IT.3503 - Architecture Virtualisée

TP 4: Fifth Generation Mobile Technologies

GUO Xiaofan

YIN Chenghao

Environment Setup

1 Install Docker Engine

```
gxf@gxf:~$ docker --version
Docker version 27.4.0, build bde2b89
gxf@gxf:~$ sudo docker run hello-world

Hello from Docker!
```

2 Install Docker Compose

```
gxf@gxf:~$ docker compose version
Docker Compose version v2.30.3-desktop.1
```

3 Install free5gc/gtp5g kernel module

```
gxf@gxf:~/gtp5g$ lsmod | grep gtp5g
gtp5g                 151552  0
udp_tunnel             32768  2 gtp5g,vxlan
```

Configuration

1 Get free5gc-compose

```
gxf@gxf:~/gtp5g/free5gc-compose/config$ ls
amfcfg.yaml    n3iwfcfg.yaml    nrfcfg.yaml    tngfcfg.yaml     uecfg.yaml       upf-iptables.sh
ausfcfg.yaml   n3iwf-ipsec.sh   nssfcfg.yaml   udmcfg.yaml      uerouting.yaml   webuicfg.yaml
chfcfg.yaml    n3uecfg.yaml     pcfcfg.yaml    udrcfg.yaml      ULCL
gnbcfg.yaml    nefcfg.yaml      smfcfg.yaml    uecfg-ulcl.yaml  upfcfg.yaml
```

2 Explore the configuration

2.1 What is the configured Public Land Mobile Network (PLMN) ID? *nano amfcfg.yaml*

```
  supportTaiList:  # the TAI (Tracking Area Identifier) list supported by this AMF
    - plmnId: # Public Land Mobile Network ID, <PLMN ID> = <MCC><MNC>
        mcc: 208 # Mobile Country Code (3 digits string, digit: 0~9)
        mnc: 93 # Mobile Network Code (2 or 3 digits string, digit: 0~9)
      tac: 000001 # Tracking Area Code (3 bytes hex string, range: 000000~FFFFFF)
```

- PLMN ID：20893

2.2 What are the configured 5G slices? explain the "sst" and "sd"? *nano smfcfg.yaml*

```
- sNssai: # S-NSSAI (Single Network Slice Selection Assistance Information)
    sst: 1 # Slice/Service Type (uinteger, range: 0~255)
    sd: 010203 # Slice Differentiator (3 bytes hex string, range: 000000~FFFFFF)
```

sst (Slice Service Type): defines the service type of the slice, for example:

- enhanced mobile broadband (eMBB)
- ultra-reliable low latency communication (URLLC)
- massive machine type communication (mMTC)

sd (Slice Differentiator): used to further distinguish different slices under the same sst.

2.3 What are the integrity algorithms used by the Access and Mobility management Function? *nano amfcfg.yaml*

```
integrityOrder: # the priority of integrity algorithms
  - NIA2
  # - NIA0
```

- **integrityOrder:** NIA2, can use NIA0

2.4 What are the ciphering algorithms used by the Access and Mobility management Function? *nano amfcfg.yaml*

```
cipheringOrder: # the priority of ciphering algorithms
  - NEA0
  # - NEA2
```

- **cipheringOrder:** NEA0, can use NEA2

2.5 What are the supported PLMN IDs by the AUthentication Server Function? what do you notice if you compare it to AMF PLMN ID? *nano ausfcfg.yaml*

```
plmnSupportList: # the PLMNs (Public Land Mobile Network) list supported by this AUSF
  - mcc: 208 # Mobile Country Code (3 digits string, digit: 0~9)
    mnc: 93  # Mobile Network Code (2 or 3 digits string, digit: 0~9)
```

- The PLMN ID of AMF and AUSF is 20893, the configuration is consistent.

2.6 What is the configured Tracking Area Code for the gNodeB? *nano gnbcfg.yaml*

```
tac: 1 # Tracking Area Code
```

- TAC（Tracking Area Code）: 1

2.7 What are the 5G slices supported by the gNodeB ? *nano gnbcfg.yaml*

```
# List of supported S-NSSAIs by this gNB
slices:
  - sst: 0x1
    sd: 0x010203
  - sst: 0x1
    sd: 0x112233
```

- Same as smfcfg.yaml.

2.8 What is the N2 service port of the AMF? *amfcfg.yaml*

```
ngapPort: 38412 # the SCTP port listened by NGAP
```

- ngapPort indicates the SCTP service port that AMF uses to listen on the N2 interface (NGAP protocol): 38412

2.9 What is the service port of the Network Repository Function? *nrfcfg.yaml*

```
sbi: # Service-based interface information
  scheme: http # the protocol for sbi (http or https)
  registerIPv4: nrf.free5gc.org # IP used to serve NFs or register to another NRF
  bindingIPv4: nrf.free5gc.org  # IP used to bind the service
  port: 8000 # port used to bind the service
```

- NRF port: 8000

2.10 What are the available Data Network Names? *upfcfg.yaml*

```
dnnList:
  - dnn: internet # Data Network Name
    cidr: 10.60.0.0/16 # Classless Inter-Domain Routing for assigned IPv4 pool of UE
    # natifname: eth0
  - dnn: internet # Data Network Name
    cidr: 10.61.0.0/16 # Classless Inter-Domain Routing for assigned IPv4 pool of UE
    # natifname: eth0
```

- Data Network Names (DNN): internet.: 10.60.0.0/16 & 10.61.0.0/16

2.11 What are the IP pools for each Data network in each slice? *upfcfg.yaml*

```
dnnList:
  - dnn: internet # Data Network Name
    cidr: 10.60.0.0/16 # Classless Inter-Domain Routing for assigned IPv4 pool of UE
    # natifname: eth0
  - dnn: internet # Data Network Name
    cidr: 10.61.0.0/16 # Classless Inter-Domain Routing for assigned IPv4 pool of UE
    # natifname: eth0
```

- cidr: 10.60.0.0/16 & 10.61.0.0/16

2.12 What is the Subscription Permanent Identifier of the UE default configuration? *nano uecfg.yaml*

```
# IMSI number of the UE. IMSI = [MCC|MNC|MSISDN] (In total 15 digits)
supi: "imsi-208930000000001"
```

- the Subscription Permanent Identifier of the UE is: imsi-208930000000001

2.13 What are the integrity algorithms supported by the UE? *nano uecfg.yaml*

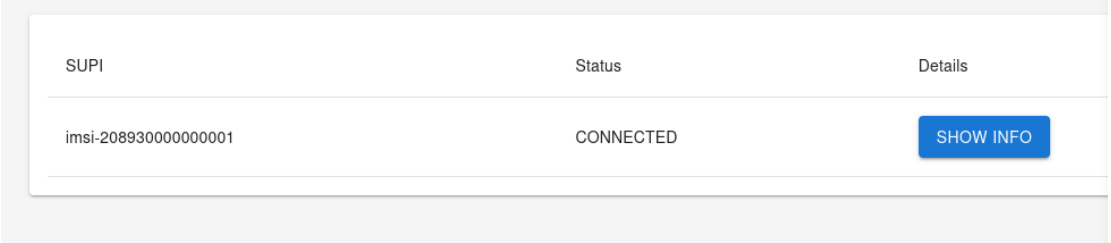
```
# Supported integrity algorithms by this UE
integrity:
  IA1: true
  IA2: true
  IA3: true
```

- Integrity algorithm: IA1, IA2, IA3 (all supported)

2.14 What are the ciphering algorithms supported by the UE? *nano uecfg.yaml*

```
# Supported encryption algorithms by this UE
ciphering:
  EA1: true
  EA2: true
  EA3: true
```

- ciphering algorithm: EA1, EA2, EA3 (all supported)



3 Deployment of 5G mobile network

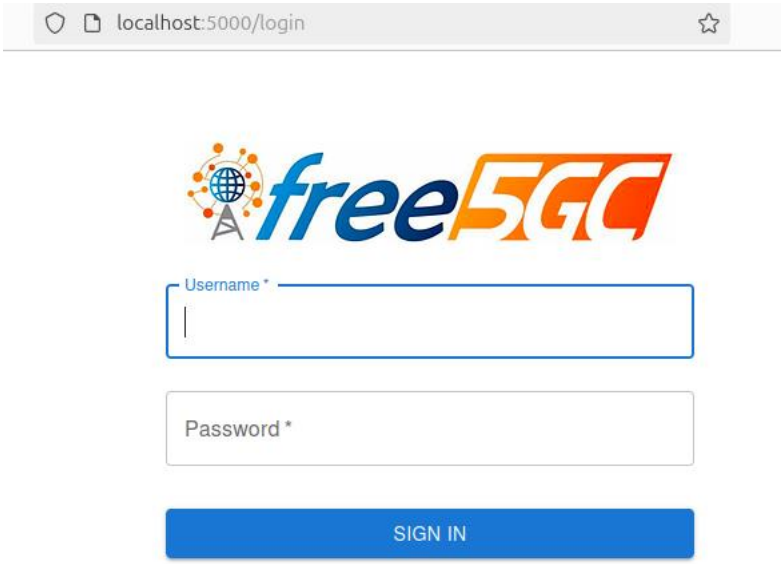
3.1 NFs

```
gxf@gxf:~/gtp5g/free5gc-compose$ docker compose up -d
WARN[0000] /home/gxf/gtp5g/free5gc-compose/docker-compos
gnored, please remove it to avoid potential confusion
[+] Running 99/33
 ✔ free5gc-nrf Pulled
 ✔ n3iwue Pulled
 ✔ free5gc-nssf Pulled
 ✔ free5gc-n3iwf Pulled
 ✔ free5gc-webui Pulled
 ✔ ueransim Pulled
 ✔ free5gc-nef Pulled
 ✔ free5gc-pcf Pulled
 ✔ free5gc-udm Pulled
 ✔ free5gc-tngf Pulled
 ✔ free5gc-smf Pulled
 ✔ db Pulled
 ✔ free5gc-upf Pulled
 ✔ free5gc-chf Pulled
 ✔ free5gc-amf Pulled
 ✔ free5gc-udr Pulled
```

```
[+] Running 19/19
 ✔ Network free5gc-compose_privnet  Created
 ✔ Volume "free5gc-compose_dbdata"  Created
 ✔ Container mongodb                Started
 ✔ Container upf                    Started
 ✔ Container nrf                    Started
 ✔ Container udm                    Started
 ✔ Container udr                    Started
 ✔ Container nssf                   Started
 ✔ Container webui                  Started
 ✔ Container smf                    Started
 ✔ Container ausf                   Started
 ✔ Container amf                    Started
 ✔ Container nef                    Started
 ✔ Container pcf                    Started
 ✔ Container chf                    Started
 ✔ Container tngf                   Started
 ✔ Container n3iwf                  Started
 ✔ Container ueransim               Started
 ✔ Container n3iwue                 Started
```

```
gxf@gxf:~/gtp5g/free5gc-compose$ docker compose ps
WARN[0000] /home/gxf/gtp5g/free5gc-compose/docker-compose.yaml: the attribute `version` is obsolete, it will be i
gnored, please remove it to avoid potential confusion
NAME       IMAGE                     COMMAND                  SERVICE         CREATED          STATUS          PO
RTS
amf        free5gc/amf:v3.4.4        "./amf -c ./config/a…"   free5gc-amf     35 seconds ago   Up 29 seconds   80
00/tcp
ausf       free5gc/ausf:v3.4.4       "./ausf -c ./config/…"   free5gc-ausf    35 seconds ago   Up 27 seconds   80
00/tcp
chf        free5gc/chf:v3.4.4        "./chf -c ./config/c…"   free5gc-chf     34 seconds ago   Up 17 seconds   21
22/tcp, 8000/tcp
mongodb    mongo:3.6.8               "docker-entrypoint.s…"   db              36 seconds ago   Up 32 seconds   27
017/tcp
n3iwf      free5gc/n3iwf:v3.4.4      "./n3iwf -c ./config…"   free5gc-n3iwf   34 seconds ago   Up 19 seconds
n3iwue     free5gc/n3iwue:latest     "bash -c 'ip route d…"   n3iwue          34 seconds ago   Up 15 seconds
nef        free5gc/nef:latest        "./nef -c ./config/n…"   free5gc-nef     35 seconds ago   Up 27 seconds   80
00/tcp
nrf        free5gc/nrf:v3.4.4        "./nrf -c ./config/n…"   free5gc-nrf     35 seconds ago   Up 31 seconds   80
00/tcp
nssf       free5gc/nssf:v3.4.4       "./nssf -c ./config/…"   free5gc-nssf    35 seconds ago   Up 26 seconds   80
00/tcp
pcf        free5gc/pcf:v3.4.4        "./pcf -c ./config/p…"   free5gc-pcf     35 seconds ago   Up 28 seconds   80
00/tcp
smf        free5gc/smf:v3.4.4        "./smf -c ./config/s…"   free5gc-smf     35 seconds ago   Up 28 seconds   80
00/tcp
udm        free5gc/udm:v3.4.4        "./udm -c ./config/u…"   free5gc-udm     35 seconds ago   Up 27 seconds   80
00/tcp
udr        free5gc/udr:v3.4.4        "./udr -c ./config/u…"   free5gc-udr     35 seconds ago   Up 28 seconds   80
00/tcp
ueransim   free5gc/ueransim:latest   "./nr-gnb -c ./confi…"   ueransim        34 seconds ago   Up 24 seconds
upf        free5gc/upf:v3.4.4        "bash -c './upf-ipta…"   free5gc-upf     36 seconds ago   Up 33 seconds
webui      free5gc/webui:v3.4.4      "./webui -c ./config…"   free5gc-webui   35 seconds ago   Up 26 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
```

3.2 WebUI

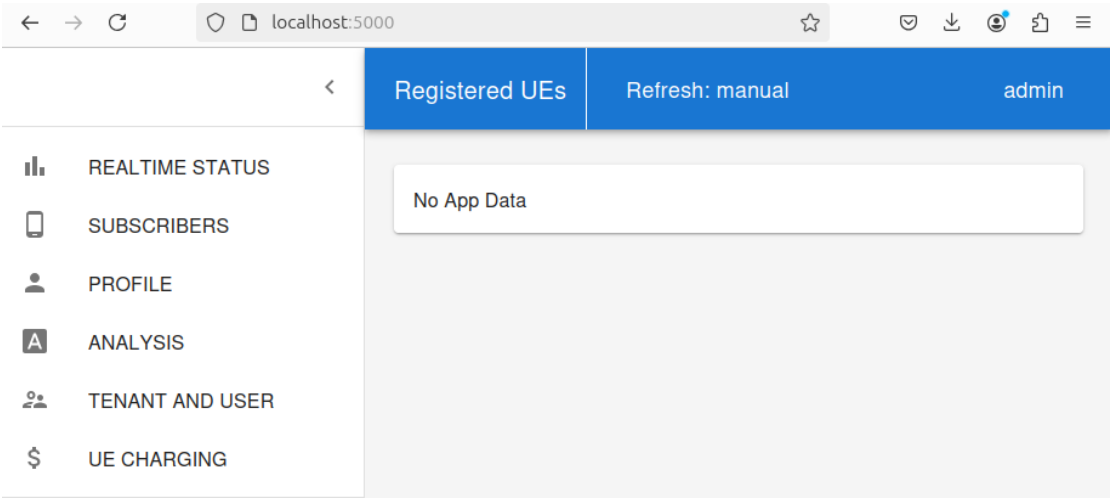


4 Attach a UE, capture traffic, and analyze

4.1 Attach a UE

1) Create in "subscribers" and ensure that the SUPI (IMSI), Key, OPC, AMF, DNN, S-NSSAI, SST, and SD information are the same as those in the

config/uexfg.yaml file of docker.

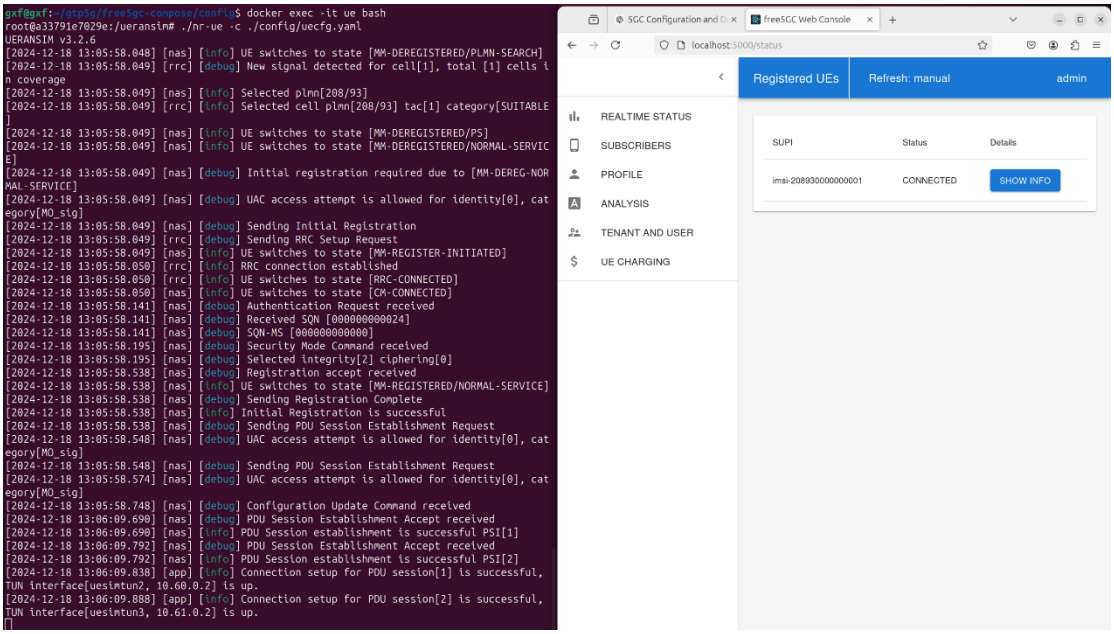


2) Add UE service to docker-compose.yaml

3) Restart and enter the UE container and start the UE simulator.

4) Observe the UE log to ensure that the registration is successful.

5) In the REALTIME STATUS page in the Free5GC WebUI, confirm that the UE has been registered.



4.2 Capturing traffic

1) Enter the UPF container, install tcpdump and capture traffic:

```
root@c73ead55b34a:/free5gc# tcpdump -i any -w /upf_traffic.pcap
tcpdump: data link type LINUX_SLL2
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144
bytes
```

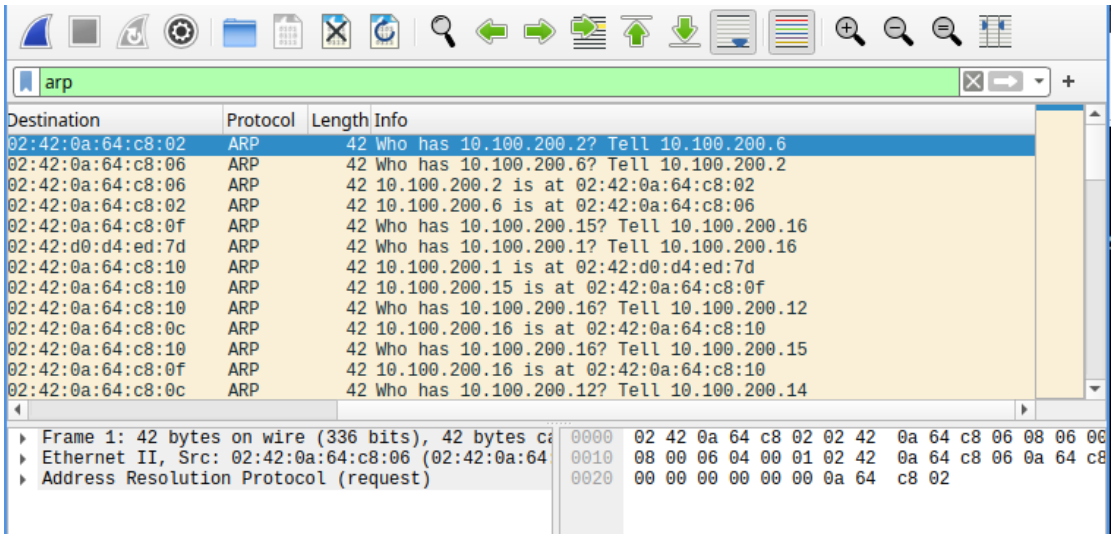
2) Enable IP forwarding within the VM

```
gxf@gxf:~$ sudo sysctl -w net.ipv4.ip_forward=1
[sudo] password for gxf:
net.ipv4.ip_forward = 1
gxf@gxf:~$ sudo nano /etc/sysctl.conf
gxf@gxf:~$ sudo sysctl -p
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-iptables = 1
net.bridge.bridge-nf-call-ip6tables = 1
gxf@gxf:~$
```

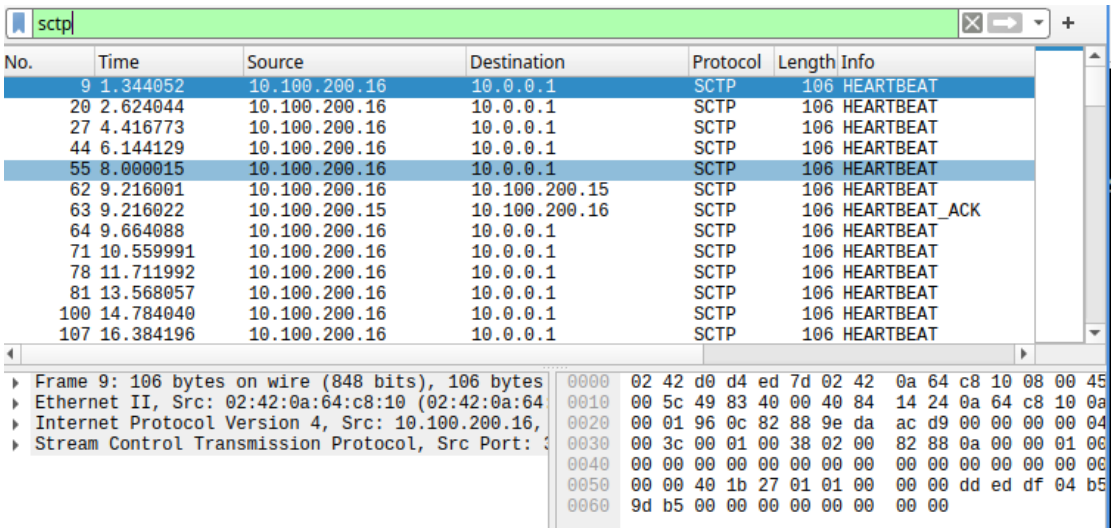
```
gxf@gxf:~/gtp5g/free5gc-compose$ docker exec -it upf bash
root@c73ead55b34a:/free5gc# ip route
default via 10.100.200.1 dev eth0
10.60.0.0/16 dev upfgtp proto static
10.61.0.0/16 dev upfgtp proto static
10.100.200.0/24 dev eth0 proto kernel scope link src 10.100.200.3
```

Capture the traffic on the *br-free5gc* network interface using a command and write it to the free5gc_traffic.pcap file and open the captured free5gc_traffic.pcap file in Wireshark to analyze the packet contents.

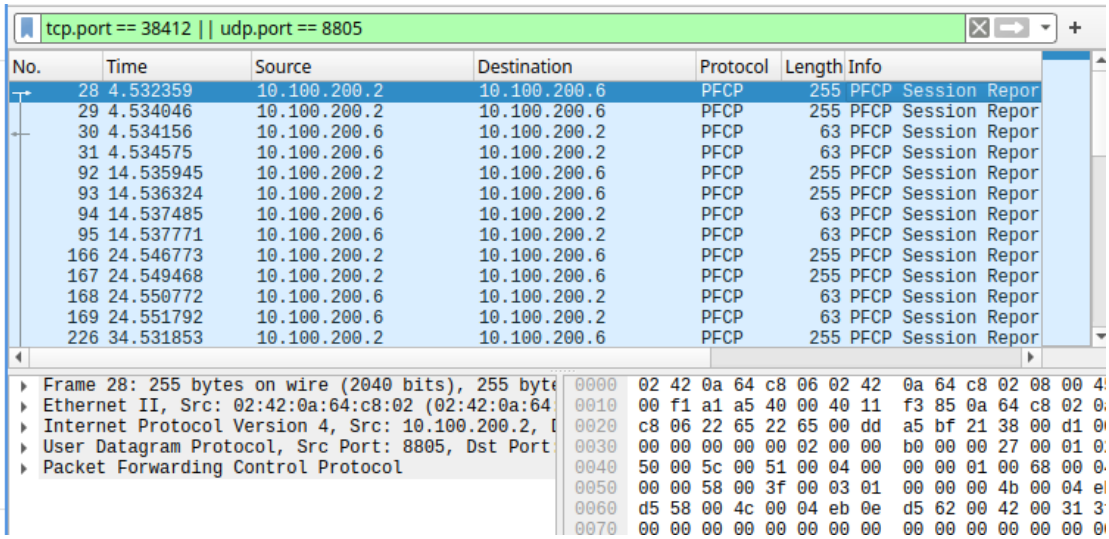
```
palpitate30@palpitate30-virtualbox:~$ sudo tcpdump -i br-free5gc -w fr
ee5gc_traffic.pcap
tcpdump: listening on br-free5gc, link-type EN10MB (Ethernet), snapsho
t length 262144 bytes
```



Analysis: ARP protocol broadcast and response packets, which are typically used to resolve IP addresses to MAC addresses. The ARP packets indicate that the IP to MAC mapping between the Free5GC core modules is working properly, indicating that basic communication between network interfaces is available.



Analysis: SCTP is the main communication protocol between AMF and gNB, used to deliver signaling messages. HEARTBEAT represents the maintenance message of SCTP connection. Paired HEARTBEAT and HEARTBEAT_ACK packets indicate that the SCTP link has been established, realizing signaling communication between AMF and gNB modules.



Analysis: PFCP traffic flows from the source IP: 10.100.200.2 (SMF) to the destination IP: 10.100.200.6 (UPF), used for session management and user data forwarding configuration. The Session Report Request indicates that the SMF is sending a session report request to the UPF, demonstrating that communication between the SMF and UPF is functioning properly.

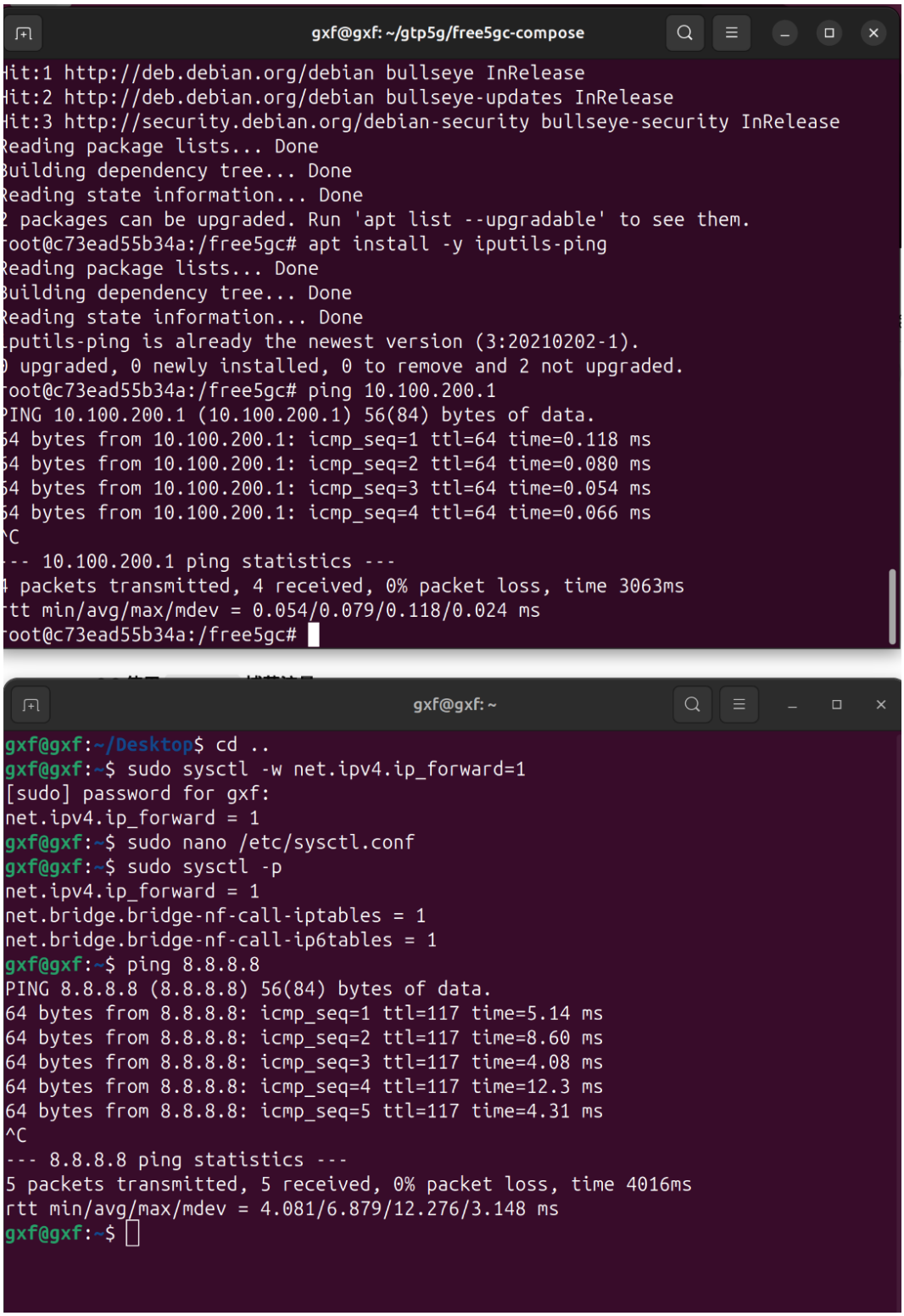
Monitor the transfer directly in the terminal:

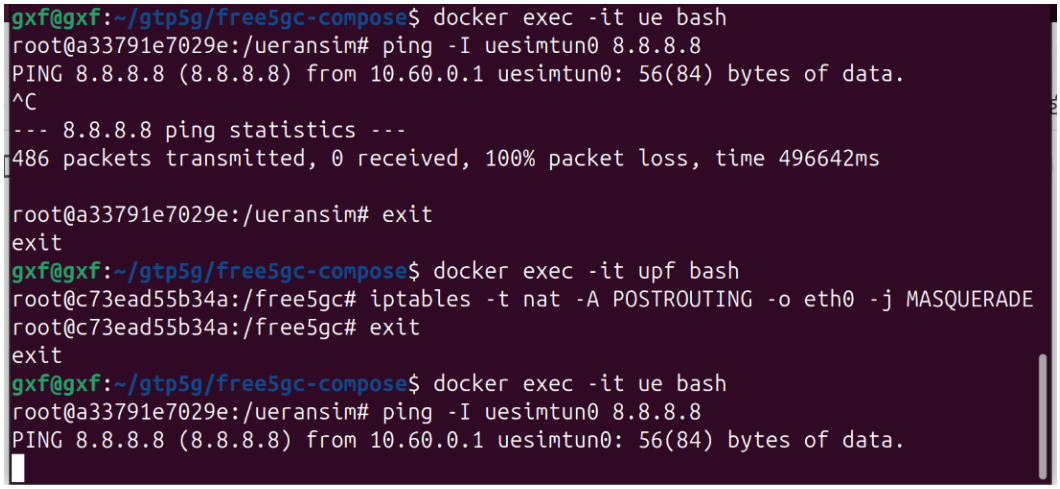
```
gxf@gxf:~/gtp5g/free5gc-compose$ docker exec -it ue bash
root@a33791e7029e:/ueransim# ping -I uesimtun0 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 10.60.0.1 uesimtun0: 56(84) bytes of data.
```

Analysis:

- UPF configures NAT: iptables has successfully configured the MASQUERADE forwarding rule.
- The UE has started sending packets (ICMP requests).
- However, no packets are returned, and no response can be received after the traffic is sent out.

After confirming that IP forwarding is enabled and checking the network configuration:





Analysis:

- The UPF container can ping the default gateway 10.100.200.1 normally.
- The host has successfully enabled IP forwarding and can ping the public network 8.8.8.8 normally.
- But the UE still cannot successfully ping 8.8.8.8
- The problem lies in the UPF traffic forwarding and NAT configuration.

Solution:

- If can see ICMP request packets but no response packets: the problem is in the network egress (host or external network).
- If there is no ICMP packet: the problem is in the UPF NAT configuration, and the traffic is not forwarded successfully.

```
    0     0 MASQUERADE  all  --  *      eth0    0.0.0.0/0            0.0.0.0/0

    0     0 MASQUERADE  all  --  *      eth0    0.0.0.0/0            0.0.0.0/0


Chain DOCKER_OUTPUT (1 references)
 pkts bytes target     prot opt in     out     source               destination

    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            127.0.0.11
        to:127.0.0.11:34457
  614 44442 DNAT       udp  --  *      *       0.0.0.0/0            127.0.0.11
        to:127.0.0.11:42891

Chain DOCKER_POSTROUTING (1 references)
 pkts bytes target     prot opt in     out     source               destination

    0     0 SNAT       tcp  --  *      *       127.0.0.11           0.0.0.0/0
        to::53
    0     0 SNAT       udp  --  *      *       127.0.0.11           0.0.0.0/0
        to::53
root@c73ead55b34a:/free5gc# tcpdump -i eth0 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
```

Analysis:

- iptables configuration: can see two MASQUERADE rules, but the pkts and bytes counts are still 0. This means that the traffic did not hit the NAT rule and the data packet was not forwarded correctly.
- tcpdump packet capture: The UPF container is listening to ICMP traffic on eth0, but the screenshot does not show ICMP packets yet (probably the traffic did not reach the UPF exit).

After confirming the UPF export network card name, manually set it and restart to try to connect:

```
gxf@gxf:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t 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 noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 08:00:27:bc:de:0f brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
       valid_lft 67646sec preferred_lft 67646sec
    inet6 fe80::a00:27ff:febc:de0f/64 scope link
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOW
N group default
```

```
gxf@gxf:~/gtp5g/free5gc-compose/config$ sudo ip link add gtp5g type gtp
RTNETLINK answers: Invalid argument
gxf@gxf:~/gtp5g/free5gc-compose/config$ ip -V
ip utility, iproute2-6.1.0, libbpf 1.3.0
gxf@gxf:~/gtp5g/free5gc-compose/config$
```

Analysis: Although the GTP5G module has been loaded, the current system does not recognize the gtp type. The problem may be that the gtp5g module is incompatible with the current kernel or toolchain.

Confirmed: GTP5G module loaded normally; UPF and GTP5G configured correctly. But still can't complete the transmission correctly.

Consider the following issues:

- **Deep compatibility issues between GTP5G module and kernel version:** Although the module is loaded successfully, the core functions (such as interface creation and packet processing) are not fully effective, which may be related to the kernel version or module implementation.
- **Virtualization environment limitation:** When running in a virtual machine, the virtualization layer may prevent the correct processing of GTP packets

(for example, IP in bridge mode is unstable or the virtual machine does not support a specific protocol).