Warszawa, 21Z

Create UERANSIM VM

UE/RAN-SIM is a simulator of User Equipment/Radio Access Network.

Pracownia Dyplomowa Inżynierska 1

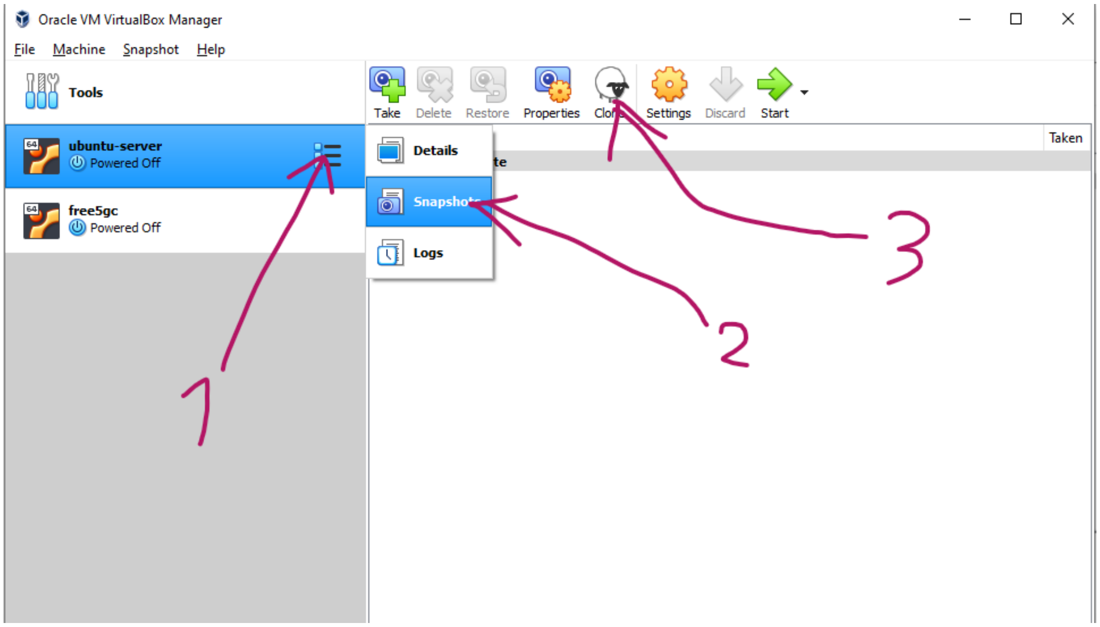
Andrzej Gawor

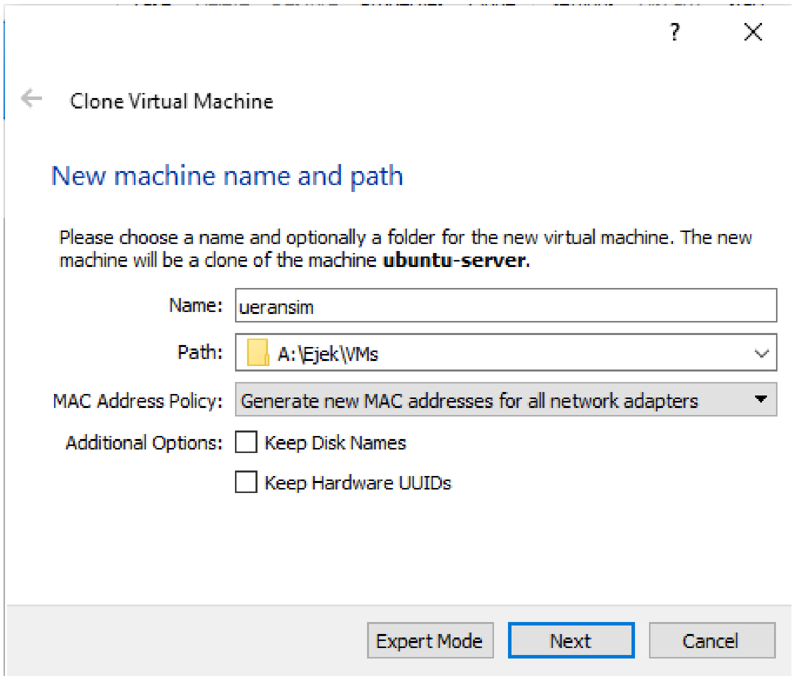
Contents

Step 1 - Setup VM

1.1 Clone ubuntu-server machine

Activities in this step are like the ones done in Create 5G Core VM.





Clone Virtual Machine

New machine name and path

Please choose a name and optionally a folder for the new virtual machine. The new machine will be a clone of the machine **ubuntu-server**.

Name: ueransim

Path: A:\Ejek\VMs

MAC Address Policy: Generate new MAC addresses for all network adapters

Additional Options: □ Keep Disk Names

□ Keep Hardware UUIDs

- In the next step both "full clone" or "linked clone" option will work.

1.2 Set static IP address

Run `ifconfig` to note IP address of Host-Only Network Interface

```
ifconfig
```

In my case it was again `192.168.56.102`, so first thing I did was to set a static IP address to `192.168.56.103`. We assigned `192.168.56.102` as a static IP address to free5gc VM. Two machines can't have the same address, so we need to change it.

Probably while free5gc VM is not running DHCP on my local network takes 102 as first available address.

```
cd /etc/netplan
cat 00-installer.yaml
```

Change the file to:

```
# This is the network config written by 'subiquity'
network:
ethernets:
  enp0s3:
    dhcp4: true
  enp0s8:
    dhcp4: no
    addresses: [192.168.56.103/24]
version: 2

sudo netplan try
```

Press Enter

```
sudo netplan apply
```

Run `ifconfig` to check if network setting has been changed correctly.

```
ifconfig
```

1.3 Setup SSH Connection

- Run your favorite SSH Client (Mine is MobaXterm).
- Open new terminal and type

```
ssh 192.168.56.103 -l ejek
```

1.4 Change hostname

Run

```
sudo nano /etc/hostname
```

Change `ubuntu-server` to `ueransim` (or the same name you gave the ueransim VM).

Click`ctrl+o` + `enter` + `ctrl+x` to exit nano.

Run to see if changes saved.

```
cat /etc/hostname
```

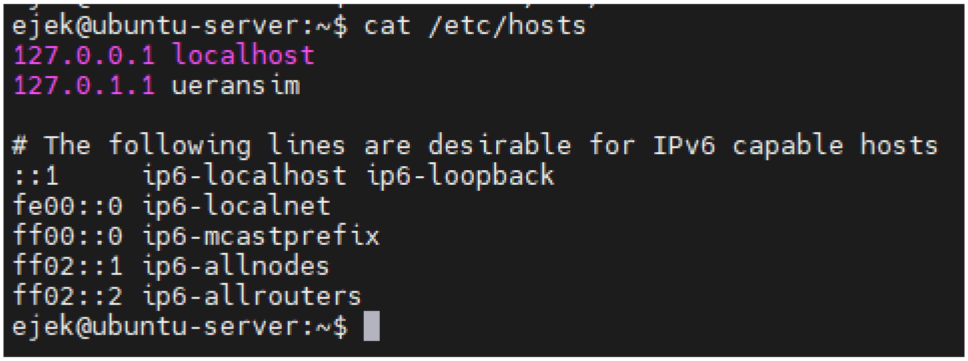
Now open `/etc/hosts`

```
sudo nano /etc/hosts
```

Check the file content

```
cat /etc/hosts
```

It should look like this



```
ejek@ubuntu-server:~$ cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 ueransim

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ejek@ubuntu-server:~$
```

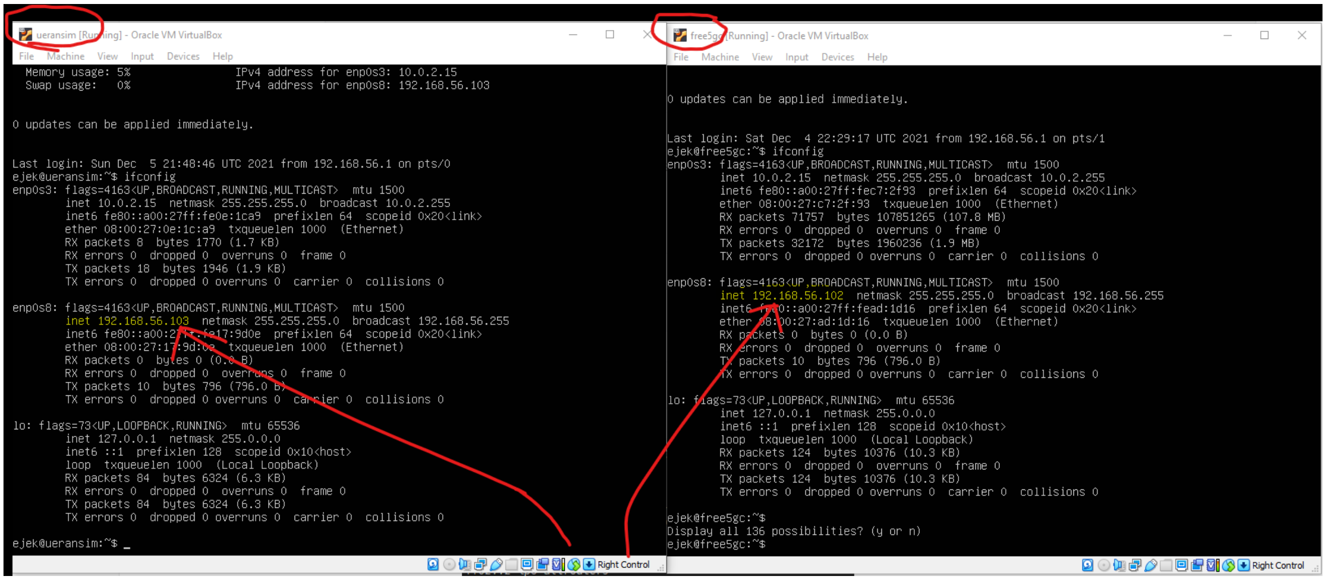
1.5 Reboot the machine

```
sudo shutdown -r now
```

In the meantime run the free5gc VM.

Log in to the machines and run `ifconfig` on both of them, to check IP addresses configuration.

In my case it is:


ueransim [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Control
free5gc [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Control

```
  Memory usage: 5%                IPv4 address for enp0s3: 10.0.2.15
  Swap usage:   0%                IPv4 address for enp0s8: 192.168.56.103


0 updates can be applied immediately.


Last login: Sun Dec  5 21:48:46 UTC 2021 from 192.168.56.1 on pts/0
ejek@ueransim:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe0e:1ca9  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:0e:1c:a9  txqueuelen 1000  (Ethernet)
        RX packets 8  bytes 1770 (1.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 18  bytes 1946 (1.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.103  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe17:9d0e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:17:9d:0e  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 796 (796.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 84  bytes 6324 (6.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 84  bytes 6324 (6.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ejek@ueransim:~$ _
```

```
0 updates can be applied immediately.


Last login: Sat Dec  4 22:29:17 UTC 2021 from 192.168.56.1 on pts/1
ejek@free5gc:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fec7:2f93  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:c7:2f:93  txqueuelen 1000  (Ethernet)
        RX packets 71757  bytes 107851265 (107.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32172  bytes 1960236 (1.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.102  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fead:1d16  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:ad:1d:16  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10  bytes 796 (796.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 124  bytes 10376 (10.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 124  bytes 10376 (10.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ejek@free5gc:~$
Display all 136 possibilities? (y or n)
ejek@free5gc:~$
```

1.6 Ping VMs

Log in to the machines via SSH (it may require opening a new terminal card).

Ping both machines from each other.

```
ping <address_of_second_VM>
```

If both machines can ping each other everything is ok. You can move on.

Step 2 - UERANSIM installation

2.1 Visit Github

Visit this page and follow instructions.

Below (from 2.2) is the documentation of installation state that I found. It may differ in your case.

2.2 Clone repository

Go to `$HOME` directory and clone repository.

```
cd $HOME
git clone https://github.com/aligungr/UERANSIM
```

2.3 Install dependencies

It is a good practice as a Linux User to update your apt repositories and upgrade packages before every installation.

```
sudo apt update
sudo apt upgrade
```

Run a few `apt install`'s

```
sudo apt install make
sudo apt install gcc
sudo apt install g++
sudo apt install libsctp-dev lksctp-tools
sudo apt install iproute2
sudo snap install cmake --classic
```

2.4 Build UERANSIM

```
cd ~/UERANSIM
make
```

Step 3 - Use WebConsole to Add an UE

3.1 Run WebConsole server

Switch to free5gc SSH session and run

```
cd ~/free5gc/webconsole
go run server.go
```

3.2 Open Web Browser

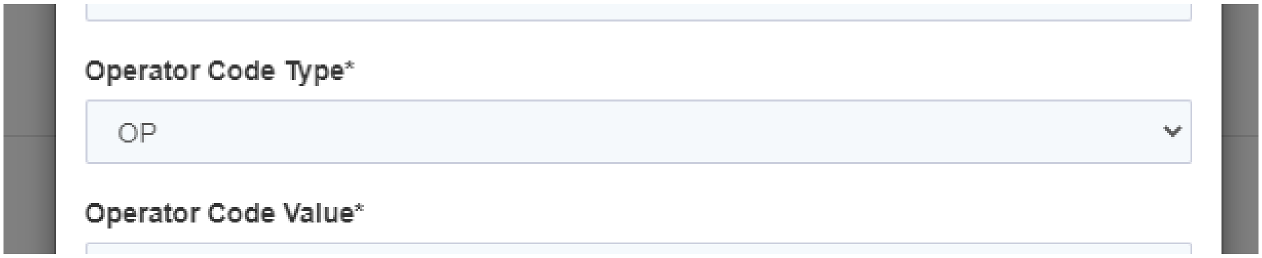
Open your web browser from your host machine, and enter the URL `http://<free5gc_VM_ip>:5000`

iny my case:

```
http://192.168.56.102:5000/#/
```

3.3 Add an UE

- On the login page, enter username `admin` and password `free5gc`
- Once logged in, widen the page until you see “Subscribers” on the left-hand side column.
- Choose "Subscribers"and create a new data:
  - Note that other than the “Operator Code Type” field which you should choose “OP” for now, leave other fields unchanged. This registration data is used for ease of testing and actual use later.
- After the data is created successfully, you can press Ctrl-C on the terminal to quit WebConsole



Operator Code Type*

OP

Operator Code Value*

- After you submitted you can stop the server using `ctrl+c` and `clear` command to clear the screen.

Step 4 Setting free5gc and UERANSIM params

In free5gc VM, we need to edit three files:

- `~/free5gc/config/amfcfg.yaml` - Config of ***Access and Mobility Management Function***
- `~/free5gc/config/smfcfg.yaml` - Config of ***Session Management Function***
- `~/free5gc/NFs/upf/build/config/upfcfg.yaml` - Config of ***User Plan Function***

4.1 AMF

First SSH into free5gc VM, and change `~/free5gc/config/amfcfg.yaml`

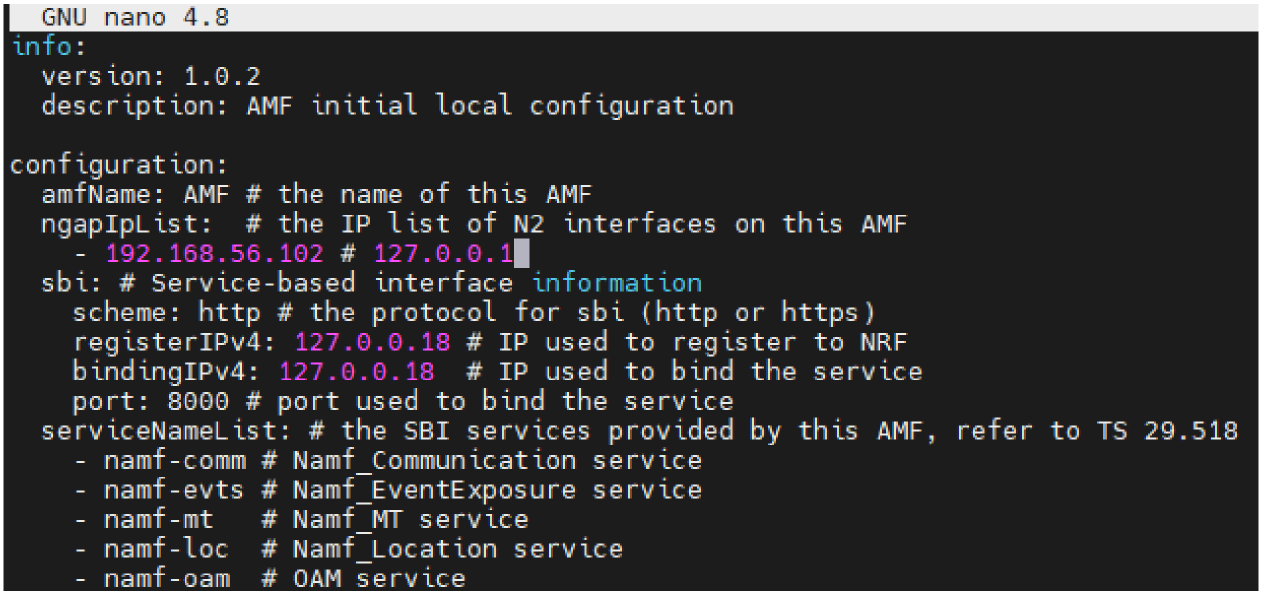
```
cd ~/free5gc
nano config/amfcfg.yaml
```

Replace ngapIpList IP from `127.0.0.1` to `<free5gc_VM_ip>` (`192.168.56.102` in my case), namely from:

```
...
ngapIpList:  # the IP list of N2 interfaces on this AMF
 - 127.0.0.1
...
```

into

```
...
ngapIpList:  # the IP list of N2 interfaces on this AMF
...
```



```
  GNU nano 4.8
info:
  version: 1.0.2
  description: AMF initial local configuration

configuration:
  amfName: AMF # the name of this AMF
  ngapIpList:  # the IP list of N2 interfaces on this AMF
    - 192.168.56.102 # 127.0.0.1
  sbi: # Service-based interface information
    scheme: http # the protocol for sbi (http or https)
    registerIPv4: 127.0.0.18 # IP used to register to NRF
    bindingIPv4: 127.0.0.18  # IP used to bind the service
    port: 8000 # port used to bind the service
  serviceNameList: # the SBI services provided by this AMF, refer to TS 29.518
    - namf-comm # Namf_Communication service
    - namf-evts # Namf_EventExposure service
    - namf-mt   # Namf_MT service
    - namf-loc  # Namf_Location service
    - namf-oam  # OAM service
```

4.2 SMF

Edit `~/free5gc/config/smfcfg.yaml`

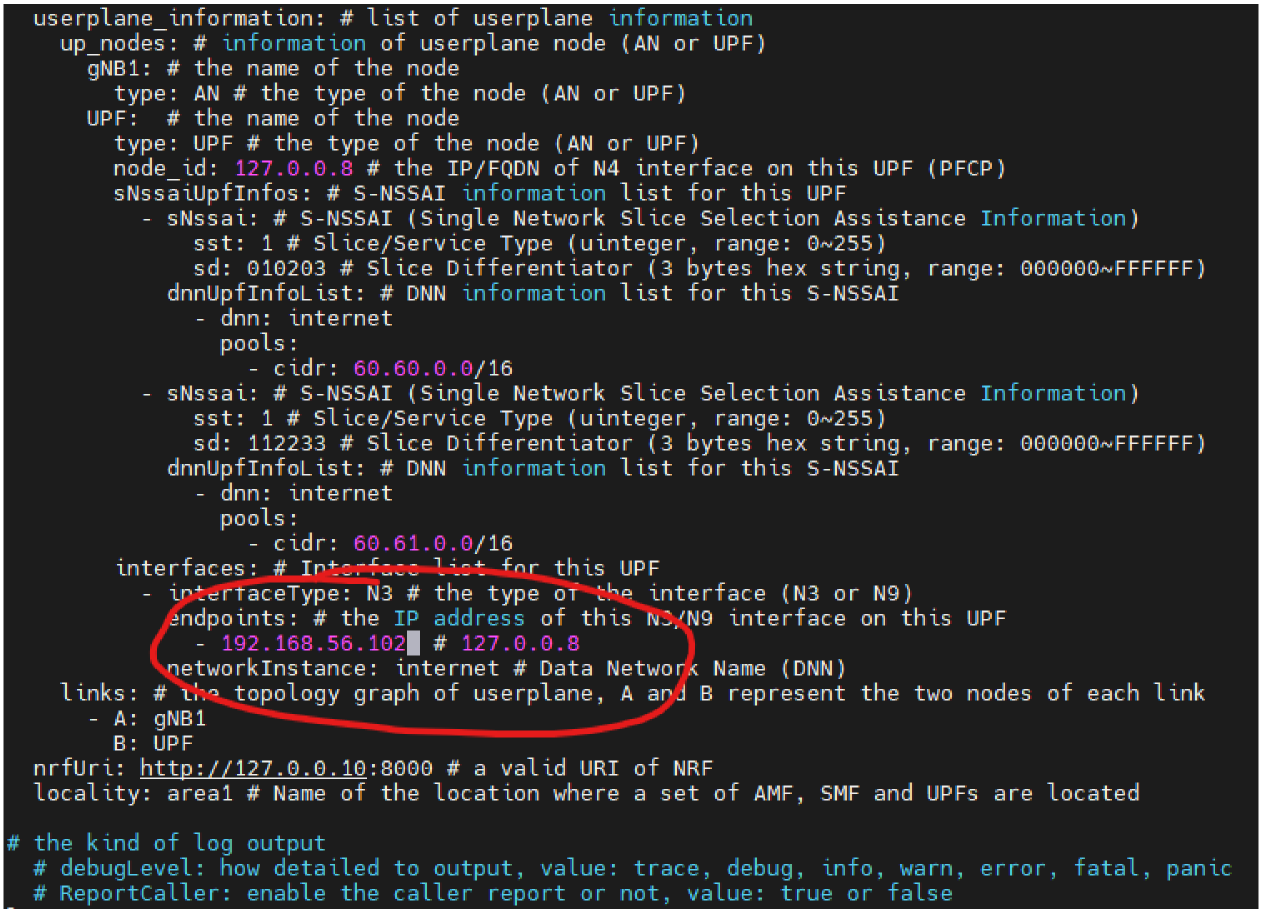
```
nano config/smfcfg.yaml
```

and in the entry inside userplane_information / up_nodes / UPF / interfaces / endpoints, change the IP from `127.0.0.8` to `192.168.56.102`, namely from

```
...

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
     - 127.0.0.8
```

into

```
...
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
     - 192.168.56.102  # 127.0.0.8
```



```
 userplane_information: # list of userplane information
  up_nodes: # 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)
    node_id: 127.0.0.8 # the IP/FQDN of N4 interface on this UPF (PFCP)
    sNssaiUpfInfos: # S-NSSAI information list for this UPF
     - 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)
      dnnUpfInfoList: # DNN information list for this S-NSSAI
       - dnn: internet
        pools:
         - cidr: 60.60.0.0/16
     - sNssai: # S-NSSAI (Single Network Slice Selection Assistance Information)
       sst: 1 # Slice/Service Type (uinteger, range: 0~255)
       sd: 112233 # Slice Differentiator (3 bytes hex string, range: 000000~FFFFFF)
      dnnUpfInfoList: # DNN information list for this S-NSSAI
       - dnn: internet
        pools:
         - cidr: 60.61.0.0/16
    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
       - 192.168.56.102 # 127.0.0.8
      networkInstance: internet # Data Network Name (DNN)
  links: # the topology graph of userplane, A and B represent the two nodes of each link
   - A: gNB1
    B: UPF
 nrfUri: http://127.0.0.10:8000 # a valid URI of NRF
 locality: area1 # Name of the location where a set of AMF, SMF and UPFs are located

# the kind of log output
 # debugLevel: how detailed to output, value: trace, debug, info, warn, error, fatal, panic
 # ReportCaller: enable the caller report or not, value: true or false
```

4.3 UPF

Finally, edit `~/free5gc/NFs/upf/build/config/upfcfg.yaml`

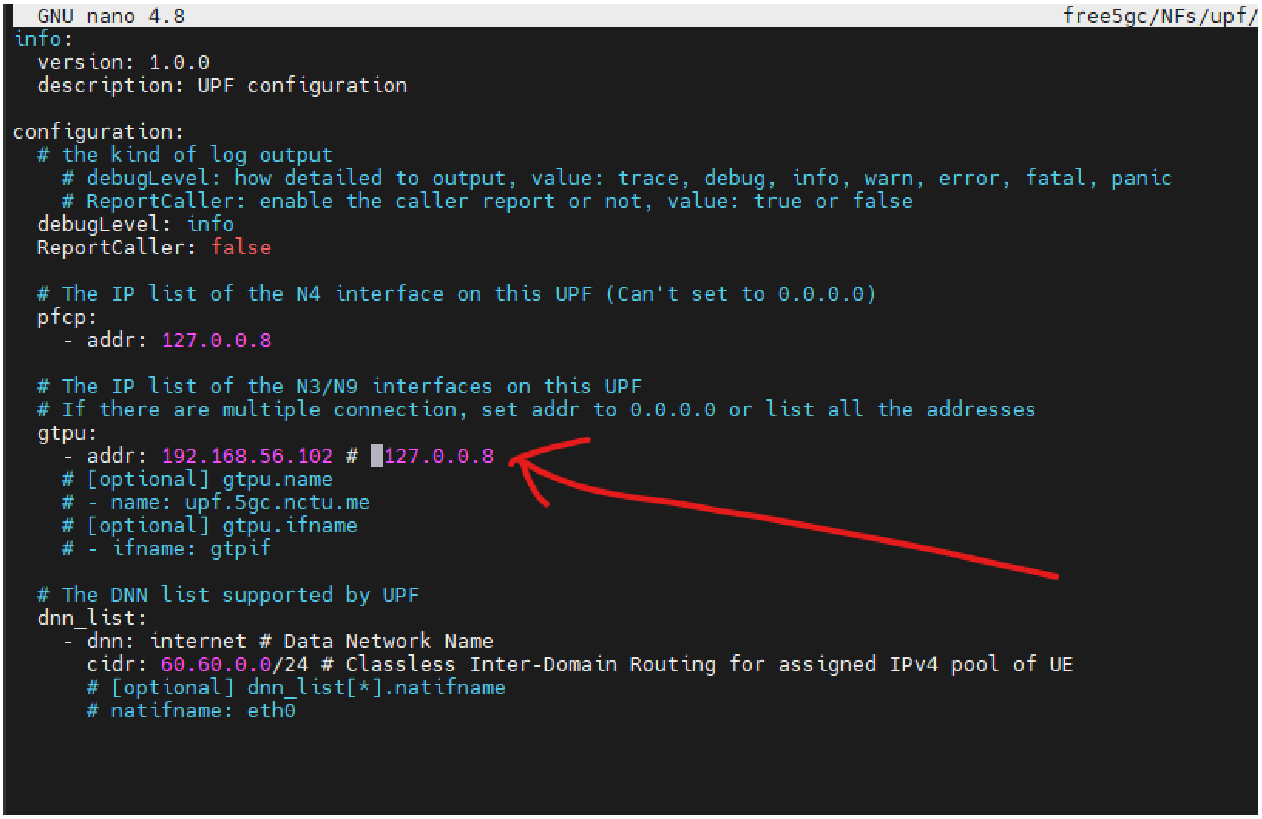
```
cd $HOME
nano free5gc/NFs/upf/build/config/upfcfg.yaml
```

and change gtpu IP from `127.0.0.8` into `192.168.56.102`, namely from

```
...
gtpu:
 - addr: 127.0.0.8
```

into

```
...
gtpu:
 - addr: 192.168.56.101  # 127.0.0.8
```



```
  GNU nano 4.8                                              free5gc/NFs/upf/
info:
  version: 1.0.0
  description: UPF configuration

configuration:
  # the kind of log output
    # debugLevel: how detailed to output, value: trace, debug, info, warn, error, fatal, panic
    # ReportCaller: enable the caller report or not, value: true or false
  debugLevel: info
  ReportCaller: false

  # The IP list of the N4 interface on this UPF (Can't set to 0.0.0.0)
  pfcp:
    - addr: 127.0.0.8

  # 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
  gtpu:
    - addr: 192.168.56.102 # 127.0.0.8
    # [optional] gtpu.name
    # - name: upf.5gc.nctu.me
    # [optional] gtpu.ifname
    # - ifname: gtpif

  # The DNN list supported by UPF
  dnn_list:
    - dnn: internet # Data Network Name
      cidr: 60.60.0.0/24 # Classless Inter-Domain Routing for assigned IPv4 pool of UE
      # [optional] dnn_list[*].natifname
      # natifname: eth0
```

Step 5 - Setting UERANSIM

In the ueransim VM, there are two files related to free5GC：

- `~/UERANSIM/config/free5gc-gnb.yaml`
- `~/UERANSIM/config/free5gc-ue.yaml`

The second file is for UE, which we don’t have to change if the data inside is consistent with the (default) registration data we set using WebConsole previously.

5.1 Network setup

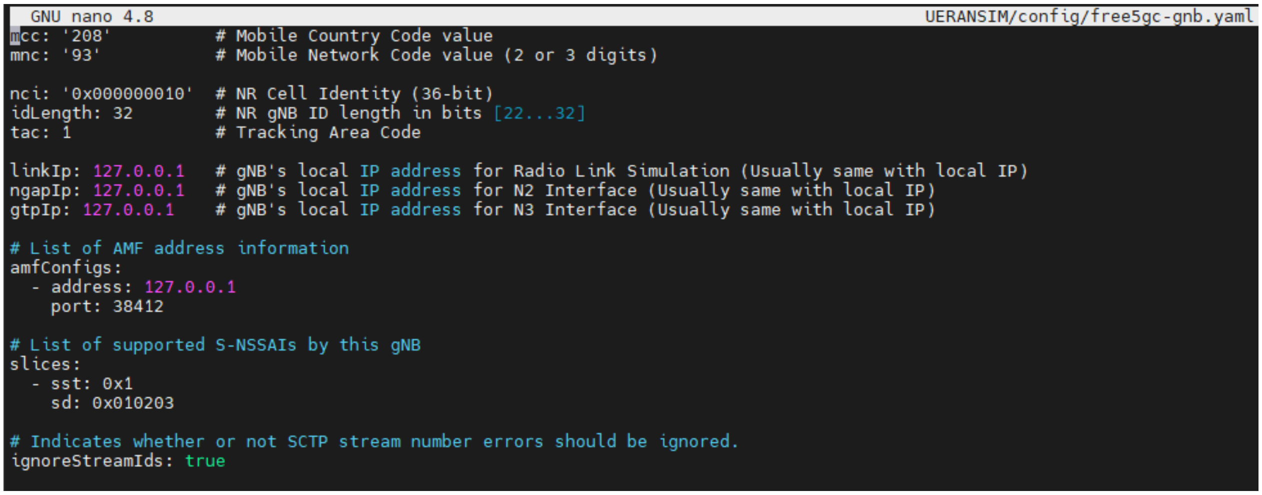
First SSH into ueransim, and edit the file `~/UERANSIM/config/free5gc-gnb.yaml`

```
cd $HOME
nano UERANSIM/config/free5gc-gnb.yaml
```

Change:

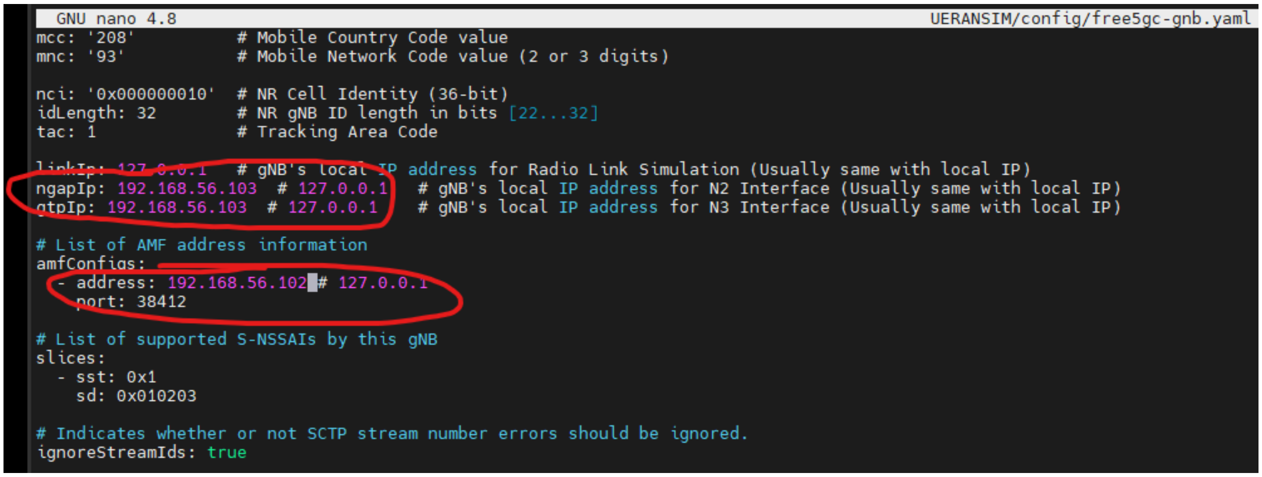
- ngapIp IP from 127.0.0.1 to 192.168.56.103,
- gtpIp IP, from 127.0.0.1 to 192.168.56.103,
- change the IP in amfConfig into 192.168.56.102

From:



```
  GNU nano 4.8                                                    UERANSIM/config/free5gc-gnb.yaml
mcc: '208'          # Mobile Country Code value
mnc: '93'           # Mobile Network Code value (2 or 3 digits)

nci: '0x000000010'  # NR Cell Identity (36-bit)
idLength: 32        # NR gNB ID length in bits [22...32]
tac: 1              # Tracking Area Code

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

# List of AMF address information
amfConfigs:
  - address: 127.0.0.1
    port: 38412

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

# Indicates whether or not SCTP stream number errors should be ignored.
ignoreStreamIds: true
```

To:



```
  GNU nano 4.8                                                    UERANSIM/config/free5gc-gnb.yaml
mcc: '208'          # Mobile Country Code value
mnc: '93'           # Mobile Network Code value (2 or 3 digits)

nci: '0x000000010'  # NR Cell Identity (36-bit)
idLength: 32        # NR gNB ID length in bits [22...32]
tac: 1              # Tracking Area Code

linkIp: 127.0.0.1   # gNB's local IP address for Radio Link Simulation (Usually same with local IP)
ngapIp: 192.168.56.103  # 127.0.0.1   # gNB's local IP address for N2 Interface (Usually same with local IP)
gtpIp: 192.168.56.103  # 127.0.0.1    # gNB's local IP address for N3 Interface (Usually same with local IP)

# List of AMF address information
amfConfigs:
  - address: 192.168.56.102 # 127.0.0.1
    port: 38412

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

# Indicates whether or not SCTP stream number errors should be ignored.
ignoreStreamIds: true
```

5.2 Examine UE file

Next we examine the file `~/UERANSIM/config/free5gc-ue.yaml`, and see if the settings is consistent with those in free5GC (via WebConsole), for example:

```
cd $HOME
nano UERANSIM/config/free5gc-ue.yaml

# IMSI number of the UE. IMSI = [MCC|MNC|MSISDN] (In total 15 or 16 digits)
supi: 'imsi-208930000000003'
# Mobile Country Code value
# Mobile Network Code value (2 or 3 digits)
mnc: '93'

# Permanent subscription key
key: '8baf473f2f8fd09487cccbd7097c6862'
# Operator code (OP or OPC) of the UE
op: '8e27b6af0e692e750f32667a3b14605d'
# This value specifies the OP type and it can be either 'OP' or 'OPC'
opType: 'OP'

...

# Initial PDU sessions to be established
sessions:
- type: 'IPv4'
  apn: 'internet'
  slice:
    sst: 0x01
    sd: 0x010203

# List of requested S-NSSAIs by this UE
slices:
- sst: 0x01
  sd: 0x010203

...
```

Step 6 Testing UERANSIM against free5gc

6.1 Run 5G Core

SSH into free5gc. If you have rebooted free5gc, remember to do:

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
sudo systemctl stop ufw
```

In addition, execute the following command:

```
sudo iptables -I FORWARD 1 -j ACCEPT
```

Also, make sure you have make proper changes to the free5GC configuration files, then run `./run.sh`

```
cd ~/free5gc
./run.sh
```

At this time free5GC has been started.

6.2 Prepare Configuration

Prepare three additional SSH terminals from your host machine

6.2.1 Terminal 1

In terminal 1: SSH into ueransim, make sure UERANSIM is built, and configuration files have been changed correctly, then execute `nr-gnb`

```
cd ~/UERANSIM
make #to make sure its build
build/nr-gnb -c config/free5gc-gnb.yaml
```

6.2.2 Terminal 2

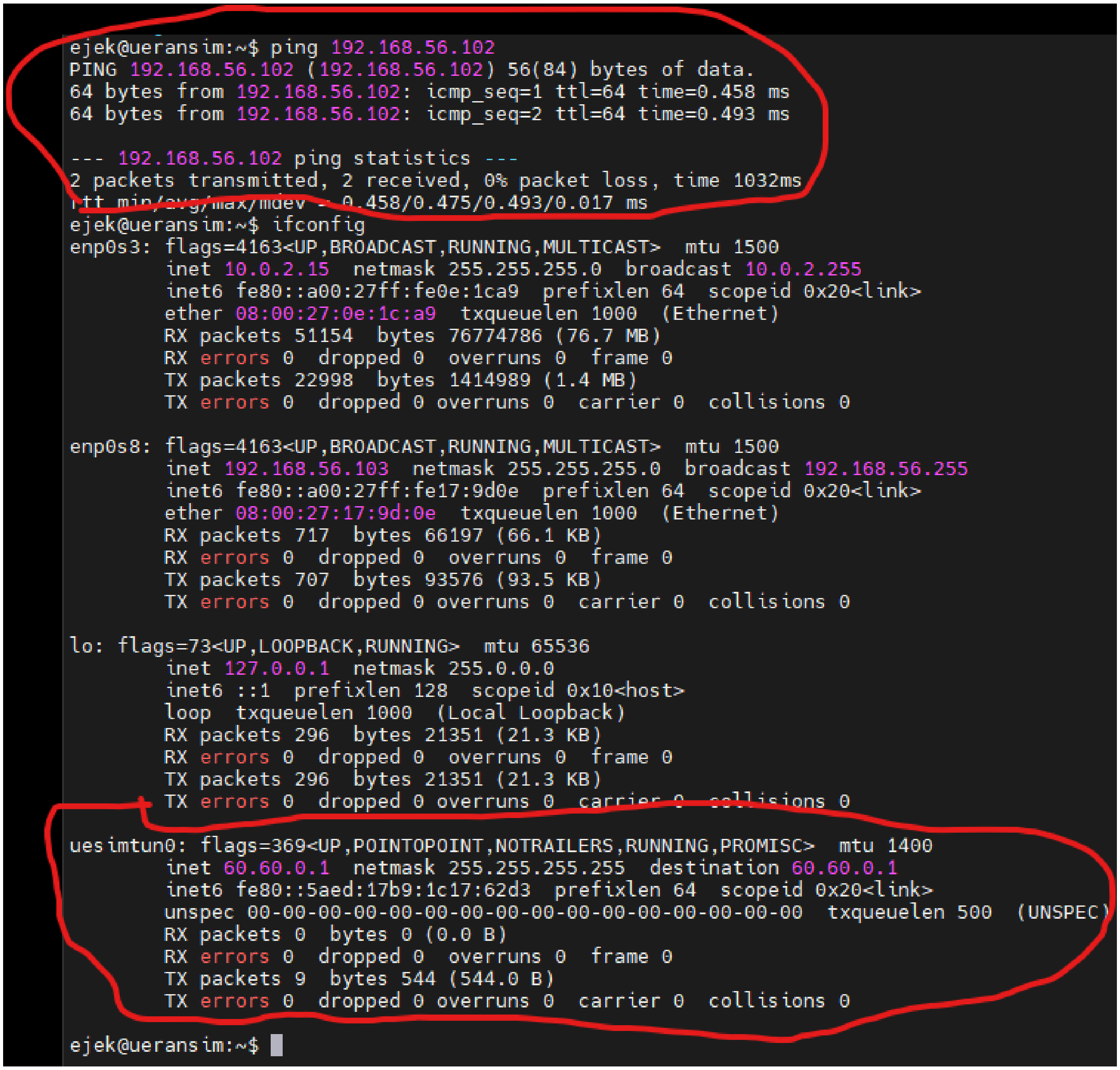
In terminal 2, SSH into ueransim, and execute `nr-gnb` with admin right:

```
cd ~/UERANSIM
sudo build/nr-ue -c config/free5gc-ue.yaml
```

6.2.3 Terminal 3

In terminal 3, SSH into ueransim, and `ping 192.168.56.102` to see free5gc is alive. Then, use `ifconfig` to see if the tunnel `uesimtun0` has been created (by nr-ue):

```
ping 192.168.56.102
ifconfig
```



```
ejek@ueransim:~$ ping 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data.
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=0.458 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=0.493 ms

--- 192.168.56.102 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1032ms
rtt min/avg/max/mdev = 0.458/0.475/0.493/0.017 ms
ejek@ueransim:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe0e:1ca9  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:0e:1c:a9  txqueuelen 1000  (Ethernet)
        RX packets 51154  bytes 76774786 (76.7 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 22998  bytes 1414989 (1.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.103  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe17:9d0e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:17:9d:0e  txqueuelen 1000  (Ethernet)
        RX packets 717  bytes 66197 (66.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 707  bytes 93576 (93.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 296  bytes 21351 (21.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 296  bytes 21351 (21.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

uesimtun0: flags=369<UP,POINTOPOINT,NOTRAILERS,RUNNING,PROMISC>  mtu 1400
        inet 60.60.0.1  netmask 255.255.255.255  destination 60.60.0.1
        inet6 fe80::5aed:17b9:1c17:62d3  prefixlen 64  scopeid 0x20<link>
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 9  bytes 544 (544.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ejek@ueransim:~$ 
```

6.3 Test with ping

Now use `ping`:

```
ping -I uesimtun0 google.com
```

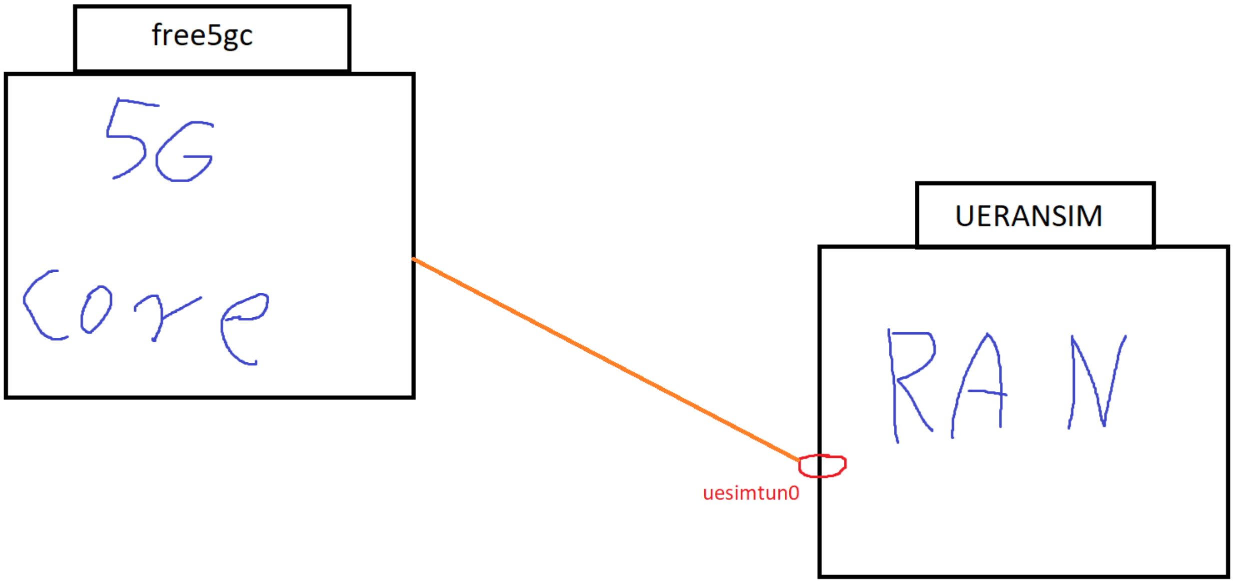
If `ping` gets replies, then free5GC is running properly. Congratulations!

ping -I option

-I *interface address*

Set source address to specified interface address. Argument may be numeric IP address or name of device. When pinging IPv6 link-local address this option is required.

What have we've built looks like this:


free5gc
5G
Core
UERANSIM
RAN
uesimtun0