtualBox:/home/fiveg/UEANSIM/build# ./nr-ue -c ../config/opensgs-ue0.yaml
UEANSIM v3.2.6
[2023-04-20 00:27:41.044] [nas] [Info] UE switches to state [MM-DEREGISTERED/PLMN-SEARCH]
[2023-04-20 00:27:41.044] [rrc] [debug] New signal detected for cell[1], total [1] cells in coverage
[2023-04-20 00:27:41.045] [nas] [Info] Selected plmn[001/01]
[2023-04-20 00:27:41.045] [rrc] [Info] Selected cell plmn[001/01] tac[1] category[SUITABLE]
[2023-04-20 00:27:41.045] [nas] [Info] UE switches to state [MM-DEREGISTERED/PS]
[2023-04-20 00:27:41.045] [nas] [Info] UE switches to state [MM-DEREGISTERED/NORMAL-SERVICE]
[2023-04-20 00:27:41.045] [nas] [debug] Initial registration required due to [MM-DEREG-NORMAL-SERVICE]
[2023-04-20 00:27:41.046] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-04-20 00:27:41.046] [nas] [debug] Sending Initial Registration
[2023-04-20 00:27:41.047] [rrc] [debug] Sending RRC Setup Request
[2023-04-20 00:27:41.047] [nas] [Info] UE switches to state [MM-REGISTER-INITIATED]
[2023-04-20 00:27:41.047] [rrc] [Info] RRC connection established
[2023-04-20 00:27:41.047] [rrc] [Info] UE switches to state [RRC-CONNECTED]
[2023-04-20 00:27:41.047] [nas] [Info] UE switches to state [CM-CONNECTED]
[2023-04-20 00:27:41.366] [nas] [debug] Authentication Request received
[2023-04-20 00:27:41.366] [nas] [debug] Sending Authentication Failure due to SQN out of range
[2023-04-20 00:27:41.396] [nas] [debug] Authentication Request received
[2023-04-20 00:27:41.401] [nas] [debug] Security Mode Command received
[2023-04-20 00:27:41.401] [nas] [debug] Selected integrity[2] ciphering[0]
[2023-04-20 00:27:41.549] [nas] [debug] Registration accept received
[2023-04-20 00:27:41.549] [nas] [Info] UE switches to state [MM-REGISTERED/NORMAL-SERVICE]
[2023-04-20 00:27:41.549] [nas] [debug] Sending Registration Complete
[2023-04-20 00:27:41.549] [nas] [Info] Initial Registration is successful
[2023-04-20 00:27:41.549] [nas] [debug] Sending PDU Session Establishment Request
[2023-04-20 00:27:41.549] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-04-20 00:27:41.760] [nas] [debug] Configuration Update Command received
[2023-04-20 00:27:42.300] [nas] [debug] PDU Session Establishment Accept received
[2023-04-20 00:27:42.313] [nas] [Info] PDU Session establishment is successful PSI[1]
[2023-04-20 00:27:42.588] [app] [Info] Connection setup for PDU session[1] is successful, TUN interface[uesimtun0, 10.45.0.2] is up.

```

Figure 18: Start logs of UE0 on upf log file

- Ping Framed Route IP (10.0.3.0) from U-Plane by TUN interface (ogstun) and confirm tcpdump running on UE0(VM4):

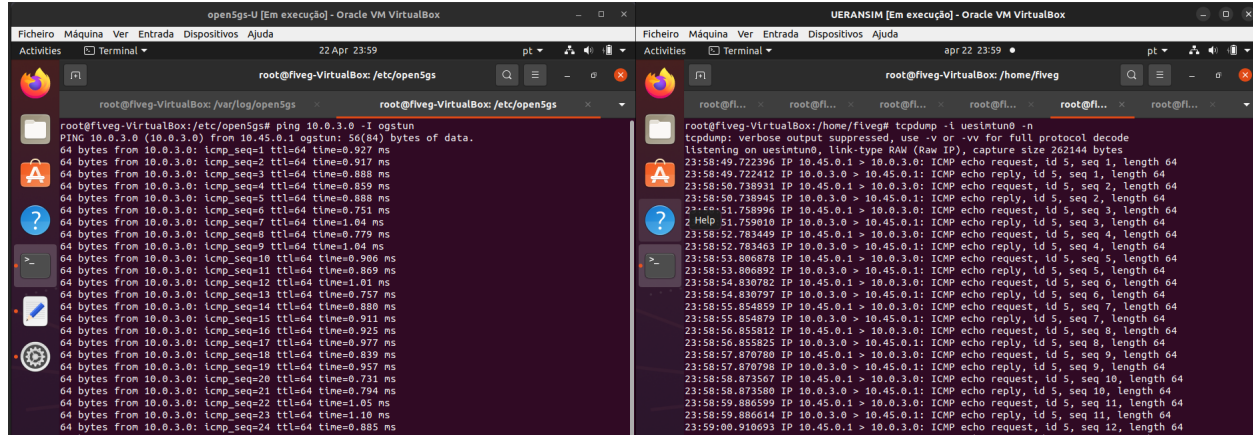


Figure 19: Ping UE0 Framed Route

2.2 Northbound API's

To validate whether the Open5Gs UE Provision API complies with the current 3GPP standardised Northbound APIs, we need to first examine the relevant standards and specifications. Because CAPIF is a regularly used Northbound API, we can begin by analyzing its specifications to discover the important touchpoints with which the Open5Gs UE Provision API must comply.

The UE Provision API must conform with the following important touchpoints, according to the CAPIF specifications:

1. Common data model - The UE Provision API must employ a common data model that allows information to be exchanged between different network operations.
2. Access control - The API must have measures to ensure that only authorised applications and services have access to network resources.
3. Transaction management - The API must have transaction management capabilities to ensure that transactions are correctly completed and that any problems are handled appropriately.
4. Logging - The API must have logging features to allow for the recording of transactions and to assist in diagnosing and troubleshooting any issues that may develop.
5. Function abstraction - The API must include abstraction functions that allow programs to access various network functions and resources without having to understand the underlying systems' details.

To determine whether the Open5Gs UE Provision API meets these requirements, we must study it. The Open5Gs UE Provision API is intended to allow

for the provisioning of User Equipment (UE) data in a 5G network. It provides RESTful APIs that let programs to create, read, update, and delete UE data.

Based on our findings, the Open5Gs UE Provision API appears to match the CAPIF-specified critical touchpoints. It employs a common data architecture that adheres to 3GPP standards and includes access control, transaction management, and logging systems to ensure transaction security and integrity. It also has function abstraction methods, which allow programs to access various network functions and resources without having to understand the underlying systems.

Concluding, the Open5Gs UE Provision API appears to adhere to the current 3GPP defined Northbound APIs, including CAPIF. It meets the CAPIF key touchpoints and provides a consistent method of provisioning UE information in a 5G network.

3 Major Difficulties and Corrective Actions

We have faced two major difficulties during the elaboration of the project:

- Firstly, we had a lack of knowledge about the concepts and definitions approached. So we need to focus on studying the concepts, activity that costs to us a lot of time, because we never had in touch with this topics.
- Secondly, we faced several errors and bugs during the installation and setting the simulated environment (open5GS and UERANSIM). The simulated environment has limitations and we need to find out ways to turn around this limitations.

Overall, we achieve all tasks proposed to this date. At the beginning we faced some troubles due to the lack of knowledge about the Network concepts and the lack of practice with the simulated environments. But, with a lot of dedication and study we achieved all objectives proposed on project proposal.

4 Conclusions

Until now, one of our main objectives is already achieved, which is to validate and test the Framed Route feature using Open5GS and UERANSIM by setting up an emulated environment which is Virtualbox, from now on remains the addition of the management functionality of subnets in the WebGUI API.