

Mobile Cellular Communications (5G)

I. Objectives

The objectives of this laboratory are:

- Identify configuration parameters required for the different components
- Understand the main procedures in a mobile cellular (5G) at the control and data planes by running opensource implementations of the main components

II. Duration

This laboratory should last 2h30.

III. Used tools

This laboratory will use:

- An opensource implementation of a 5G Core: Free5GC [free5Gcore]
- A gNB and UE opensource implementation: UERANSIM [ueransim]
- A VirtualBox VM with both components already installed in the laboratory PCs
- Wireshark also installed in the laboratory PCs

The VM is also available via SSH at port 2222 for user '**ubuntu**' (e.g. '**ssh -p 2222 ubuntu@localhost**', from the hosting machine); password is '**ubuntu**' for users '**ubuntu**' and root.

IV. Network diagram

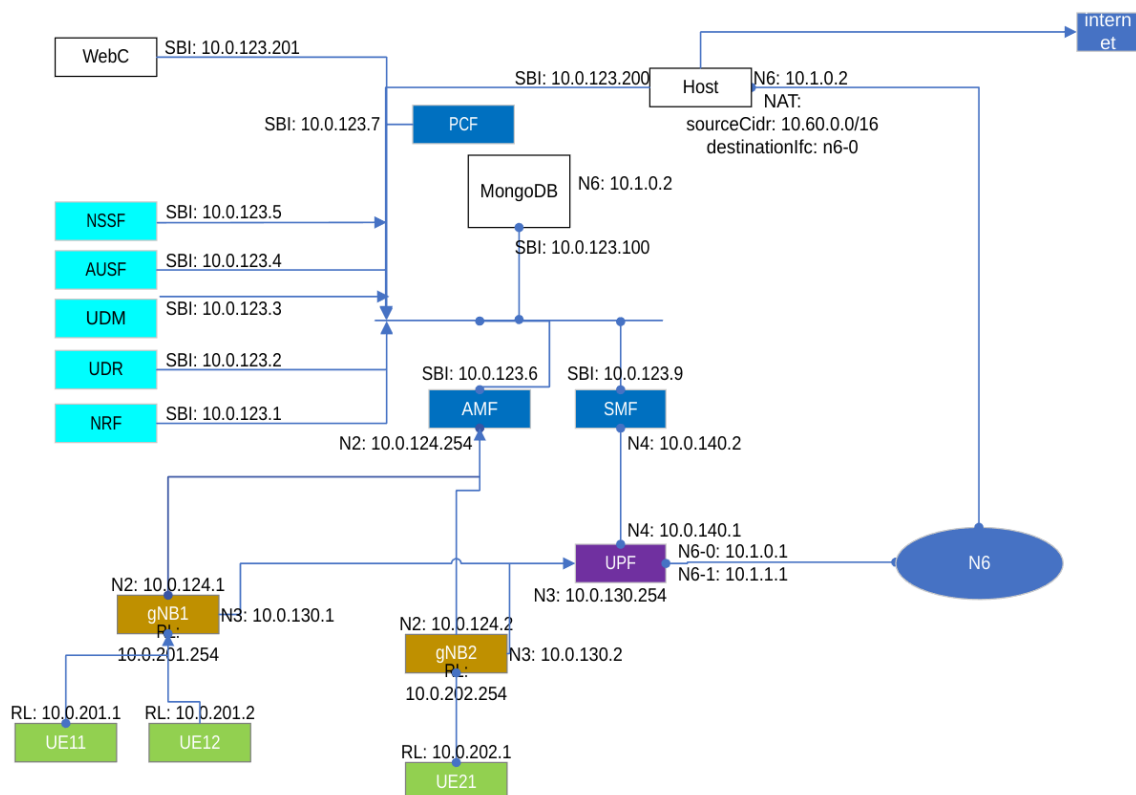


Figure 1: Network diagram

Notes:

1. 5G Core components are represented by dark and light blue boxes, gNBs by brown boxes and UEs by green boxes
2. With UERANSIM, the 5G-NR radio interface ('Radio Link') is emulated over UDP between the UEs (11, 12 and 21, green boxes) and the gNBs (gNB1 and gNB2) they are connected to.
3. IP addresses:
 - a. 10.0.123.0/24: SBI; Core components, Web Console and DB (control plane)
 - b. 10.0.124.0/24: N2 interfaces (control plane)
 - c. 10.0.130.0/24: N3 interfaces (data plane)
 - d. 10.0.140.0/24: N4 interfaces (control plane)
 - e. 10.0.20[1|2].0/24: radio interfaces emulation
 - f. 10.1.[1|2].0/24: N6 DNNs (data plane)
4. Via the 'Host', emulated UEs can reach the Internet A **hosts** file has been added to Wireshark (**/root/.config/wireshark**) for IP addresses resolution so that Wireshark presents components' names instead of IP addresses and you can better interpret the messages exchange (see that file contents in Annex F at the end).
5. The shown MongoDB in the diagram component serves as persistent data repository for the other components while the network is running.

V. Procedure

During the laboratory execution (VirtualBox Virtual Machine), the 5G network components will be started and stop following this order:

1. **5G Network:** 5G Core (2) → gNB1 (3.3) → gNB2 (3.5) →
2. **UEs creation:** → UE provisioning at the 5G Core (4.2) →
3. **UEs start:** → UE11 (4.4) → UE12 (5.5) → UE21 (5.7) →
4. **Stop the system:** → UEs, gNBs and 5G Core (8.1 and 8.3)

Linux Namespaces are used to have each of the nine 5GC Network Functions (AMF, AUSF, NRF, NSSF, PCF, SMF, UDM, UDR, UPF) running inside its own namespace [konrad]. This allows the usage of Wireshark (shall be started with 'sudo') to capture traffic packets exchanged between any two NF, on their own interfaces (you will next figure when selecting capturing interfaces after 5G Core components have been started).

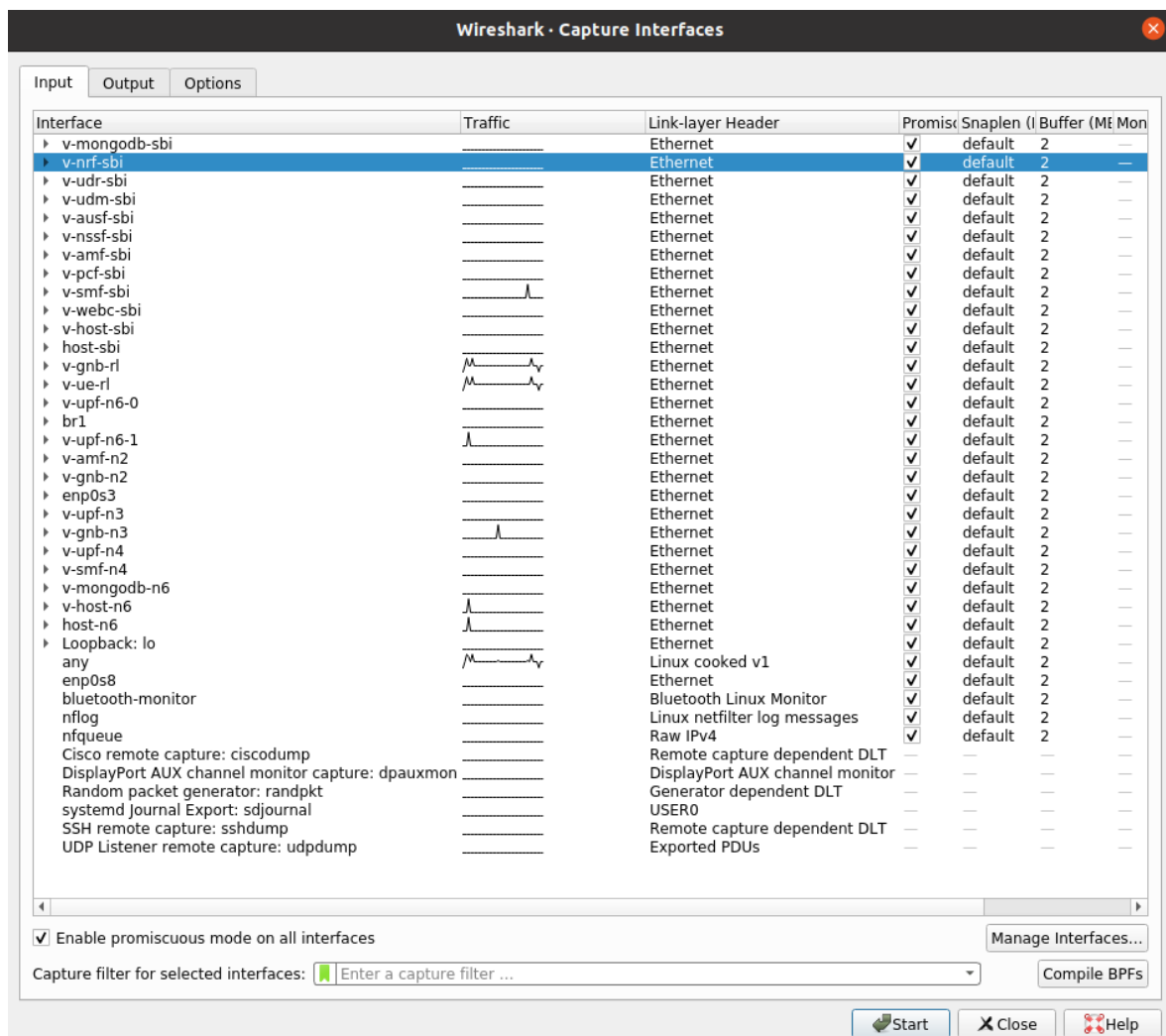


Figure 2: Logical interfaces as seen in Wireshark, after 5G Core start

1. Configurations analysis

- 1) Analyse the yaml configuration files in the list below (1.1.a), located in folder `~/5GLab/netns5g/config` (you may open them with the File Manager) and search for the listed configuration parameters in 1.1.b.
 - a. **Files:**
 - i. 5G Core: **amfcfg.yaml**, **smfcfg.yaml**, **upfcfg.yaml**
 - ii. 5G RAN: **free5gc-gnb1.yaml**, **free5gc-gnb2.yaml**
 - iii. 5G UEs:
 - **free5gc-ue11.yaml**
 - **free5gc-ue12-sl1.yaml**, **free5gc-ue12-sl2.yaml**
 - **free5gc-ue21.yaml**
 - b. **Parameters:**
 - i. MCC and MNC
001 and ?
 - ii. NR Cell Identities and TACs
0x00000010, ? and ?
 - iii. Supported slices at gNB1 and gNB2 (SST+SD)
Slice1: 0x1 and ?; Slice2: ? and 0x112233
 - iv. Supported DNN
internet
 - v. List of SUPIs (UE11, UE12 and UE21)
imsi-001010000000011, imsi-001010000000012 and imsi-001010000000021

2. 5GC start

- 1) Open a terminal window
- 2) Change to directory (~5GLab/netns5g) containing the scripts needed to setup and run the 5G environment
- 3) Initialize environment (create the namespaces and the virtual interfaces)


```
~/5GLab/netns5G$ sudo ./5Gsetup.sh
```

(check its contents; it enables routing, creates nat and forwarding rules, stops firewalling, and calls another script that creates the ns and links)
- 4) Check created namespaces and connecting links


```
~/5GLab/netns5G$ sudo ip netns – lists created namespaces
```


16 namespaces, numbered 0 to 15, one for each of the components shown in the figure above, except the 'Host'

```
~/5GLab/netns5G$ sudo ip link – lists created links
```

33 links are listed, numbered 0 to 57
- 5) Start a Wireshark capture in the interface 'br1' (this will capture all the traffic; you can start other Wireshark instances at specific interfaces, e.g. 'v-amf-sbi')


```
~/5GLab/netns5G$ sudo wireshark &
```
- 6) Start the 5G Core (free5gc)


```
~/5GLab/netns5G$ sudo ./5Gstart.sh
```

At this point 5G Core Network Functions have started, each in its own namespace. 

Observe the script output and identify the order by which 5G Core components have been started.

Order: NRF, UDR, UDM, AUSF, NSSF, AMF, PCF, UPF, SMF, WebUI

Observe the successive interactions with NRF; what is that for?

To register against the NRF (remember the NRF role)

Related the order appear with the existing inter dependencies.
- 7) Stop the capture and identify the involved protocols (to facilitate it, order the capture by the 'Protocol' column by pressing the respective column top); which of those are specific 5G Core protocols?

Protocols: ARP, HTTP2, ICMPv6, MDNS, PFCP, TCP

5G protocols: PFCP and, indirectly, HTTP2
- 8) Identify the dialogs for the 5G protocols (suggestion: apply a display filter to those protocols and check the involved entities)

HTTP2: NRF and all the others, as origin or destination

PFCP: SMF and UPF

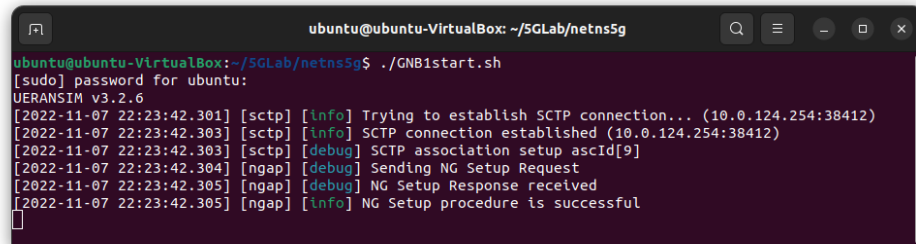
3. gNBs start

- 1) Open a (new) terminal window/tab
- 2) (re)Start Wireshark and start capturing in interface br1 (do not stop wireshark until the end of this section, step 3.7)

\$ **sudo wireshark**

Capture -> Options -> select 'br1'

- 3) From the same directory, start the first gNB (gNB1)
~/5GLab/netns5G\$ **./GNB1start.sh**



```

ubuntu@ubuntu-VirtualBox: ~/5GLab/netns5G
ubuntu@ubuntu-VirtualBox:~/5GLab/netns5G$ ./GNB1start.sh
[sudo] password for ubuntu:
UERANSIM v3.2.6
[2022-11-07 22:23:42.301] [sctp] [info] Trying to establish SCTP connection... (10.0.124.254:38412)
[2022-11-07 22:23:42.303] [sctp] [info] SCTP connection established (10.0.124.254:38412)
[2022-11-07 22:23:42.303] [sctp] [debug] SCTP association setup ascId[9]
[2022-11-07 22:23:42.304] [ngap] [debug] Sending NG Setup Request
[2022-11-07 22:23:42.305] [ngap] [debug] NG Setup Response received
[2022-11-07 22:23:42.305] [ngap] [info] NG Setup procedure is successful

```

- 4) In the live Wireshark capture, observe/note the following:
 - a. Repeat the identification of the involved protocols and the specific 5G ones
All: ARP, NGAP, SCTP, SSL and TCP
5G: NGAP, SCTP
 - b. The SCTP connection setup and later the exchanged heartbeats (suggestion: filter the displayed packets by identified 5G protocols)
 - c. Identify the involved entities
NGAP: gNB1-N2 and AMF-N2
SCTP: gNB1-N2 and AMF-N2
 - d. Detail to the maximum extent, in the Packet Details window, the *NGsetupRequest* and *NGsetupResponse* messages (with mouse right button in 'Packet Details', select 'Expand Subtrees'); Confirm observed values with the ones obtained from the configuration files analysis
- 5) Start the second gNB (gNB2)
~/5GLab/netns5G\$ **./GNB2start.yaml**
- 6) In the live Wireshark compare the new *NGsetupRequest* and *NGsetupResponse* messages with previous ones (gNB1) (suggestion: apply a display filter for the NGCP protocol only and order the capture by the 'Info' column and then move the two pairs of captured packets)
Cell identifiers and the supported slices are different between the two cells
- 7) Observe the logs in the screen (Core, gNB1 and gNB2) and logfiles in: ~/5GLab/netns5G/logs (suggestion: use the 'Files' application to see and open the most recent files, the ones generated until now, executing this 5G Lab)

4. UE creation, registration and default PDU creation

- 1) Open the Free5GC Web Console from the web browser:
 - a. **`http://10.0.123.201:5000`**
 - b. credentials: '**`admin`**'/'**`free5gc`**'
- 2) Create the 3 UEs from the table below ('***New Subscriber***'; see screen capture in the Annexes):

| | UE11 | UE12 | UE21 |
|-------------------|----------------------|----------------------------------|----------------------|
| PLMN ID (MCC/MNC) | 00101 | 00101 | 00101 |
| SUPI (IMSI) | 001010000000011 | 001010000000012 | 001010000000021 |
| SST/SD | 1/010203 (sl1) | 1/010203 (sl1) 2/112233 (sl2) | 1/010203 (sl1) |
| DNN | internet | internet | internet |
| UL/DL AMBR | 10/20 Mbps | 100/200 Mbps | 1/2 Mbps |
| 5QI | 9 | 9 | 9 |
| Note | Will connect to gNB1 | Will connect to gNB1 | Will connect to gNB2 |

Notes:

- 1) Only change the parameters shown in the table and if required
 - a. do not change: Authentication method, K*, Operator Code Type, Operator Code Value*, and SQN*)
 - b. you may search and interpret the other parameters.
- 2) In the Free5GC "New Subscriber" form, delete the second appearing S-NSSAI (*Single Network Slice Selection Assistance Information*) and the second DNN ('internet2')
- 3) Restart Wireshark, keeping the capture in the same interface ('br1')
- 4) Start the first UE (UE11)

~/5GLab/netns5G\$./UE11start.yaml

```

ubuntu@ubuntu-VirtualBox: ~/5GLab/netns5G
ubuntu@ubuntu-VirtualBox:~/5GLab/netns5G$ ./UE11start.sh
UE11start.sh v3.2.6
[2022-11-07 22:24:37.666] [nas] [info] UE switches to state [MM-DEREGISTERED/PLMN-SEARCH]
[2022-11-07 22:24:37.666] [rrc] [debug] New signal detected for cell[1], total [1] cells in coverage
[2022-11-07 22:24:37.667] [nas] [info] Selected plmn[001/01]
[2022-11-07 22:24:37.667] [rrc] [info] Selected cell plmn[001/01] tac[1] category[SUITABLE]
[2022-11-07 22:24:37.667] [nas] [info] UE switches to state [MM-DEREGISTERED/PS]
[2022-11-07 22:24:37.667] [nas] [info] UE switches to state [MM-DEREGISTERED/NORMAL-SERVICE]
[2022-11-07 22:24:37.667] [nas] [debug] Initial registration required due to [MM-DEREG-NORMAL-SERVICE]
[2022-11-07 22:24:37.667] [nas] [debug] UAC access attempt is allowed for Identity[0], category[MO_sig]
[2022-11-07 22:24:37.667] [nas] [debug] Sending Initial Registration
[2022-11-07 22:24:37.667] [nas] [info] UE switches to state [MM-REGISTER-INITIATED]
[2022-11-07 22:24:37.667] [rrc] [debug] Sending RRC Setup Request
[2022-11-07 22:24:37.668] [rrc] [info] RRC connection established
[2022-11-07 22:24:37.668] [rrc] [info] UE switches to state [RRC-CONNECTED]
[2022-11-07 22:24:37.668] [nas] [info] UE switches to state [CN-CONNECTED]
[2022-11-07 22:24:37.683] [nas] [debug] Authentication Request received
[2022-11-07 22:24:37.690] [nas] [debug] Security Mode Command received
[2022-11-07 22:24:37.690] [nas] [debug] Selected integrity[2] ciphering[0]
[2022-11-07 22:24:37.762] [nas] [debug] Registration accept received
[2022-11-07 22:24:37.762] [nas] [info] UE switches to state [MM-REGISTERED/NORMAL-SERVICE]
[2022-11-07 22:24:37.762] [nas] [debug] Sending Registration Complete
[2022-11-07 22:24:37.763] [nas] [info] Initial Registration is successful
[2022-11-07 22:24:37.763] [nas] [debug] Sending PDU Session Establishment Request
[2022-11-07 22:24:37.763] [nas] [debug] UAC access attempt is allowed for Identity[0], category[MO_sig]
[2022-11-07 22:24:38.007] [nas] [debug] PDU Session Establishment Accept received
[2022-11-07 22:24:38.007] [nas] [info] PDU Session establishment is successful PSI[1]
[2022-11-07 22:24:38.016] [app] [info] Connection setup for PDU session[1] is successful, TUN interface[uesimtun0, 10.60.0.2] is up.
  
```

- 5) Observe the states the UE went through, during the process; observe the other messages and its sequence
- 6) Observe the creation of the new TUN interface ('uesimtun0'); in a new terminal window, you can check the creation of this interface 'ue11' and note its IP address (10.60.0.2, in the example)

~/5GLab/netns5G\$ **sudo ip netns exec ue11 ip addr**

```

ubuntu@ubuntu-VirtualBox: ~/5GLab/netns5g
ubuntu@ubuntu-VirtualBox:~/5GLab/netns5g$ sudo ip netns exec ue11 ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
3: uesimtun0: <POINTOPOINT,PROMISC,NOTRAILERS,UP,LOWER_UP> mtu 1400 qdisc fq_codel state UNKNOWN group default qlen 500
    link/none
    inet 10.0.0.2/32 scope global uesimtun0
        valid_lft forever preferred_lft forever
    inet6 fe80::e85b:22f9:e9bd:e373/64 scope link stable-privacy
        valid_lft forever preferred_lft forever
48: ue11-rl@if47: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 4e:1a:5d:d1:fd:ab brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.0.201.1/24 scope global ue11-rl
        valid_lft forever preferred_lft forever
ubuntu@ubuntu-VirtualBox:~/5GLab/netns5g$

```

This interface will be used to exchange the traffic via the 5G network.

- Order the Wireshark displayed capture by 'Protocol' (press the respective column name) and list the relevant protocols
ARP, HTTP2, HTTP2/NAS-5GS, HTTP2/NAS-5GS/NGAP, HTTP2/NGAP, NGAP, NGAP/NAS-5GS, PFCP, SCTP, SSL, SSLv2, TCP, UDP
- 7) Go to 'Statistics' and select 'Conversations' in Wireshark, order by 'IPv4 - 24' and enable 'Name resolution'; observe the 5G dialogs, ordering by 'Address A' and 'Address B'
- 8) Apply a Display Filter to see just NGAP, SCTP and PFCP protocols ("**ngap or pfcip or sctp**")
 - a. Identify the involved 5G control functions (IP addresses are already translated to the functional entity interface, according to the diagram above); identify the dialogs UE-AMF, AMF-SMF, SMF-UPF and their sequence
 - b. Observe the sequence of exchanged messages, looking into their details in the Packet Details window (see, for instance, the '*PFCP Session Establishment Request*' and compare with message '*PFCP Session Modification Request*')
 - c. You may filter the display of messages by protocol and pair of entities, filtering the protocol and their IP addresses (e.g. for HTTP2 between AMF and AUSF: "**ip.addr==10.0.123.4 and ip.addr==10.0.123.6**) and http2")

5. Connectivity

- 1) Start a Wireshark capture in the interface 'upf-n3' and another capture in the interface 'upf-n6-0'
- 2) Apply a Display Filter to see protocols GTP and ICMP
- 3) In a terminal window, start a ping to 8.8.8.8 from UE11


```
~/5GLab/netns5G$ sudo ip netns exec ue11 ping 8.8.8.8 -I uesimtun0
```
- 4) Analyse, in the Wireshark Packet Details, the GTP encapsulation
 - Observe the *Tunnel Endpoint Identifier* (TEID) in both directions of the communication
- 5) In a new Terminal Window/Tab, Start UE12


```
~/5GLab/netns5G$ ./UE12start-sl1.yaml
```

 (check the contents of file `./config/free5gc-ue12-sl1.yaml`)
- 6) Make a ping from UE11 to UE12


```
~/5GLab/netns5G$ sudo ip netns exec ue11 ping <U12 IP addr> -I uesimtun0
```

 - Analyse the observed GTP packets
- 7) In a new Terminal Window/Tab, Start UE21


```
~/5GLab/netns5G$ ./UE12start-sl1.yaml
```

 (check the contents of file `./config/free5gc-ue21.yaml`)
- 8) Make a ping from UE12 to UE21 and observe the exchanged packets at the UPF

6. QoS

- 1) Open a new terminal window
- 2) Start an iperf3 server at the DNN
`$ iperf3 -s`
- 3) Check the TUN interface name and assigned IP address
`$ sudo ip netns exec ue11 ip addr`
- 4) Start an iperf3 client at UE11 towards the server instance and register the achieved bandwidth in the UL and DL directions
`$ sudo ip netns exec ue11 iperf3 -c 10.1.0.2 -B <ue11 IP address> -- uplink`
`$ sudo ip netns exec ue11 iperf3 -c 10.1.0.2 -R -B <ue11 IP address> -- downlink`
- 5) Repeat previous measurements with the other two UEs (UE12 and UE21) and compare the results

7. Slicing

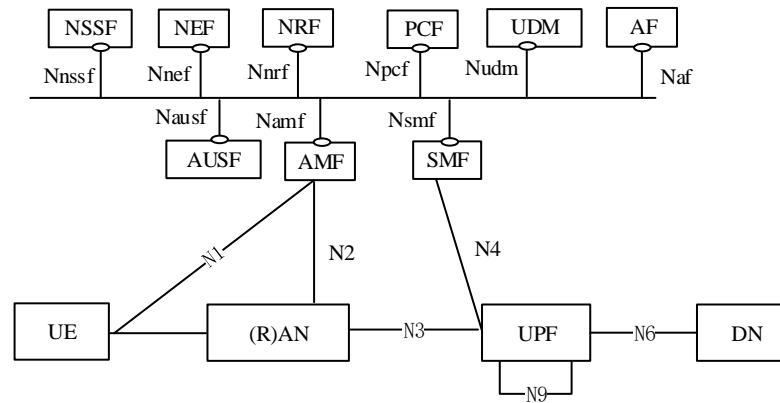
- 1) Stop UE12 (Ctrl-C)
- 2) Restart UE12, now in the second slice (2/112233) with a new configuration file and check the results
`~/5GLab/netns5G$./UE12start-sl2.yaml`
(check the contents of file `./config/free5gc-ue12-sl2.yaml`)
- 3) Observe the newly assigned IP address; what are the changes?
- 4) Make a ping from UE11 to UE12, now in different slices and observe the exchanged packets at the UPF; Is there connectivity?
 - a. Check routing at the UPF namespace
`$ sudo ip netns exec upf ip route`
 - b. Add a new route in the UPF namespace
`$ sudo ip netns exec upf ip route add 10.61.0.0/24 dev upfgtp`
- 5) Repeat the ping above.

8. Stop and reset the environment

- 1) Stop the the UEs, gNB nodes (Ctrl-C), and the 5G Core
- 2) Wait for final processes to close (this takes some seconds, ending with "NRF terminated")
- 3) Delete the namespaces
`~/5GLab/netns5G$ sudo ./5Gcleanup.sh`

Anexes

A. 5G System architecture



B. Example procedure

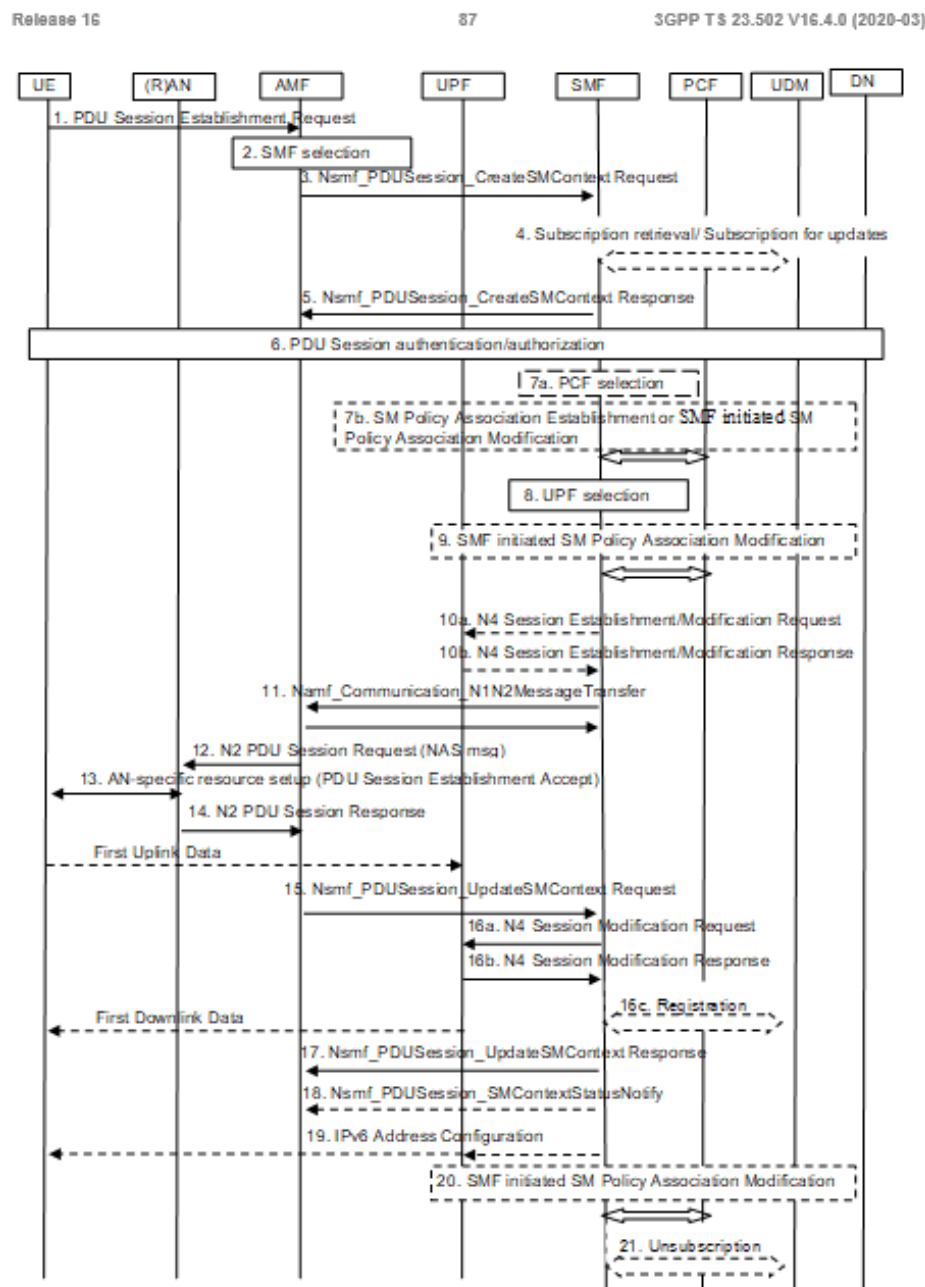
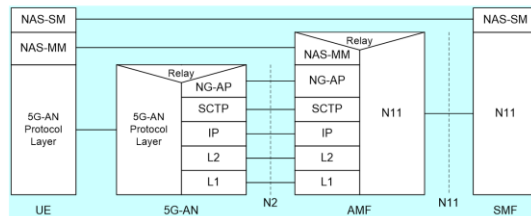
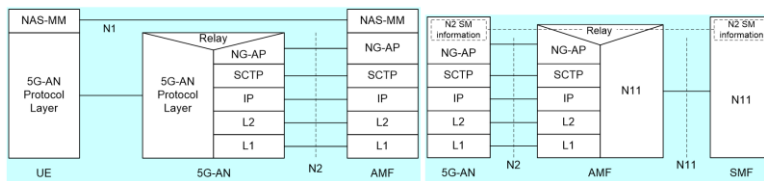
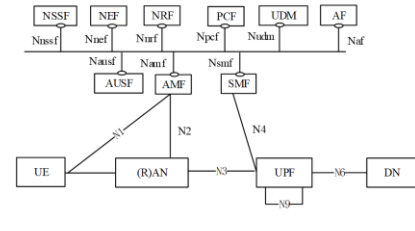
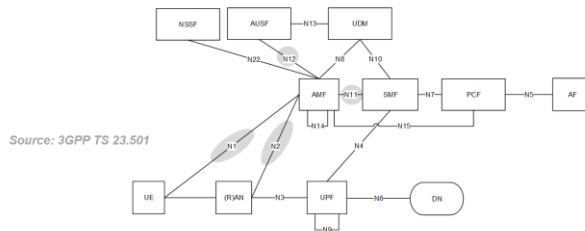


Figure 4.3.2.2.1-1: UE-requested PDU Session Establishment for non-roaming and roaming with local breakout

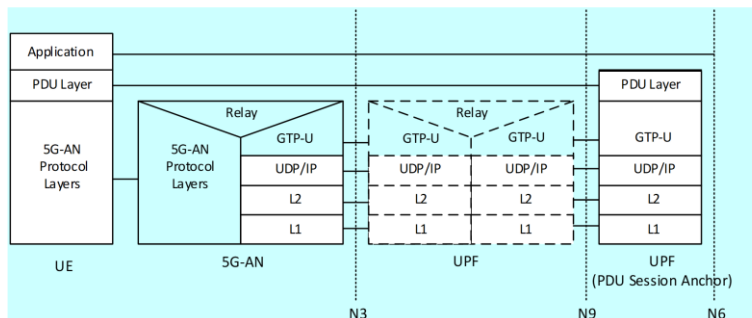
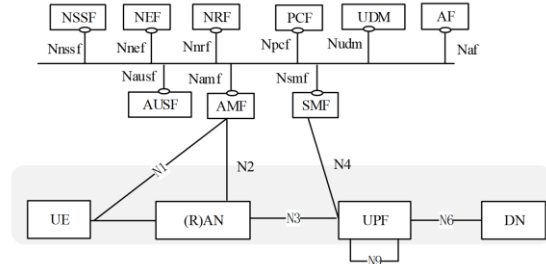
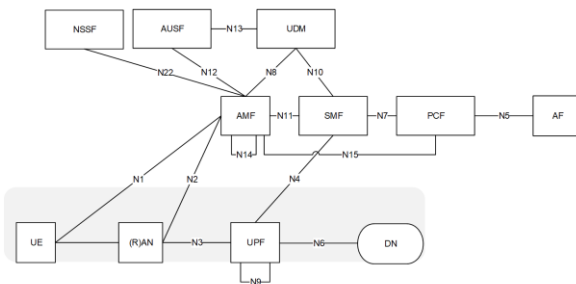
C. 5G Protocol stacks

Protocol stacks – control plane



SCTP: Stream Control Transmission Protocol
PCFP: Packet Forwarding Control Protocol
NG-AP: NG Application Protocol
NAS-MM: NAS Mobility Management
NAS-SM: NAS Session Management
NAS: Non-Access-Stratum

Protocol stacks – user plane



GTP: GPRS Tunnelling Protocol

Source: 3GPP TS 23.501

D. Free5GC New Subscriber creation form (UE11)

The screenshot shows the 'New Subscriber' form in the Free5GC web interface. The browser address bar displays '10.0.123.201:5000/#/subscriber'. The left sidebar contains navigation links: REALTIME STATUS, SUBSCRIBERS (active), ANALYTICS, and TENANT AND USER. The main form area is titled 'New Subscriber' and includes the following sections:

- Subscriber data number (auto-increased with SUP):** A dropdown menu showing '1'.
- PLMN ID:** A text input field containing '00101'.
- SUPI (IMSI):** A text input field containing '001010000000011'.
- Authentication Method:** A dropdown menu showing '5G_AKA'.
- K:** A text input field containing '8ba4f7328f8d09487ccbd7097c6862'.
- Operator Code Type:** A dropdown menu showing 'CPN'.
- Operator Code Value:** A text input field containing '8c27b5d0e692e750f2267a3b14605d'.
- SQN:** A text input field containing '14f82a3f70f6c2'.
- S-NSSAI Configuration:**
 - s-nssai:** A dropdown menu with a blue 'x' button.
 - SST:** A dropdown menu showing '1'.
 - SD:** A text input field containing '010203'.
 - ☒ Default S-NSSAI
- DNN Configurations:**
 - Data Network Name:** A dropdown menu with a blue 'x' button, showing 'Internet'.
 - Uplink AMBR:** A text input field containing '10 Mbps'.
 - Downlink AMBR:** A text input field containing '20 Mbps'.
 - Default SQI:** A dropdown menu showing '9'.
- Flow Rules:** A section with a blue '+' button and a checkbox for 'UP Security'.

At the bottom of the form, there is a blue 'Subscribe' button. The right sidebar contains a 'Log out' link and a 'New Subscriber' button.

E. Example of capture with Wireshark, with addresses resolution and display filter

Wireshark packet capture showing NGAP and SCTP traffic. The packet list shows a sequence of SACK and DATA packets. The packet details pane shows the structure of a DATA chunk and an NGAP PDU.

| No. | Time | Source | Destination | Length | Protocol | Info |
|-----|-------------|---------|-------------|--------|-------------------------|------------------------------------------------------------------------------------------|
| 367 | 7.176821... | gNB1-N2 | AMF-N2 | 190 | NGAP/NAS-5GS | SACK (Ack=2, Arwnd=106496) , UplinkNASTransport |
| 674 | 7.337237... | AMF-N2 | gNB1-N2 | 246 | NGAP/NAS-5GS | SACK (Ack=3, Arwnd=106496) , InitialContextSetupRequest |
| 675 | 7.337538... | gNB1-N2 | AMF-N2 | 98 | NGAP | SACK (Ack=3, Arwnd=106496) , InitialContextSetupResponse |
| 687 | 7.541107... | AMF-N2 | gNB1-N2 | 62 | SCTP | SACK (Ack=4, Arwnd=106496) |
| 688 | 7.541150... | gNB1-N2 | AMF-N2 | 242 | NGAP/NAS-5GS | UplinkNASTransport, UplinkNASTransport |
| 929 | 7.579055... | SMF-SPI | AMF-SPI | 930 | HTTP2/JSON/NAS-5GS/NGAP | DATA[3], JavaScript Object Notation (application/json), PDU session establishment accept |
| 937 | 7.580135... | AMF-N2 | gNB1-N2 | 258 | NGAP/NAS-5GS | SACK (Ack=6, Arwnd=106496) , PDU SessionResourceSetupRequest |
| 942 | 7.582203... | gNB1-N2 | AMF-N2 | 118 | NGAP | SACK (Ack=4, Arwnd=106496) , PDU SessionResourceSetupResponse |
| 951 | 7.583173... | AMF-SPI | SMF-SPI | 703 | HTTP2/JSON/NGAP | DATA[3], JavaScript Object Notation (application/json) |
| 981 | 7.785075... | AMF-N2 | gNB1-N2 | 62 | SCTP | SACK (Ack=7, Arwnd=106496) |

Frame 688: 242 bytes on wire (1936 bits), 242 bytes captured (1936 bits) on interface br1, id 0
 Ethernet II, Src: gNB1-NR (4a:19:ce:1a:48:c0), Dst: AMF-N2 (9a:7b:71:e2:04:2d)
 Internet Protocol Version 4, Src: gNB1-N2 (10.0.124.1), Dst: AMF-N2 (10.0.124.254)
 Stream Control Transmission Protocol, Src Port: 48619 (48619), Dst Port: 38412 (38412)
 Source port: 48619
 Destination port: 38412
 Verification tag: 0x7f9816e7
 [Association index: disabled (enable in preferences)]
 Checksum: 0x00000000 [unverified]
 [Checksum Status: Unverified]
 DATA chunk (ordered, complete segment, TSN: 5, SID: 1, SSN: 5, PPID: 60, payload length: 57 bytes)
 > Chunk type: DATA (0)
 > Chunk flags: 0x03
 Chunk length: 73
 Transmission sequence number (relative): 5
 Transmission sequence number (absolute): 3845702161
 Stream identifier: 0x0001
 Stream sequence number: 5
 Payload protocol identifier: NGAP (60)
 Chunk padding: 000000
 NG Application Protocol (UplinkNASTransport)
 NGAP-PDU: initiatingMessage (0)
 initiatingMessage
 procedureCode: id-UplinkNASTransport (46)
 criticality: ignore (1)
 value
 UplinkNASTransport
 protocolIEs: 4 items
 Item 0: id-AMF-UE-NGAP-ID
 ProtocolIE-Field
 id: id-AMF-UE-NGAP-ID (10)
 criticality: reject (0)
 value
 AMF-UE-NGAP-ID: 1
 Item 1: id-RAN-UE-NGAP-ID
 ProtocolIE-Field
 id: id-RAN-UE-NGAP-ID (85)
 criticality: reject (0)
 value

Wireshark - Packet 937 - 2022.11.09.free5GC.UE11start.pcapng

```

> Frame 937: 258 bytes on wire (2064 bits), 258 bytes captured (2064 bits) on interface br1, id 0
> Ethernet II, Src: AMF-N2 (9a:7b:71:e2:04:2d), Dst: gNB1-NR (4a:19:ce:1a:48:c0)
> Internet Protocol Version 4, Src: AMF-N2 (10.0.124.254), Dst: gNB1-N2 (10.0.124.1)
> Stream Control Transmission Protocol, Src Port: 38412 (38412), Dst Port: 48619 (48619)
▼ NG Application Protocol (PDUSessionResourceSetupRequest)
  ▼ NGAP-PDU: InitiatingMessage (0)
    ▼ InitiatingMessage
      procedureCode: id-PDUSessionResourceSetup (29)
      criticality: reject (0)
      ▼ value
        ▼ PDUSessionResourceSetupRequest
          ▼ protocolIEs: 4 items
            ▼ Item 0: id-AMF-UE-NGAP-ID
              ▼ ProtocolIE-Field
                id: id-AMF-UE-NGAP-ID (10)
                criticality: reject (0)
                ▼ value
                  AMF-UE-NGAP-ID: 1
            ▼ Item 1: id-RAN-UE-NGAP-ID
              ▼ ProtocolIE-Field
                id: id-RAN-UE-NGAP-ID (85)
                criticality: reject (0)
                ▼ value
                  RAN-UE-NGAP-ID: 1
            ▼ Item 2: id-PDUSessionResourceSetupListSUReq
              ▼ ProtocolIE-Field
                id: id-PDUSessionResourceSetupListSUReq (74)
                criticality: reject (0)
                ▼ value
                  ▼ PDUSessionResourceSetupListSUReq: 1 item
                    ▼ Item 0
                      ▼ PDUSessionResourceSetupItemSUReq
                        pDUSessionID: 1
                        ▼ pDUSessionNAS-PDU: 7e02cad24c2a027e00680100432e0101c211000901000631310101ff090606001406000a...
                          ▼ Non-Access-Stratum 5GS (NAS)PDU
                            ▼ Security protected NAS 5GS message
                              Extended protocol discriminator: 5G mobility management messages (126)
                              0000 .... = Spare Half Octet: 0
                              .... 0010 = Security header type: Integrity protected and ciphered (2)
                              Message authentication code: 0xcad24c2a
                              Sequence number: 2
                              Encrypted data
                            ▼ s-NSSAI
                              sST: 01
                              sD: 010203
                            ▼ pDUSessionResourceSetupRequestTransfer: 000004008200090c0131200020989680008b000a01f00a0082fe00000001008600010000...
                              ▼ PDUSessionResourceSetupRequestTransfer
                                ▼ protocolIEs: 4 items
                                  ▼ Item 0: id-PDUSessionAggregateMaximumBitRate
                                    ▼ ProtocolIE-Field
                                      id: id-PDUSessionAggregateMaximumBitRate (130)
                                      criticality: reject (0)
                                      ▼ value
                                        ▼ PDUSessionAggregateMaximumBitRate
                                          pDUSessionAggregateMaximumBitRateDL: 20000000bits/s
                                          pDUSessionAggregateMaximumBitRateUL: 10000000bits/s
                                  ▼ Item 1: id-UL-NGU-UP-TNLIInformation
                                    ▼ ProtocolIE-Field
                                      id: id-UL-NGU-UP-TNLIInformation (139)
                                      criticality: reject (0)
                                      ▼ value
                                        ▼ UPTransportLayerInformation: gTPTunnel (0)
                                          ▼ gTPTunnel
                                            ▼ transportLayerAddress: 0a0082fe [bit length 32, 0000 1010 0000 0000 1000 0010 1111 1110 decimal value 167805694]
                                              TransportLayerAddress (IPv4): 10.0.130.254 (10.0.130.254)
                                            gTP-TEID: 00000001
                                  ▼ Item 2: id-PDUSessionType
                                    ▼ ProtocolIE-Field
                                      id: id-PDUSessionType (134)
                                      criticality: reject (0)
                                      ▼ value
                                        PDUSessionType: ipv4 (0)
                                  ▼ Item 3: id-QosFlowSetupRequestList
                                    ▼ ProtocolIE-Field
                                      id: id-QosFlowSetupRequestList (136)
                                      criticality: reject (0)
                                      ▼ value
                                        ▼ QosFlowSetupRequestList: 1 item
                                          ▼ Item 0
                                            ▼ QosFlowSetupRequestItem
                                              qosFlowIdentifier: 9
                                              ▼ qosFlowLevelQosParameters
                                                ▼ qosCharacteristics: nonDynamic5QI (0)
                                                  ▼ nonDynamic5QI
                                                    fiveQI: 9
                                                  ▼ allocationAndRetentionPriority
                                                    priorityLevelARP: 15
                                                    pre-emptionCapability: shall-not-trigger-pre-emption (0)
                                                    pre-emptionVulnerability: not-pre-emptable (0)
                                  ▼ Item 3: id-UEAggregateMaximumBitRate
                                    ▼ ProtocolIE-Field
                                      id: id-UEAggregateMaximumBitRate (110)
                                      criticality: ignore (1)
                                      ▼ value
                                        ▼ UEAggregateMaximumBitRate
                                          uEAggregateMaximumBitRateDL: 2000000000bits/s
                                          uEAggregateMaximumBitRateUL: 1000000000bits/s
          ▼ Item 3: id-UEAggregateMaximumBitRate
            ▼ ProtocolIE-Field
              id: id-UEAggregateMaximumBitRate (110)
              criticality: ignore (1)
              ▼ value
                ▼ UEAggregateMaximumBitRate
                  uEAggregateMaximumBitRateDL: 2000000000bits/s
                  uEAggregateMaximumBitRateUL: 1000000000bits/s

```

☐ Show packet bytes

Close Help

F. Hosts file

```
#5G Core
10.0.123.1      NRF-SBI
10.0.123.2      UDR-SBI
10.0.123.3      UDM-SBI
10.0.123.4      AUSF-SBI
10.0.123.5      NSSF-SBI
10.0.123.6      AMF-SBI
10.0.123.7      PCF-SBI
10.0.123.9      SMF-SBI
10.0.123.100    MongoDB-SBI
10.0.123.201    WebConsole

10.0.124.254    AMF-N2
10.0.124.1      gNB1-N2
10.0.124.2      gNB2-N2

10.0.140.2      SMF-N4
10.0.140.1      UPF-N4

#5G dataplane
10.1.0.1         UPF-N6
10.1.0.1         Host-N6

#RAN1
10.0.201.1       UE11-NR
10.0.201.2       UE12-NR
10.0.201.254     gNB1-NR

#RAN2
10.0.202.1       UE11-NR
10.0.202.254     gNB1-NR
```


G. Useful links

- Free5GC:
 - [free5Gcore] <https://www.free5gc.org/>
 - [free5gcwiki] <https://github.com/free5gc/free5gc/wiki>
 - [konrad] <https://github.com/konradkar2/netns5g>
- UERANSIM:
 - [ueransim] <https://github.com/aligungr/UERANSIM/wiki>
- 3GPP
 - [3gpp] <https://www.3gpp.org>