In-depth project

"Implementing and Testing a 5G Core Network with Free5GC."

Submitted by: ThomasZielasny
Matriculation number: 1177217
Study program: Electrical Engineering and Information
Technology Specialization: Information and Communication
Technology University: Frankfurt University of Applied Sciences
Supervisor: Prof. Dr.- Ing. Ulrich Trick
Period: 04.05.2022 - 28.06.2023 (summer semester 23)

Affidavit

I hereby certify that I have written this thesis independently and have not used any sources other than those listed in the bibliography. Passages taken verbatim or in spirit from published or as yet unpublished sources are identified as such. Drawings or illustrations in this thesis have been made by myself or have been provided with an appropriate source reference. This work has not been submitted in the same or similar form to any other examination authority.

28.06.2023, Thomas Zielasny

Table of contents

1 Introduction

The report produced as part of the in-depth project deals with *Free5GC*. *Free5GC* is an open source implementation of a 5G core network based on the 3GPP standard. Where the 5G represents the fifth generation of wireless technology and enables a significant improvement in wireless communications with high bandwidth, low latency and support for a large number of simultaneous connections. The *5G* core is the heart of the 5G network and manages and coordinates various services and resources. Through *Free5GC,* researchers and developers are provided with a platform to implement and test new ideas, protocols and techniques related to the 5G core networks. It provides a complete implementation of the functions and protocols required in a 5G core network, which are customizable and extensible by the user. Among other things, new routing and signaling algorithms can be developed, network functions can be virtualized and scaled, and security solutions can be tested.

This report provides an introduction to the Free5GC framework, describes its architecture and features, and discusses potential application areas and research questions. It also simulates the implementation of the *Free5GC* in conjunction with the *Radio Access Network (RAN)* and *User- Equipment (UE).* By leveraging and enabling this platform, scientists and researchers can make a valuable contribution to the advancement of 5G technology.

2 Theoretical Background

In the theoretical background, the technologies used are presented. In addition, essential 5G system elements are presented and special protocols used in connection with the technology are examined in more detail.

2.1 Used technologies

The technologies used describe programming languages that were used as part of the project work, databases used to store the data, and other software solutions that were used.

2.1.1 VirtualBox and Ubuntu 20.04 (Server)

VirtualBox is a virtualization software that allows users to install multiple operating systems on a host computer and run them in parallel. The software, provided by *Oracle,* allows creating and configuring so-called virtual machines (VMs) on which operating systems such as *Windows, Linux, macOS* and others can be set up. The technology is based on the hardware virtualization method present in modern processors. This allows the computer to run different operating systems simultaneously, running in isolated virtual environments, virtual machines. It also supports network communication between the host machine and the virtual machines. (cf. Oracle n.d.)

The *Ubuntu 20.04 (Server)* operating system is a special version of *Ubuntu that is* optimized for use as a server. It is particularly suitable as a platform for running web servers, as it already includes the *Apache http server, Nginx* and other web server software that can host web applications. In addition, the operating system supports various database servers such as *MySQL*, *PostgreSQL* and *MongoDB.* (cf. Kurek, T. 2019)

2.1.2 Go

The compiled programming language *Go is* very powerful and consumes little memory, offers native support for multithreading and also brings an extensive standard library, so it is particularly suitable for use as an *API server*. In this context, it offers the additional advantage of high scalability in a production environment, allowing complex processes to be executed efficiently in parallel. (cf. Google n.d.) It forms the basis for the *Free5GC* software and handles all requests in this 5G core environment.

2.1.3 MongoDB

Designed to support scalable and flexible data solutions, the document-oriented NoSQL database, unlike relational databases, provides data organization in JSON-like documents, allowing for easy handling of structured and unstructured data is enabled. As a database base for *Free5GC,* features such as persistence of configuration and user data, state management, scalability, and real-time data processing are supported. (cf. MongoDB Inc. n.d.)

2.1.4 T-Shark

T-Shark is a terminal program and part of the *Wireshark package* and is used for network protocol analysis. (cf. Ross, J. 2020) This is intended to monitor, filter and analyze the network interfaces on the Ubuntu servers in order to uncover possible errors and gain a better understanding of the functionality of the core network.

2.1.5 Free5GC

Free5GC is an open source implementation of a 5G core network and provides core functions for managing and operating the 5G core network. This includes functions such as access control, session management, user and control plane, authentication and authorization. As an open-source implementation, it is particularly suitable for exploring features and protocols and testing new functionalities. *Free5GC* is written in *Go and* uses *MongoDB* as a database to manage data. *Free5GC* represents the core of a 5G network and must interoperate with other components such as base station, radio access networks, and terminals to represent a complete 5G network. (cf. Free5GC n.d.)

2.1.6 UERANSIM

The open-source software emulates a virtual *5G UE (user equipment)* and is often used in combination with *Free5GC* for testing and development purposes. It allows developers and researchers to create virtual 5G user equipment and connect it to a 5G network. This makes it possible to test different use cases, network parameters and protocols without relying on physical devices. The use of the software in conjunction with *Free5GC* offers many opportunities, especially in use cases where a high number of end devices is required, as high scalability is possible through virtualization. (cf. Aligungr 2021)

2.2 5G system

The 5G system presented here consists of several components that work together and are interconnected. These include the *user equipment (UE)*, the *base stations (gNodeB)* and the *core network*, which is the central control system of the 5G network. In addition, other components such as a cloud infrastructure and network functions such as *VNFs* can be integrated. (cf. Trick, U. n.d.)

2.2.1 Network functions

Network functions refer to virtualized functions implemented in the 5G network and typically run on virtualized servers or in cloud infrastructures.

These include network functions such as:

- Access Mobility Management Function (AMF)
- Session Management Function (SMF)
- User Plane Function (UPF)
- Authentication Server Function (AUSF)
- Unified Data Management (UDM)
- Policy Control Function (PCF)
- Network Slice Selection Function (NSSF)
- Application Function (AF)

The *AMF* is responsible for managing the access and mobility of the terminals in the 5G network, including authentication in the network, authorization of access to the services. In addition, the *AMF* manages the state and position of the terminals, including *handover* between different base stations to ensure continuous connectivity during movement. The *AMF* is also responsible for establishing, updating, and terminating connections between the terminals and the network. (cf. Dryanski, M. 2018)

The *SMF*, on the other hand, is responsible for managing the communication sessions in the 5G network. This includes managing the *QoS (Quality of Service) parameters* and determines which priority, bandwidth and other parameters should be assigned for various communication sessions. In addition, the *SMF is* also responsible for the allocation and management of IP addresses. (cf. Dryanski, M. 2018)

The *UPF* is responsible for processing traffic in the user *plane.* This includes packet processing and forwarding. In addition, the *UPF* can buffer data to ensure that it is processed in the correct order and with the correct priorities. In addition, the *UPF* implements the specified *QoS parameters to ensure* that traffic meets the agreed quality of service requirements. (cf. Dryanski, M. 2018)

The *AUSF* provides security functions such as encryption and integrity protection. In addition, keys for authentication are managed by the function. (cf. Dryanski, M. 2018)

The *UDM* stores and manages the user profiles, subscription information, authentication data and relevant user information. In addition, billing and payment information related to the 5G services can also be managed. The *UDM* ensures that user data is stored and managed consistently, correctly, and securely. (cf. Dryanski, M. 2018)

The *PCF* is responsible for controlling and managing policies in the 5G network. This includes controlling *QoS parameters*, access controls for network and services, and *networkslicing to* provide dedicated virtual networks for specific services or applications. (cf. Dryanski, M. 2018)

The *networkslices* are managed and assigned by the *NSSF, including the* required resources, properties and *QoS requirements*. (cf. Dryanski, M. 2018)

The AF represents specific applications or services in the 5G network. In addition, the network function interacts with other network functions to provide required network resources and services to the application. (cf. Dryanski, M. 2018)

2.2.2 Access network

The access network, also referred to as *RAN (Radio Access Network), is* an essential component of a mobile communications network and provides the connection between the terminals and the core network. It is responsible for the wireless transmission of data, voice and other communications services. It enables the establishment of a wireless connection between the terminal and the *base station (gNB)*. The base station is part of the access network and serves as an interface between the terminal and the core network. (cf. Sultan, A. 2022) To ensure efficient transmissionbetween the terminal and the base station, it encodes, modulates and demodulates signals. It also manages the capacities of the radio channel and optimizes resource allocation. This includes the allocation of frequency bands, management of transmission schedules and interference control. *Handover* between different base stations, for example when the terminal moves from one radio cell to another, is also controlled by the access network. Finally, the access network implements security mechanisms such as authentication, encryption, and integrity protection to ensure the confidentiality and integrity of the transmitted data. (cf. Sultan, A. 2022).

2.2.3 User equipment

User Equipment (UE) refers to the terminal equipment in a mobile network. It is the hardware used by users to establish wireless communication and access network services. The *UE* is responsible for accessing the cellular network by sending credentials to authenticate itself on the network and is assigned a network address in response in order to benefit from services. The *UE* can take various forms such as smartphones, tablets, laptops, IoT *devices*, and other wireless terminals. (cf. 3GPP 2021)

2.3 Protocols

Several protocols are used in the 5G network to enable communication and exchange of data and control information between network components and function.

2.3.1 Packet Forwarding Control Protocol (PFCP)

The *PFCP protocol is* based on the IP protocol and is transported via *UDP (User Datagram Protocol).* It uses a client-serverbased architecture where the *SMF* acts as the client and the *UPF* acts as the server.

The protocol is used to provide packet flow control and resource management between the *SMF (Session Management Function)* and the *UPF (User Plane Function).* Among other things, the protocol enables the establishment, modification, and termination of sessions between the *SMF* and the *UPF,* as well as the management of packet flow resources, including bandwidth and *QoS parameters*. In addition, the *PFCP protocol* can be used to detect errors or deviations in the packet flow. (cf. Devopedia 2023)

Figure 1 shows an example registration initiated by the *SMF* to the *UPF*. Here, an association is first initiated as in message (1), which is then confirmed by the *UPF* (2). The

Association allows control and exchange of packet flow and resource management information between the two functions. Next, a message (3) is sent from the *SMF* to the *UPF* to establish a new session, which is subsequently acknowledged
(4). Finally, a message is sent that enables the modification of the session (5). The modification is confirmed by the *UPF* accordingly (6). The message types described here represent only a subset of the total set of types.

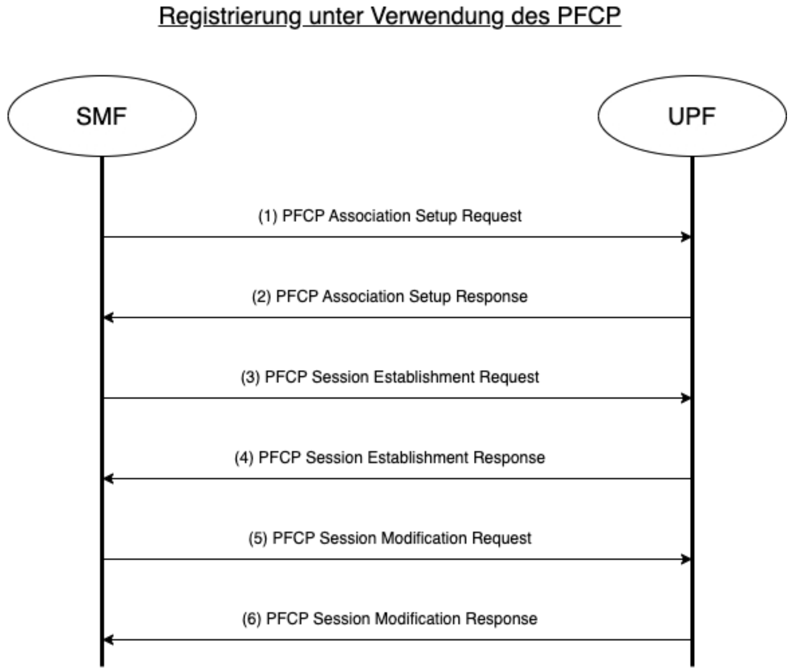


Figure 1: SMF registration process

2.3.2 GPRS Tunneling Protocol (GTP)

This protocol is used to enable tunneling of user data packets between the cellular core network and the cellular access network in cellular networks and in this case the 5G network. It enables the transfer of IP data packets between different network elements and is used for both the *uplink* (from the access network to the core network) and *downlink* (from the core network to the access network) *directions.* (cf. Devopedia 2023)

3 Requirements analysis

The requirements analysis will highlight the general objective, the specific requirements, the intended goal, and the potential uses of the application.

3.1 General Objective

In the context of this work, on the one hand the requirements necessary for the implementation of the *Free5GC* software and on the other hand the open source software itself are described in more detail. In addition, the *RAN* and the registration of a *UE* will be simulated using *UERANSIM.* This is to test and simulate the basic functions of the core network.

The *Free5GC* implementation is intended to enable users to test the functionality of the 5G core network and later use it for development. Therefore, the first step is to create an environment in which the *Free5GC* software can be implemented. Then, the core functions of the 5G core network will be tested. This includes *registration, GUTI- registration, service request, Xn-handover, deregistration, PDU session release request, paging, N2handover, Non3GPP, ReSynchronization and ULCL*. Using the open-source simulator, a *5G user equipment (UE)* is then emulated. The combination between *Free5GC* and *UERANSIM* is expected to provide a comprehensive test and development platform for 5G networks, where developers can study the interactions and behavior of *5G user equipment* and the 5G core network in a simulated environment.

3.2 Clarification of requirements

The development environment is to be set up using virtual machines in *VirtualBox.* For this purpose, *Ubuntu servers* are to be created on which the *Free5GC* software is installed and configured. To verify the successful installation, the server will be started and possible errors will be fixed. Afterwards, the integrated tests of the software will be executed to get an overview of the functionality of the open source software and to ensure the successful installation of the software.

Subsequently, the UERANSIM software is to be installed on a second virtual machine via a clone of the original server and configured so that it can be used in conjunction with *Free5GC.* For this purpose, the configuration files of both sides are adapted accordingly and communication is to be enabled. The next step is to start the *RAN* and register a *UE* on the *Free5GC* web interface.

3.3 Aimed Target

The installation of *Free5GC* in combination with the *UERANSIM* software will provide a platform for developers and researchers to simulate and test the functional scope of the 5G core network and the *radio access network*. Through the simulation of the virtual

5G network, it will be possible to develop and test new functions, which will allow an evaluation to be made of the performance of the network by implementing the new functions. In addition, the development environment will be used to test different configurations to achieve the optimal performance of the network.

3.4 Use cases

Setting up and simulating the *Free5GC* results in a variety of use cases.

Developers and researchers can use *Free5GC* and *UERANSIM to* design, test and optimize new network functions and protocols. They can simulate different scenarios to evaluate 5G network performance and make improvements. New link optimization algorithms can be developed and the network can be analyzed under high load.

By using this development environment, companies can also test their 5G applications in a simulated environment before deploying them in a production environment. This allows them to ensure that their applications interact smoothly with the 5G network and perform.

The combination also provides an ideal environment for training and educational purposes in the field of 5G. It provides the opportunities for students, researchers and professionals to gain hands-on experience by simulating different scenarios.

Integration testing and compatibility testing also provide an opportunity for *Free5GC* and *UERANSIM* to ensure that new 5G components or devices work seamlessly with the 5G network and meet the required standards prior to launch.

The use cases are just a few examples of the potential of using *Free5GC* in combination with *UERANSIM*. This combination provides a flexible and cost-effective solution for the development, testing and exploration of 5G networks.

4 Realization

In this section, the implementation of the technologies presented in the previous chapters is carried out. This requires a computer on which *VirtualBox* must be installed. The official instructions are followed for the installation of *VirtualBox. In* addition, the corresponding ISO file with the *Ubuntu 20.04 (Server)* distribution is downloaded from the official website. (cf. Free5GC o.D.)

After each subchapter, smaller tests are performed to ensure the successful installation and implementation of the respective technology. For this purpose, the *Wireshark* program is additionally installed on the computer so that the respective recordings can be viewed and analyzed.

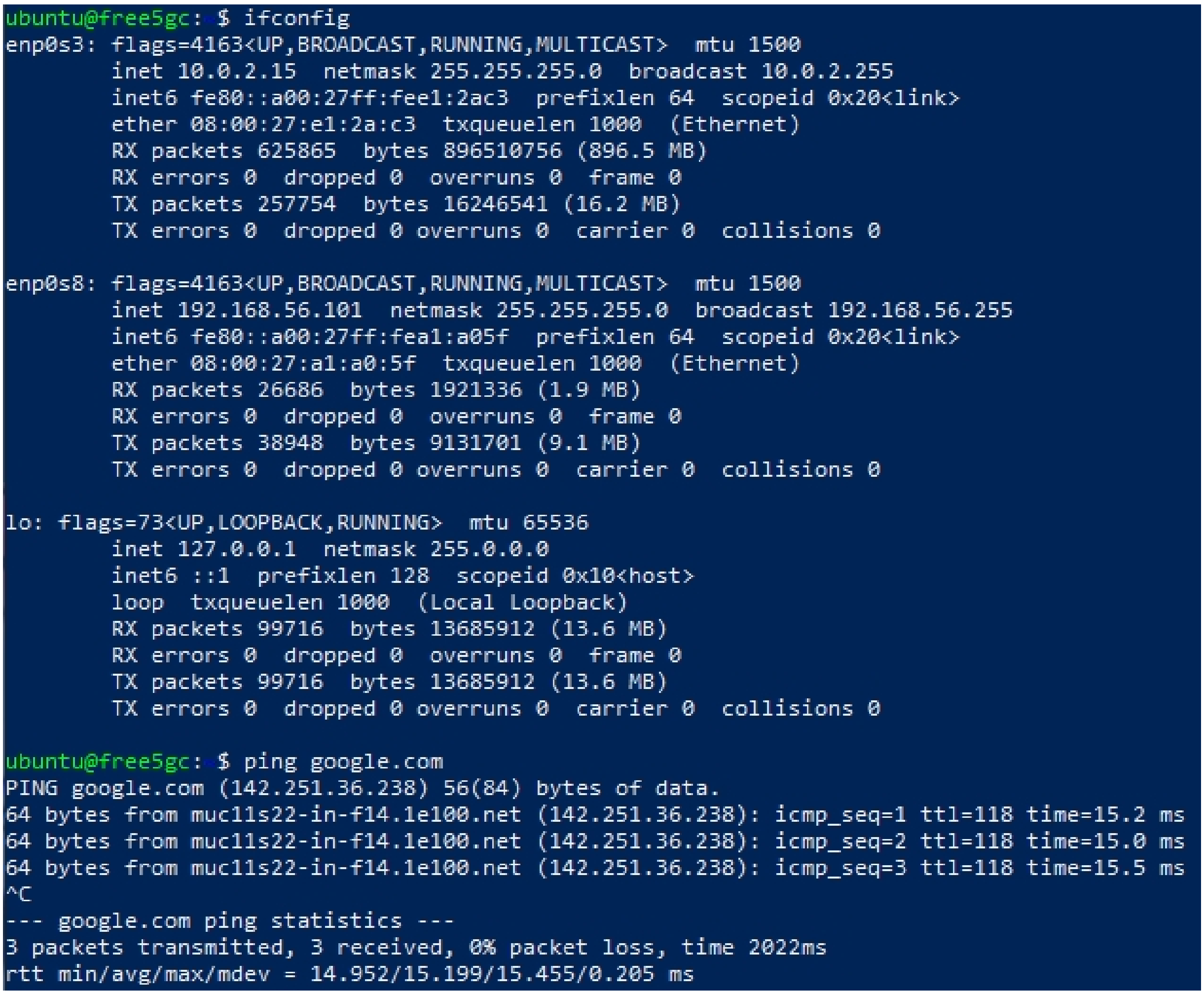
4.1 Setting up the virtual machines in VirtualBox

To prepare for the implementation of *Free5GC*, a virtual machine is configured and started according to the *Free5GC* instructions. It should be noted that a second network interface is added, which is configured as *'host-only'* alongside the existing *'NAT'*. Also, when the operating system is installed, the SSH server is installed as well. This will later enable connection via *SSH*. (cf. Free5GC o.D.)

Network interface (type)	IP address
enp0s3 (NAT)	10.0.2.15
enp0s8 (host-only)	192.168.56.101

To test the network connections, an *SSH connection is* established via a Powershell terminal using the *'host-only'* interface and, after successful login, a *ping is* sent to
'google.com' to check the connection of the *'NAT'* interface to the Internet. (see figure 2)

```
ssh 192.168.56.101 -l ubuntu
ping google.com
```



```
ubuntu@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:fee1:2ac3  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:e1:2a:c3  txqueuelen 1000  (Ethernet)
        RX packets 625865  bytes 896510756 (896.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 257754  bytes 16246541 (16.2 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.101  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fea1:a05f  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:a1:a0:5f  txqueuelen 1000  (Ethernet)
        RX packets 26686  bytes 1921336 (1.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 38948  bytes 9131701 (9.1 MB)
        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 99716  bytes 13685912 (13.6 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 99716  bytes 13685912 (13.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ubuntu@free5gc:~$ ping google.com
PING google.com (142.251.36.238) 56(84) bytes of data.
64 bytes from muc11s22-in-f14.1e100.net (142.251.36.238): icmp_seq=1 ttl=118 time=15.2 ms
64 bytes from muc11s22-in-f14.1e100.net (142.251.36.238): icmp_seq=2 ttl=118 time=15.0 ms
64 bytes from muc11s22-in-f14.1e100.net (142.251.36.238): icmp_seq=3 ttl=118 time=15.5 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2022ms
rtt min/avg/max/mdev = 14.952/15.199/15.455/0.205 ms
```

Figure 2: Free5GC network configuration

After that, the *t-shark* program is also installed to record traffic over the network interfaces at terminal level. This can then later be copied to the host computer using *SCP* and analyzed using *Wireshark.* (cf. Ross, J. 2019)

The current state of the virtual machine is now saved as a snapshot and now serves as a starting point for the clones for *Free5GC* and *UERANSIM.*

4.2 Installation of Free5GC

Before *Free5GC* is installed, a clone is created and started based on the created VM. (see Figure 3)

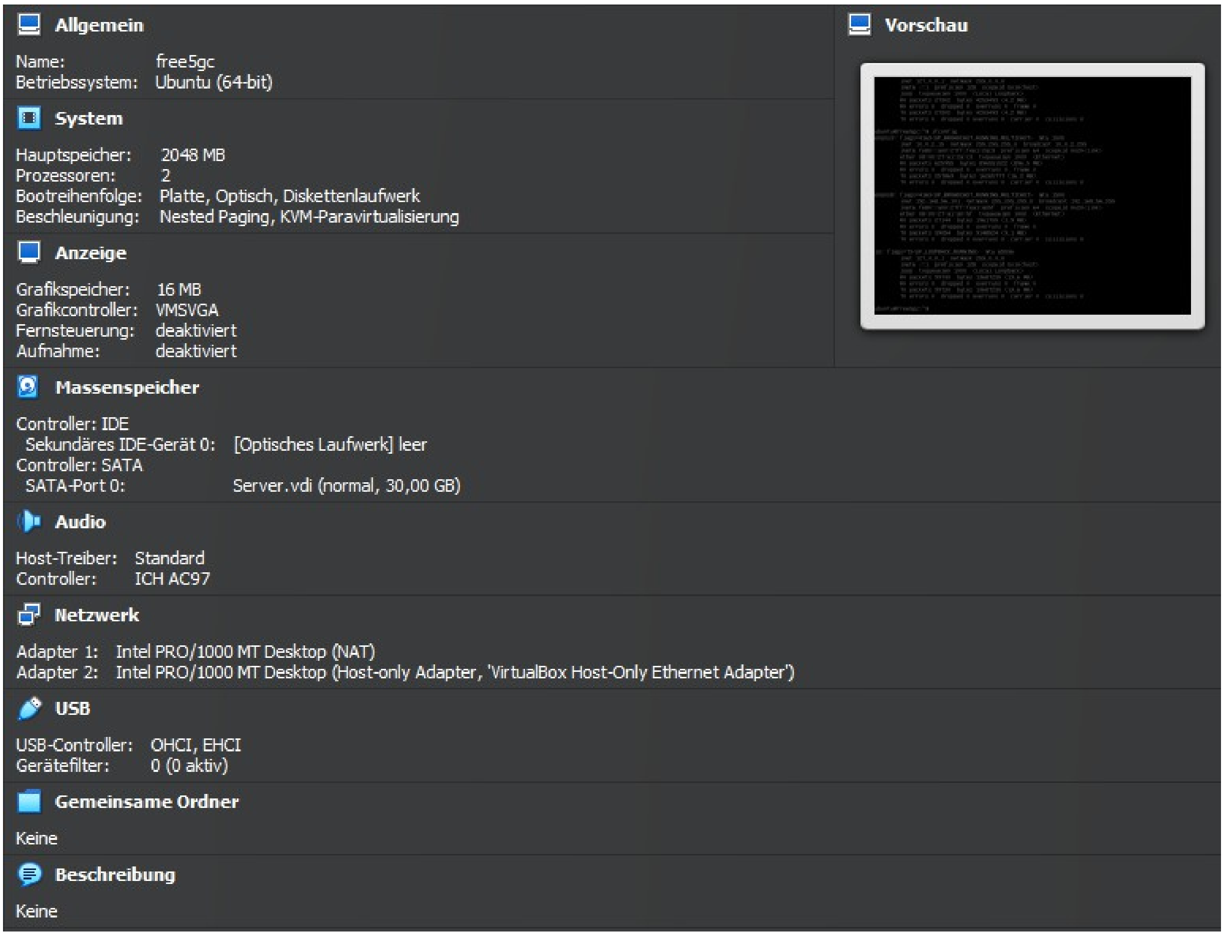


Figure 3: Configuration VM for Free5GC

For the installation, the steps of the official documentation of *Free5GC are* followed. All dependencies like *Go, MongoDB, Git* and *wget are* installed first. *MongoDB* is then started as a service. In the next step, all control-plane elements are installed and all packages required by *Go are* deployed.

```
go mod init
go mod tidy
```

After that, the control plane elements can be started and the user plane function can be installed. It should be noted that the correct kernel version (v5.0.0-23-generic or v5.4.x) is available. Finally, the WebConsole (see Figure 4) is installed and started with the following command. (cf. Free5GC o.D.)

```
go run server.go
```

This can then be reached from the host computer via the following address:

http://192.168.56.101:5000

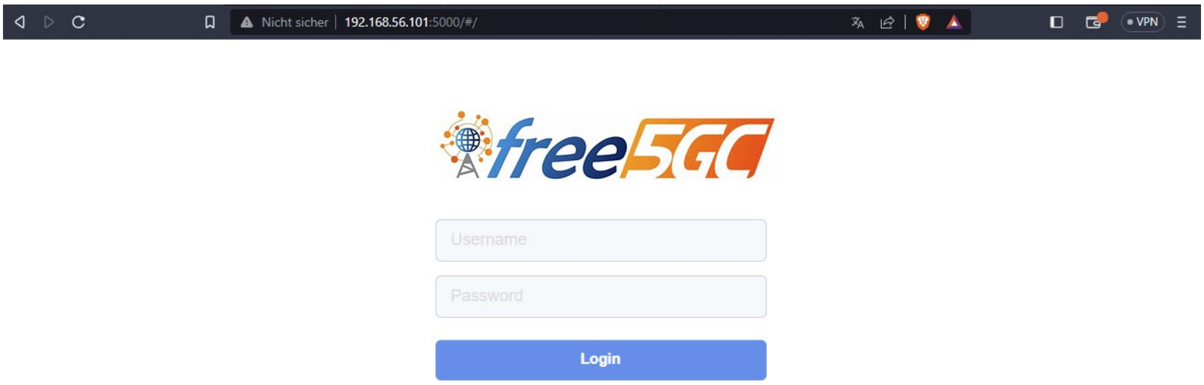


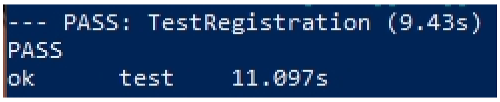
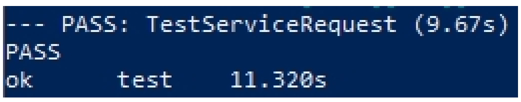
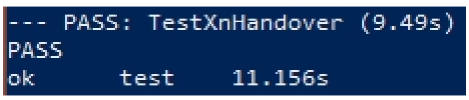
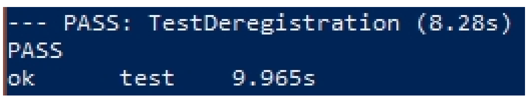
Figure 4: WebConsole

Now you can start with the tests that *Free5GC* provides in the path structure to check all functions.

4.3 Testing the core network functions

To test the functions, the tests from the test script given in the *Free5GC* directory are used, as described in the official documentation of *Free5GC*. First, however, the *UPF is* installed. (cf. Free5GC o.D.) The corresponding Wireshark cuts are in the appendix.

make upf

Test	Result
Registration (see Appendix 1)	 *Figure 5: Test Registration*
GUTI registration (see Appendix 2)	 *Figure 6: Test GUTI registration*
Service request (see Appendix 3)	 *Figure 7: Test service request*
Xn handover (see appendix 4)	 *Figure 8: Test Xn handover*
Deregistration (see Appendix 5)	 *Figure 9: Test Deregistration*

PDU Session Release Request (see Appendix 6)

```
--- PASS: TestPDUSessionReleaseRequest (10.44s)
PASS
ok      test    12.092s
```

Figure 10: Test PDU session release request

Paging (see Appendix 7)

```
--- PASS: TestPaging (11.71s)
PASS
ok      test    13.398s
```

Figure 11: Test paging

N2 handover (see appendix 8)

```
--- PASS: TestPaging (11.71s)
PASS
ok      test    13.398s
```

Figure 12: Test N2 handover

Non-3GPP (see appendix 9)

```
--- PASS: TestNon3GPPUE (9.68s)
PASS
ok      test    9.713s
```

Figure 13: Non-3GPP test

Resynchronization (see Appendix 10)

```
--- PASS: TestReSynchronization (9.55s)
PASS
ok      test    11.230s
```

Figure 14: Test Resynchronization

Request-Two PDU Sessions (see Appendix 11)

```
--- PASS: TestRequestTwoPDUSessions (11.63s)
PASS
ok      test    13.337s
```

Figure 15: Test Request-Two PDU Sessions

In order to gain a deeper insight into the processes of the tests, they are each recorded with *tshark* on the *veth0 network interface.* Afterwards, the *PCAP files are* to be copied to the host computer via *SCP* and opened for analysis with *Wireshark.*

4.4 Preparation of the virtual machine for UERANSIM

Before *UERANSIM* can be installed, a new virtual machine is cloned based on the initial machine and configured according to *Free5GC*'s documentation. (cf. Free5GC o.D.)

This is done by renaming the VM and the *hostname to* '*ueransim*', generating new *MAC addresses* for all interfaces and assigning a static IP address. (see Figure 16)

Network interface (type)	IP address
enp0s8 (host-only)	192.168.56.102

To test the network configuration, a *ping is* sent to the *Free5GC machine.* (see Figure 16)

```
ping 192.168.56.101
```

```
ubuntu@ueransim:~$ ifconfig
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:fe58:b1eb  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:58:b1:eb  txqueuelen 1000  (Ethernet)
        RX packets 5532  bytes 488289 (488.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4254  bytes 510213 (510.2 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 36333  bytes 1991612 (1.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 36333  bytes 1991612 (1.9 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ubuntu@ueransim:~$ ping 192.168.56.101
PING 192.168.56.101 (192.168.56.101) 56(84) bytes of data.
64 bytes from 192.168.56.101: icmp_seq=1 ttl=64 time=0.689 ms
64 bytes from 192.168.56.101: icmp_seq=2 ttl=64 time=0.324 ms
64 bytes from 192.168.56.101: icmp_seq=3 ttl=64 time=0.926 ms
^C
--- 192.168.56.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2440ms
rtt min/avg/max/mdev = 0.324/0.646/0.926/0.247 ms
```

Figure 16: Network configuration UERANSIM-VM

4.5 Installation of UERANSIM

To install *UERANSIM, the* official documentation of the developer is followed. Subsequently, the recommended check is performed. (see Figure 17) Since the expected files are present as described in the instructions, the installation was performed successfully. (cf. aligungr n.d.).

```
ubuntu@ueransim:~/UERANSIM/build$ ls -al
total 10876
drwxrwxr-x 2 ubuntu ubuntu    4096 Jun 20 10:24 .
drwxrwxr-x 9 ubuntu ubuntu    4096 Jun 20 10:18 ..
-rwxrwxr-x 1 ubuntu ubuntu   15560 Jun 20 10:24 libdevbnd.so
-rw-rw-r-- 1 ubuntu ubuntu     377 Jun 20 10:24 nr-binder
-rwxrwxr-x 1 ubuntu ubuntu  350568 Jun 20 10:24 nr-cli
-rwxrwxr-x 1 ubuntu ubuntu 6185128 Jun 20 10:24 nr-gnb
-rwxrwxr-x 1 ubuntu ubuntu 4563680 Jun 20 10:24 nr-ue
```

Figure 17: Directory structure /UERANSIM/build

4.6 Registration of the user equipment

Before the user equipment can be registered, some configurations must first be made in the *Free5GC VM.* For this purpose, a total of three files are opened and adapted according to the documentation. (cf. Free5GC o.D.)

First, the *ngapIpList parameter* in the AMF configuration is set to the interface that can then be used to reach the *gNB*. In this case, the IP address is changed from *127.0.0.1* to *192.168.56.101* (IP address of the *enp0s8 interface* on the *Free5GC VM*). (cf. Free5GC o.D.)

To configure the *SMF,* the corresponding configuration file is opened and the IP address of the *N3 interface is* changed from *127.0.0.1* to *192.168.56.101.* This ensures communication between the *SMF* and the *UPF* via the corresponding network segment. (cf. Free5GC o.D.)

In the last step, the *UPF* is configured. For this, the *N3 interface is* set in the *UPF* configuration file from *127.0.0.1* to the IP address of the enp0s8 interface (*192.168.56.101*). This *indicates* that the *UPF* is reachable through this interface and accepts connections from the *SMF.* (cf. Free5GC o.D.)

Now the configuration of the Free5GC-VM is completed and thus prepared for the connection with the *gNB* and the *UE.* Now the configuration of the UERANSIM-VM is continued. For this, the *free5gc-gnb.yaml file* is opened to configure the corresponding interfaces. The *N2-* interface is set to the IP address of the *enp0s8 network interface* (*192.168.56.102*) of the *UERANSIM-VM to* ensure communication with the *AMF.* The corresponding *N3 interface, which* is responsible for communication with the *UPF, is* also set to the IP address (*192.168.56.102*). Finally, the IP address of the *AMF* is specified so that the *gNB* knows via which address the *AMF* can be reached. (cf. Free5GC o.D.)

The preparation is completed as soon as the corresponding parameters in the *free5gc-ue.yaml file* match the parameters (see Figure 19) that were set in the WebConsole when the *UE* was newly created (see Figure 18). For this purpose the file is opened and the parameters are compared. It is also crucial that the *OP-Type* matches, as this is used to enable authentication and authorization of users as well as mapping to specific mobile networks. (cf. Free5GC n.d.)

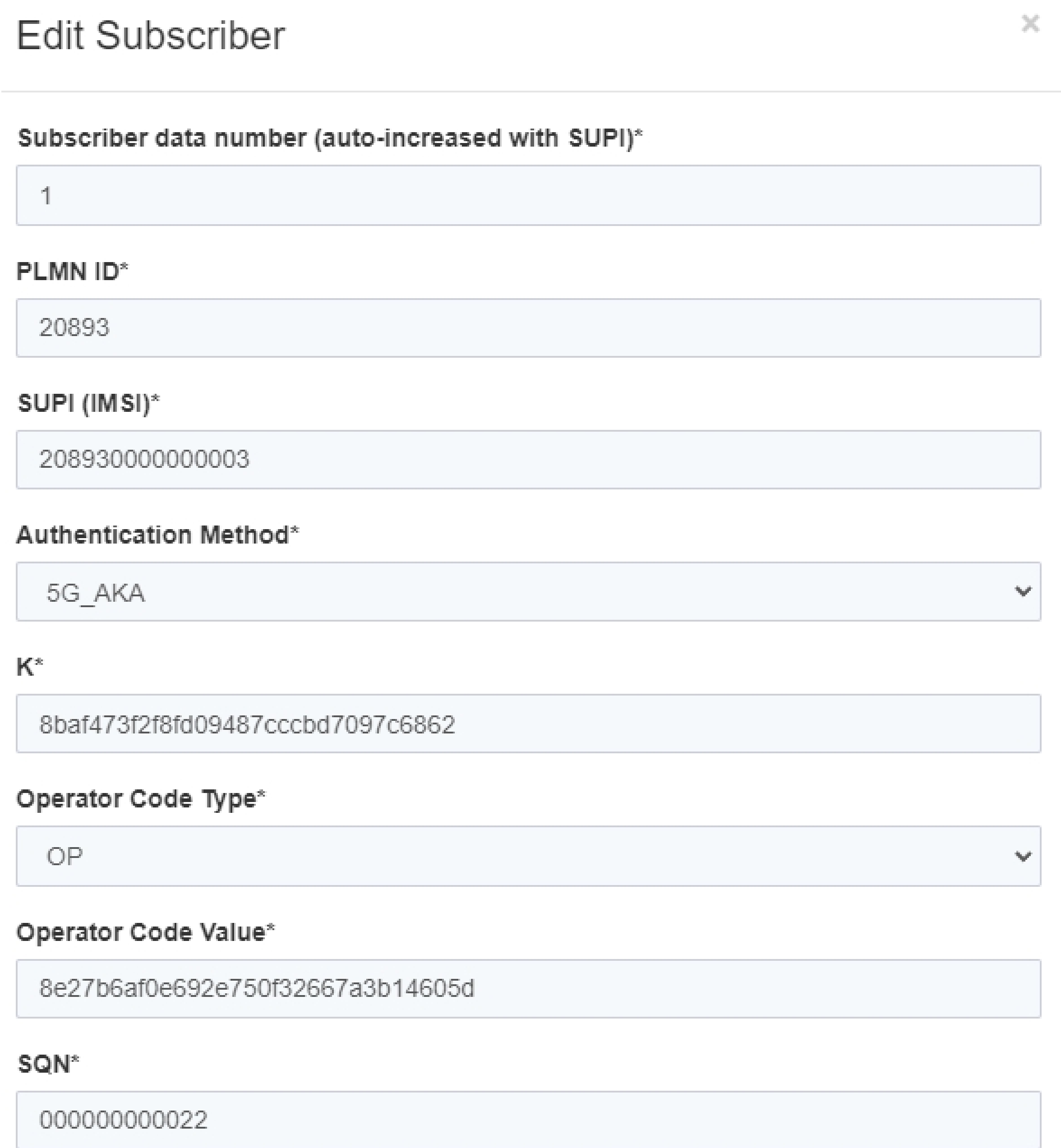
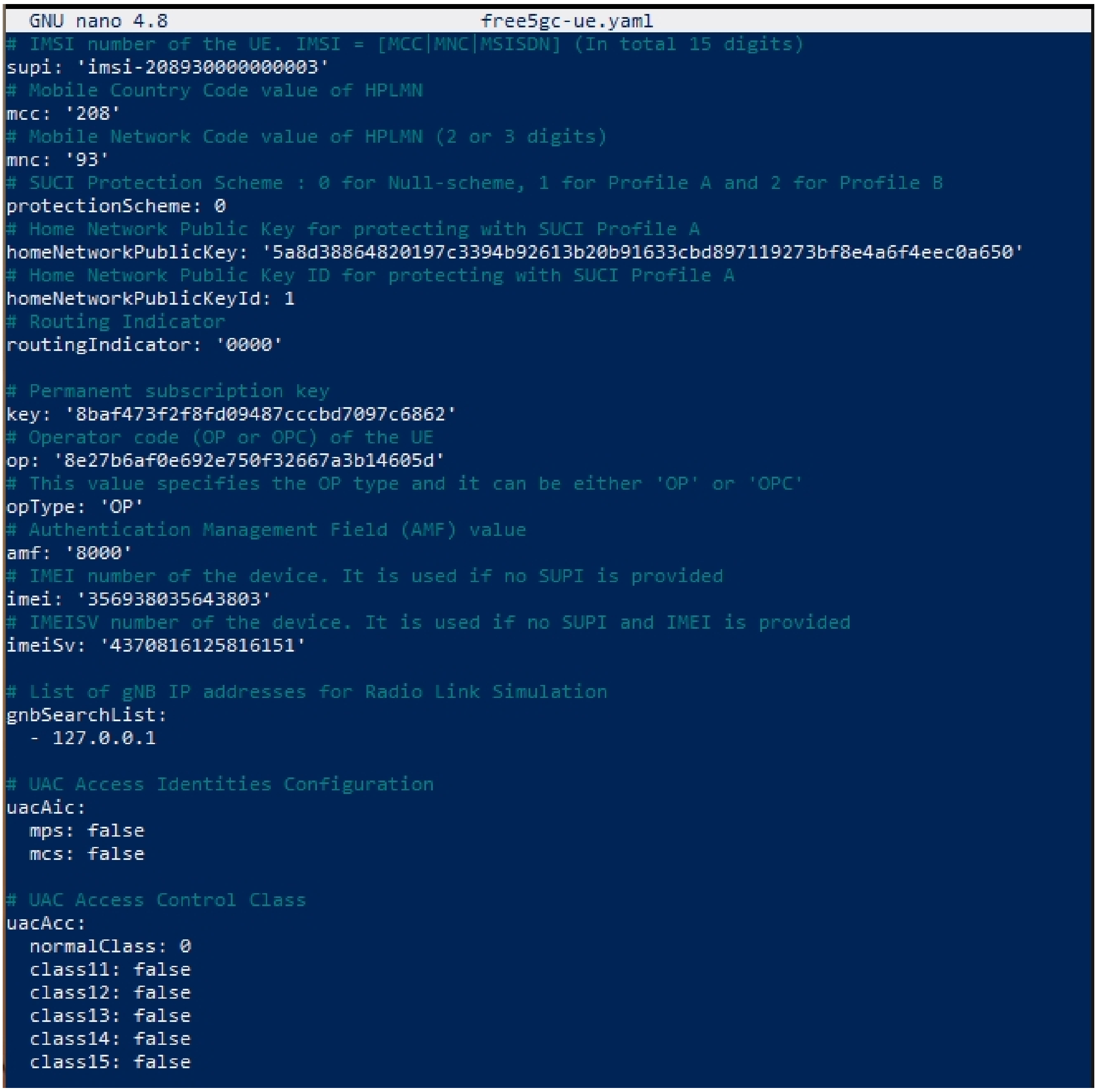


Figure 18: Subscriber Info WebConsole



```
  GNU nano 4.8                         free5gc-ue.yaml
# IMSI number of the UE. IMSI = [MCC|MNC|MSISDN] (In total 15 digits)
supi: 'imsi-208930000000003'
# Mobile Country Code value of HPLMN
mcc: '208'
# Mobile Network Code value of HPLMN (2 or 3 digits)
mnc: '93'
# SUCI Protection Scheme : 0 for Null-scheme, 1 for Profile A and 2 for Profile B
protectionScheme: 0
# Home Network Public Key for protecting with SUCI Profile A
homeNetworkPublicKey: '5a8d38864820197c3394b92613b20b91633cbd897119273bf8e4a6f4eec0a650'
# Home Network Public Key ID for protecting with SUCI Profile A
homeNetworkPublicKeyId: 1
# Routing Indicator
routingIndicator: '0000'

# 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'
# Authentication Management Field (AMF) value
amf: '8000'
# IMEI number of the device. It is used if no SUPI is provided
imei: '356938035643803'
# IMEISV number of the device. It is used if no SUPI and IMEI is provided
imeiSv: '4370816125816151'

# List of gNB IP addresses for Radio Link Simulation
gnbSearchList:
  - 127.0.0.1

# UAC Access Identities Configuration
uacAic:
  mps: false
  mcs: false

# UAC Access Control Class
uacAcc:
  normalClass: 0
  class11: false
  class12: false
  class13: false
  class14: false
  class15: false
```

Figure 19: Subscriber info UERANSIM

4.7 Testing the functionality

To test the functionality, the corresponding steps in the documentation are followed and the recommended number of SSH *connections are* established so that all services can run in parallel and be analyzed. For this purpose, one connection to the *Free5GC* VM and three connections to *UERANSIM* are established. In the Free5GC *terminal, Free5GC* is started and in the three *UERANSIM terminals, the gNB* (see Figure 20) on the one hand and the *UE* (see Figure 21) on the other hand are started. the third connection to the VM is used to check the network interfaces and to send appropriate commands. (cf. Free5GC o.D.)

```
ubuntu@ueransim:~/UERANSIM$ build/nr-gnb -c config/free5gc-gnb.yaml
UERANSIM v3.2.6
[2023-06-20 16:27:21.536] [sctp] [info] Trying to establish SCTP connection... (192.168.56.101:38412)
[2023-06-20 16:27:21.553] [sctp] [info] SCTP connection established (192.168.56.101:38412)
[2023-06-20 16:27:21.553] [sctp] [debug] SCTP association setup ascId[3]
[2023-06-20 16:27:21.553] [ngap] [debug] Sending NG Setup Request
[2023-06-20 16:27:21.560] [ngap] [debug] NG Setup Response received
[2023-06-20 16:27:21.560] [ngap] [info] NG Setup procedure is successful
[2023-06-20 16:27:40.687] [rrc] [debug] UE[1] new signal detected
[2023-06-20 16:27:43.197] [rrc] [info] RRC Setup for UE[1]
[2023-06-20 16:27:43.197] [ngap] [debug] Initial NAS message received from UE[1]
[2023-06-20 16:27:43.273] [ngap] [debug] Initial Context Setup Request received
[2023-06-20 16:27:43.553] [ngap] [info] PDU session resource(s) setup for UE[1] count[1]
```

Figure 20: UERANSIM gNB

```
ubuntu@ueransim:~/UERANSIM$ sudo build/nr-ue -c config/free5gc-ue.yaml
[sudo] password for ubuntu:
UERANSIM v3.2.6
[2023-06-20 16:27:40.687] [nas] [info] UE switches to state [MM-DEREGISTERED/PLMN-SEARCH]
[2023-06-20 16:27:40.687] [rrc] [debug] New signal detected for cell[1], total [1] cells in coverage
[2023-06-20 16:27:40.688] [nas] [info] Selected plmn[208/93]
[2023-06-20 16:27:43.187] [rrc] [info] Selected cell plmn[208/93] tac[1] category[SUITABLE]
[2023-06-20 16:27:43.187] [nas] [info] UE switches to state [MM-DEREGISTERED/PS]
[2023-06-20 16:27:43.187] [nas] [info] UE switches to state [MM-DEREGISTERED/NORMAL-SERVICE]
[2023-06-20 16:27:43.187] [nas] [debug] Initial registration required due to [MM-DEREG-NORMAL-SERVICE]
[2023-06-20 16:27:43.187] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-06-20 16:27:43.187] [nas] [debug] Sending Initial Registration
[2023-06-20 16:27:43.197] [nas] [info] UE switches to state [MM-REGISTER-INITIATED]
[2023-06-20 16:27:43.197] [rrc] [debug] Sending RRC Setup Request
[2023-06-20 16:27:43.197] [rrc] [info] RRC connection established
[2023-06-20 16:27:43.197] [rrc] [info] UE switches to state [RRC-CONNECTED]
[2023-06-20 16:27:43.197] [nas] [info] UE switches to state [CM-CONNECTED]
[2023-06-20 16:27:43.220] [nas] [debug] Authentication Request received
[2023-06-20 16:27:43.220] [nas] [debug] Sending Authentication Failure due to SQN out of range
[2023-06-20 16:27:43.232] [nas] [debug] Authentication Request received
[2023-06-20 16:27:43.239] [nas] [debug] Security Mode Command received
[2023-06-20 16:27:43.239] [nas] [debug] Selected integrity[2] ciphering[0]
[2023-06-20 16:27:43.274] [nas] [debug] Registration accept received
[2023-06-20 16:27:43.274] [nas] [info] UE switches to state [MM-REGISTERED/NORMAL-SERVICE]
[2023-06-20 16:27:43.274] [nas] [debug] Sending Registration Complete
[2023-06-20 16:27:43.274] [nas] [info] Initial Registration is successful
[2023-06-20 16:27:43.274] [nas] [debug] Sending PDU Session Establishment Request
[2023-06-20 16:27:43.274] [nas] [debug] UAC access attempt is allowed for identity[0], category[MO_sig]
[2023-06-20 16:27:43.553] [nas] [debug] PDU Session Establishment Accept received
[2023-06-20 16:27:43.553] [nas] [info] PDU Session establishment is successful PSI[1]
[2023-06-20 16:27:43.565] [app] [info] Connection setup for PDU session[1] is successful, TUN interface[uesimtun0, 10.60
.0.1] is up.
```

Figure 21: UERANSIM UE

The *ifconfig* command is used to check whether a corresponding tunnel has been created. (see Figure 22)

```
uesimtun0: flags=369<UP,POINTOPOINT,NOTRAILERS,RUNNING,PROMISC>  mtu 1400
        inet 10.60.0.1  netmask 255.255.255.255  destination 10.60.0.1
        inet6 fe80::deef:237:ff3f:1295  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 7  bytes 448 (448.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 22: Interface uesimtun0

The functionality of *Free5GC* is confirmed by sending a *ping*. (see Figure 23)

ping –I uesimtun0 google.com

```
ubuntu@ueransim:~$ ping -I uesimtun0 google.com
PING google.com (142.250.185.110) from 10.60.0.1 uesimtun0: 56(84) bytes of data.
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=1 ttl=58 time=10.4 ms
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=2 ttl=58 time=12.2 ms
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=3 ttl=58 time=12.7 ms
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=4 ttl=58 time=11.9 ms
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=5 ttl=58 time=11.7 ms
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=6 ttl=58 time=12.8 ms
^X64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=7 ttl=58 time=12.1 ms
64 bytes from fra16s49-in-f14.1e100.net (142.250.185.110): icmp_seq=8 ttl=58 time=11.1 ms
^C
--- google.com ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7046ms
rtt min/avg/max/mdev = 10.408/11.849/12.774/0.744 ms
```

Figure 23: uesimtun0 ping to google.com

In addition, the registration and connection of the *UE* can be viewed via the WebConsole. (see Figure 24 and Figure 25)

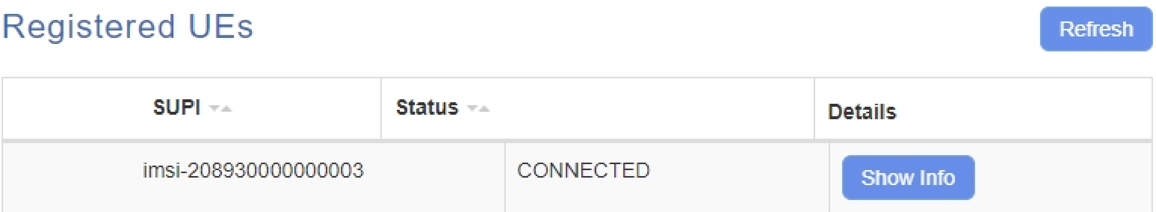


Figure 24: Successfully registered UE on the WebConsole

AMF Information [SUPI:imsi-208930000000003]

Information Entity	Value
AccessType	3GPP_ACCESS
CmState	CONNECTED
Guti	20893cafe0000000001
Mcc	208
Mnc	93
Supi	imsi-208930000000003
Tac	000001
Dnn	internet
PduSessionId	1
Sd	010203
SmContextRef	urn:uuid:bbc2e78a-c3f1-4043-b2f6-2353705d2039
Sst	1

SMF Information [SUPI:imsi-208930000000003]

Information Entity	Value
AnType	3GPP_ACCESS
Dnn	internet
LocalSEID	
PDUAddress	10.60.0.1
PDUSessionID	1
RemoteSEID	
Sd	010203
Sst	1

Figure 25: Registered UE details

5 Summary and Outlook

The implementation of *Free5GC* in combination with *UERANSIM* offers a great opportunity to gain initial practical experience in dealing with 5G technology and to put theoretical knowledge into practice. Starting from setting up and configuring the virtual machines to testing the network functions, knowledge can be deepened in many areas. For this purpose, tests were carried out and network traffic was monitored and analyzed to get a first impression of the processes involved in registering a UE in a 5G system. Also, a better understanding about the overall architecture in a 5G system is gained. The combination offers many ways to deepen and build knowledge in different areas and thus has great potential. In addition, the project has provided a good insight into 5G technology that can be built upon in the future.

The project can be extended in many directions in the future. Among other things, a *Non 3GPP Interworking Function (N3IWF)* can be implemented to enable interoperability between the 5G network and non 3GPP networks (such as *WiFi*, *Ethernet* and *DSL*). In addition, the test environment can also be extended to allow communication between the Free5GC *core network* and other core networks. Finally, the *AMF* and *SMF* in *Free5GC* do not yet have an interface to *NEF*, which can be implemented with *Go.*

Free5GC in combination with *UERANSIM* have many possibilities for further development and can be used very well for testing and learning purposes.

6 List of abbreviations

0...9

3GPP 3rd Generation Partnership Project
5G 5th Generation

A

AF Application Function
AMF Access and Mobility Management Function
API Application Programming Interface
AUSF Authentication Server Function

D

DSL Digital Subscriber Line

G

gNB gNodeB
GTP GPRS Tunneling Protocol
GUTI Global Unique Temporary Identity

I

IoT Internet of Things
IP Internet Protocol
ISO International Organization for Standardization

J

JSON JavaScript Object Notation

M

MAC Media Access Control

N

N3IWF Non-3GPP Interworking Function
NAT Network Address Translation
NEF Network Exposure Function
NSSF Network Slice Selection Function

O

OP-Type Operation-Type

P

PCAP Packet Capture
PCF Policy Control Function
PDU Protocol Data Unit
PFCP Packet Forwarding Control Protocol

Q

QoS Quality of Service

R

RAN	Radio Access Network

S

SCP	Secure Copy
SMF	Session Management Function
SQL	Structured Query Language
SSH	Secure Shell

U

UDM	Unified Data Management
UDP	User Datagram Protocol
UE	User Equipment
UPF	User Plane Function

V

VM	Virtual Machine

W

WiFi	Wireless Fidelity

7 List of Figures

8 References

1. Oracle (n.d.): Welcome to VirtualBox.org!, https://www.virtualbox.org [accessed 07.05.2023].

2. Oracle (n.d.): Why is virtualization useful?, https://www.virtualbox.org/manual/ch01.html#virt-why-useful [accessed 07.05.2023].

3. Kurek, T (24.04.2020): Ubuntu Server 20.04 LTS: stability, security and more, https://ubuntu.com/blog/ubuntu-server-20-04 [retrieved 07/05/2023].

4. Google (n.d.): Documentation, https://go.dev/doc/ [accessed 05/12/2023].

5. Google (n.d.): Download and Install, https://go.dev/doc/install [retrieved 12.05.2023].

6. MongoDB (n.d.), https://www.mongodb.com/de-de [accessed 06/18/2023].

7. Ross, J. (07.07.2019): Installing tshark only, https://tshark.dev/setup/install/ [accessed 05/14/2023].

8. Free5GC (n.d.): Create an Ubuntu Server, https://free5gc.org/guide/1-vm-en/#3-create-a- ubuntu-server-vm [accessed 05/14/2023].

9. Free5GC (n.d.): Install Free5GC, https://free5gc.org/guide/3-install-free5gc/ [accessed 05/14/2023].

10. Free5GC (n.d.): Test Free5GC, https://free5gc.org/guide/4-test-free5gc/ [retrieved 27/05/2023].

11. Free5GC (n.d.): Install UERANSIM VM, https://free5gc.org/guide/5-install-ueransim/#1-install-ueramsim-vm [accessed 06/10/2023].

12. Aligungr (01/24/2021): Installation, https://github.com/aligungr/UERANSIM/wiki/Installation [retrieved 06/14/2023]

13. Trick, U. (n.d.): 5th Generation Mobile Networks, https://e-technik.org/frame_lectures.htm [accessed 20.06.2023].

14. Sultan, A. (11.10.2022): 5G System Overview, https://www.3gpp.org/technologies/5g-system-overview [retrieved 09.06.2023].

15. Dryjanski, M. (02.03.2018): 5G Core Network Functions, https://www.grandmetric.com/5g-core-network-functions/ [accessed 09.06.2023].

16. Köpsell, Stefan/Ruzhanskiy, Andrey/Hecker, Andreas/Stachorra, Dirk (21.02.2022): Open-RAN Risk Analysis, https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/5G/5GRAN-riskanalysis.pdf? blob=publicationFile&v=5 [retrieved 21.06.2023].

17. 3GPP (FEB. 19, 2021): NG-RAN Architecture, https://www.3gpp.org/news-events/3gpp- news/ng-ran-architecture [accessed 06/19/2023].

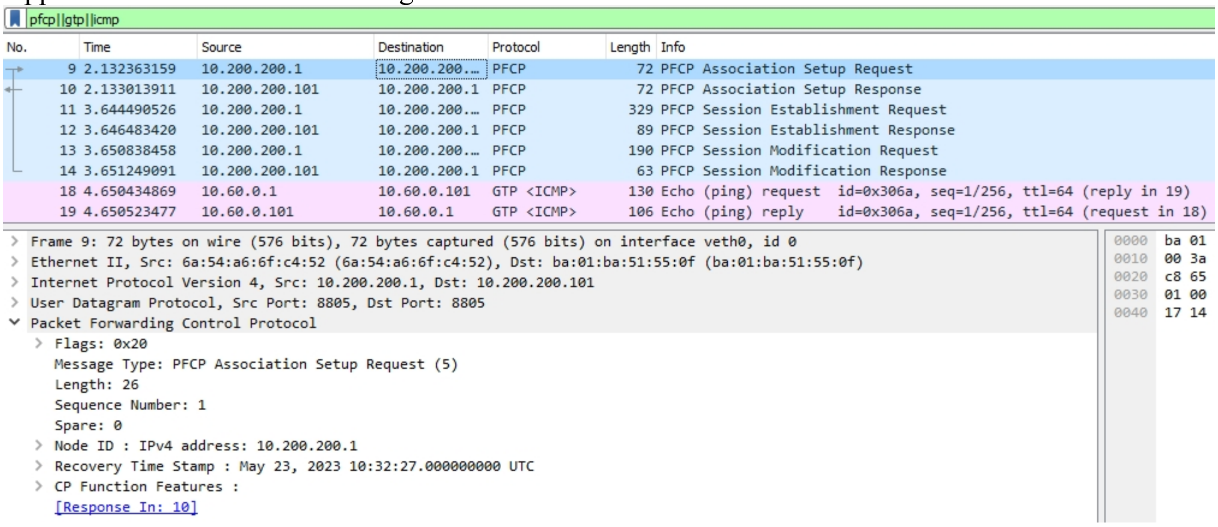
18. Huo, Yiming/Dong, Xiaodai/Xu, Wei (n.d.): 5G Cellular User Equipment: From Theory to Practical Hardware Design, https://ieeexplore.ieee.org/document/7983335 [accessed 06/18/2023].

19. Devopedia (02/05/2023): Packet Forwarding Control Protocol, https://devopedia.org/packet- forwarding-control-protocol [retrieved on 06/22/2023].

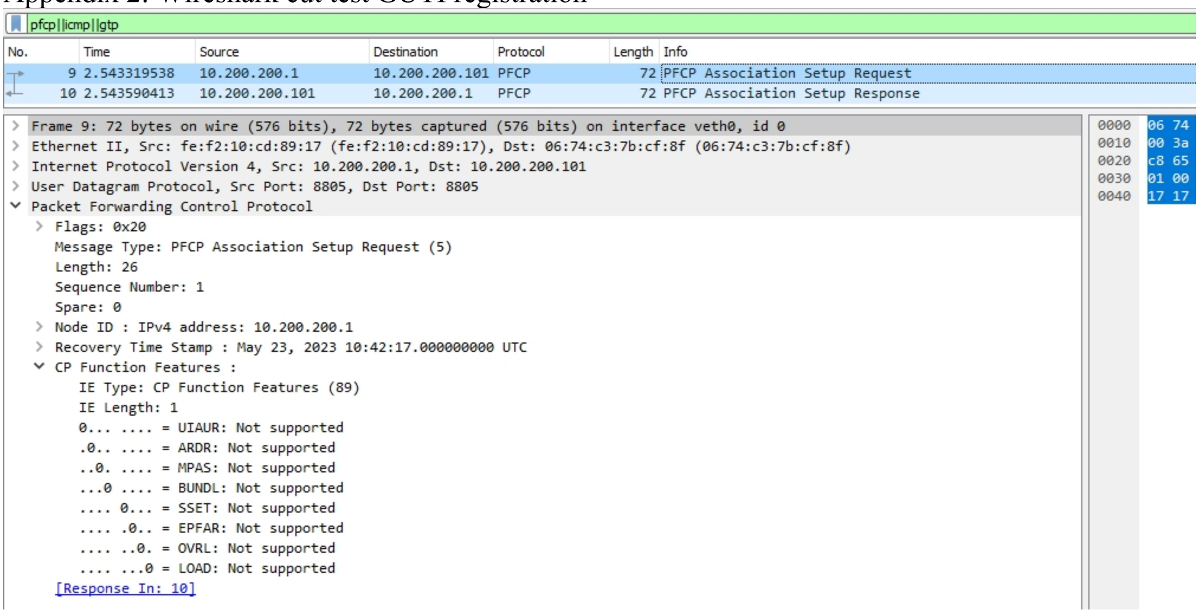
20. ETSI (2020-11): ETSI TS 129 244, https://www.etsi.org/deliver/etsi_ts/129200_129299/129244/16.05.00_60/ts_129244v160500p.pdf [retrieved 06/26/2023]

9 Appendix

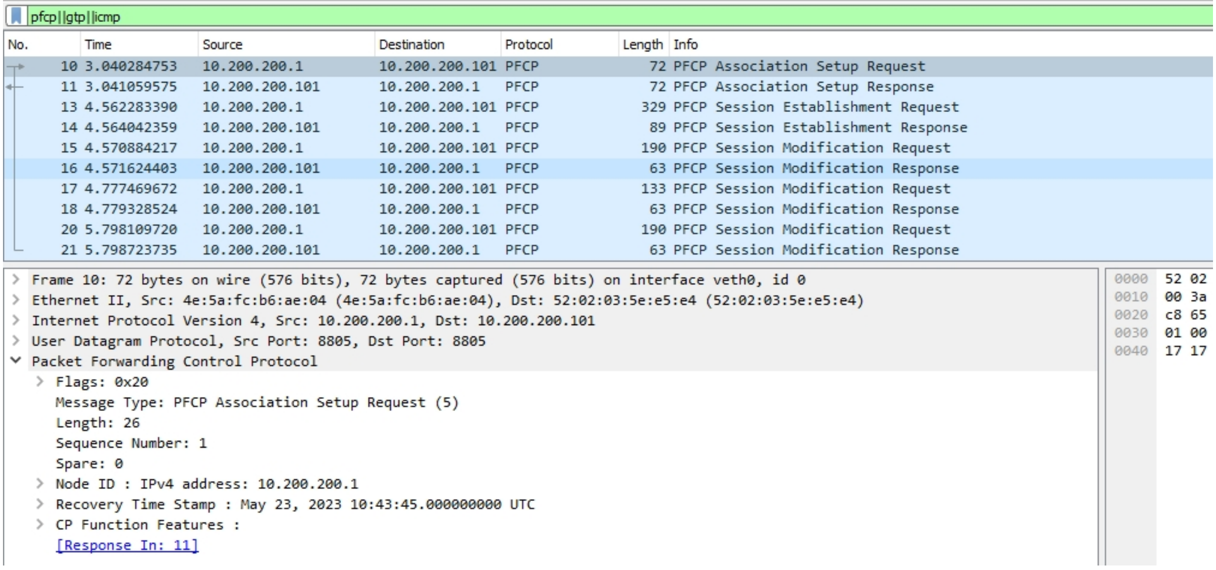
Appendix 1: Wireshark cut test registration



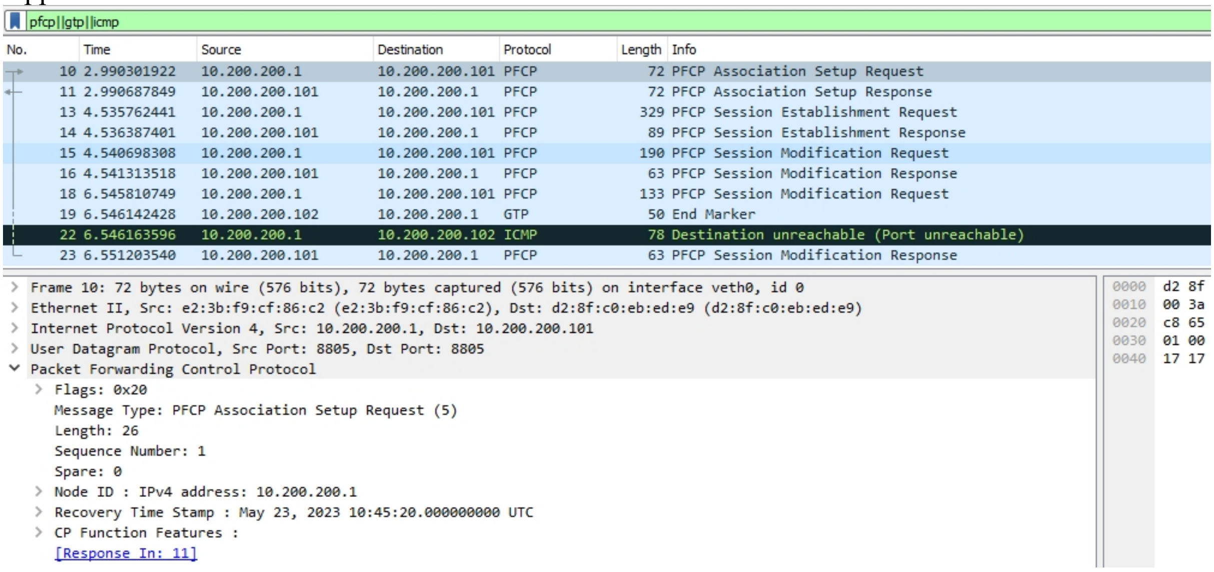
Appendix 2: Wireshark cut test GUTI registration



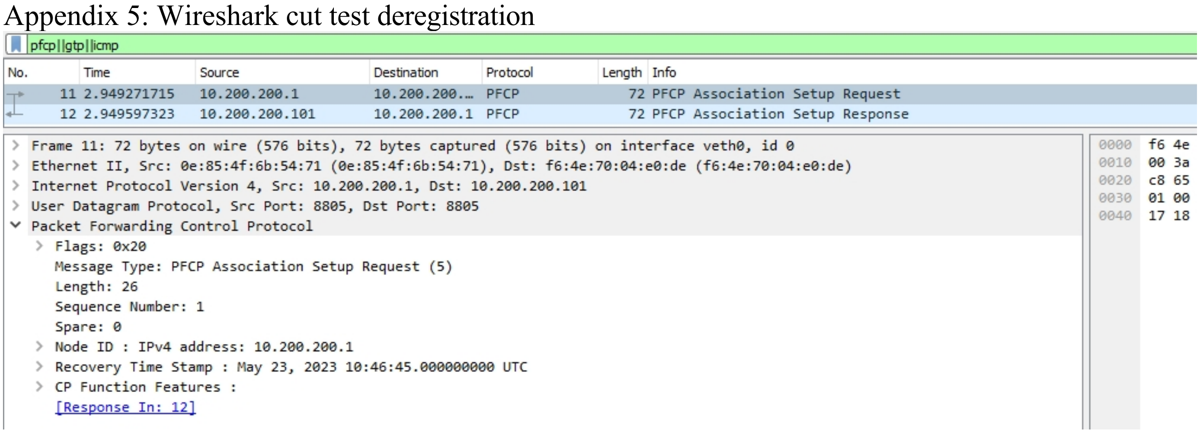
Appendix 3: Wireshark Cross-Section Test Service Request



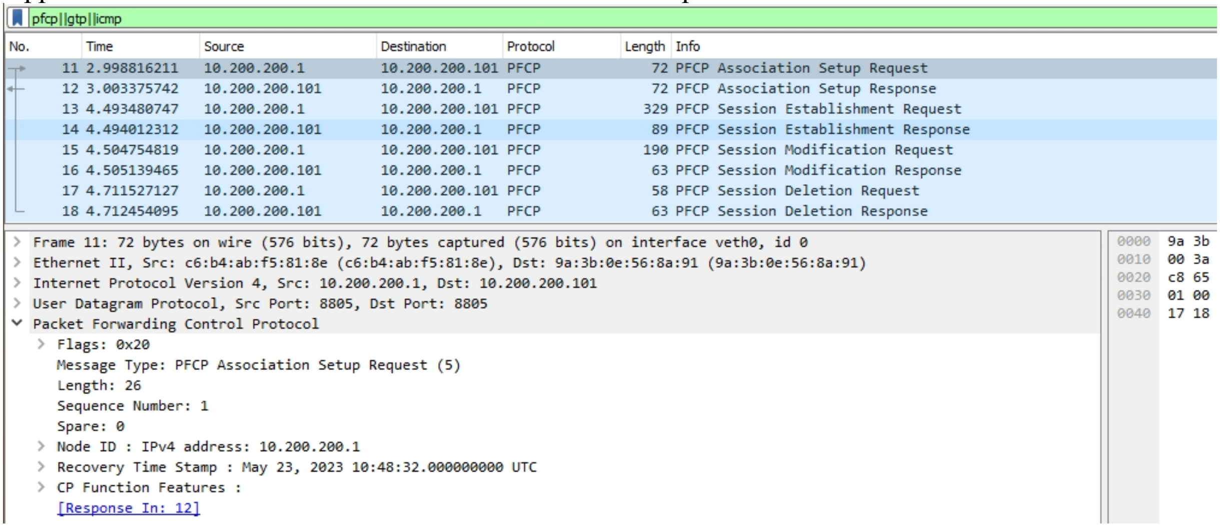
Appendix 4: Wireshark Cut Test-XnHandover



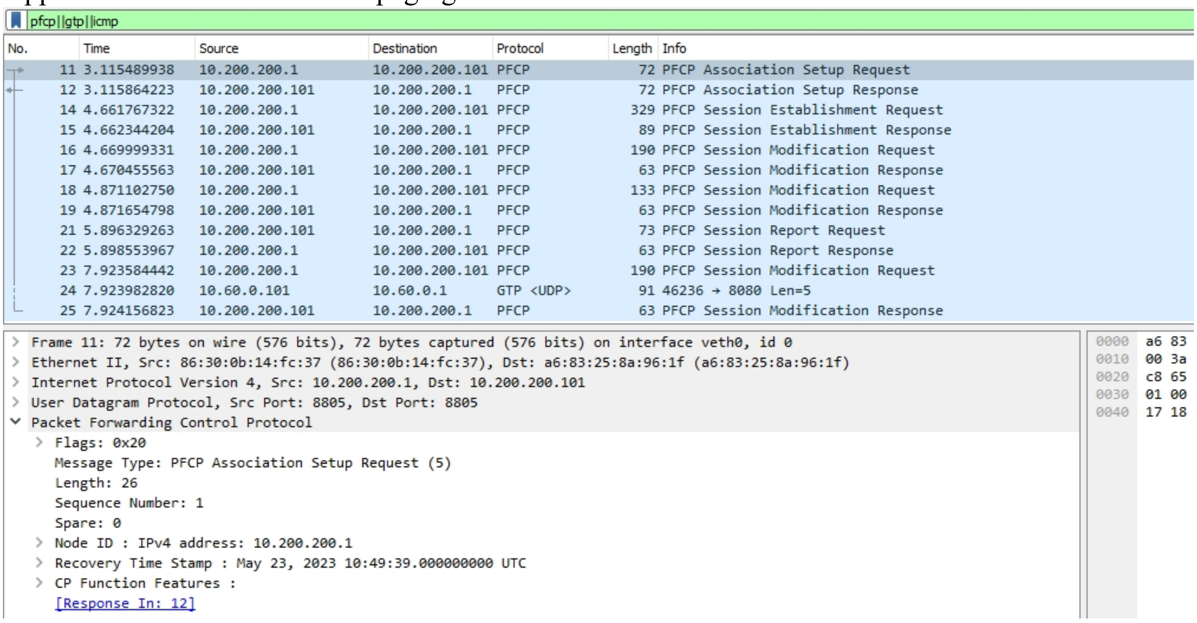
Appendix 5: Wireshark cut test deregistration



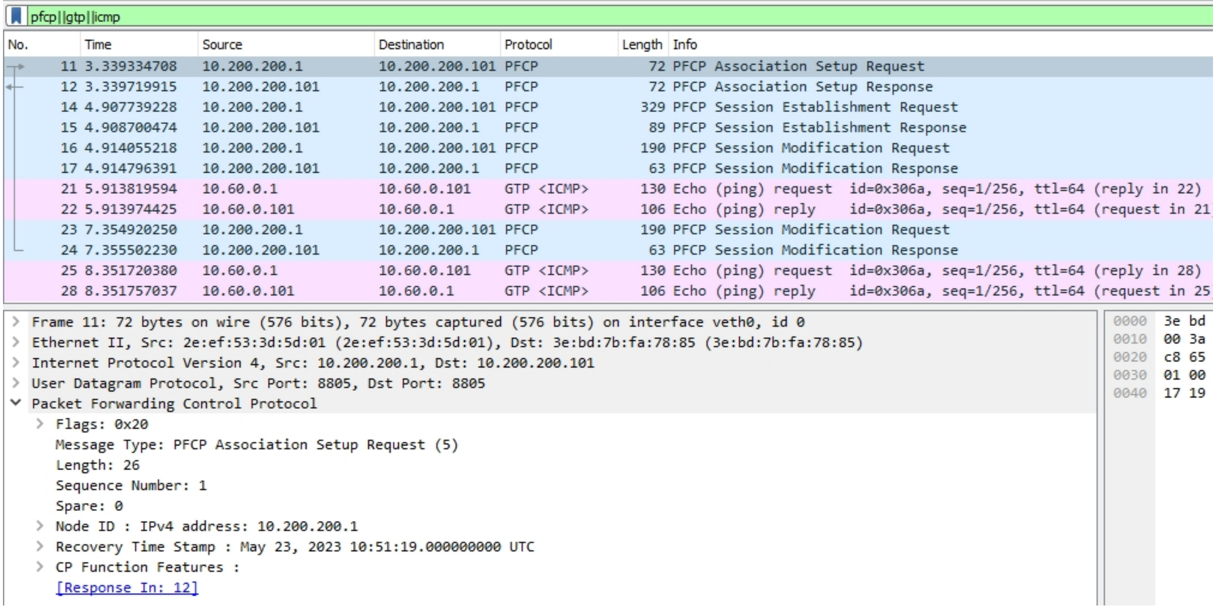
Appendix 6: Wireshark Cut Test PDU Session Release Request



Appendix 7: Wireshark cut test paging



Appendix 8: Wireshark cross-section test N2 handover



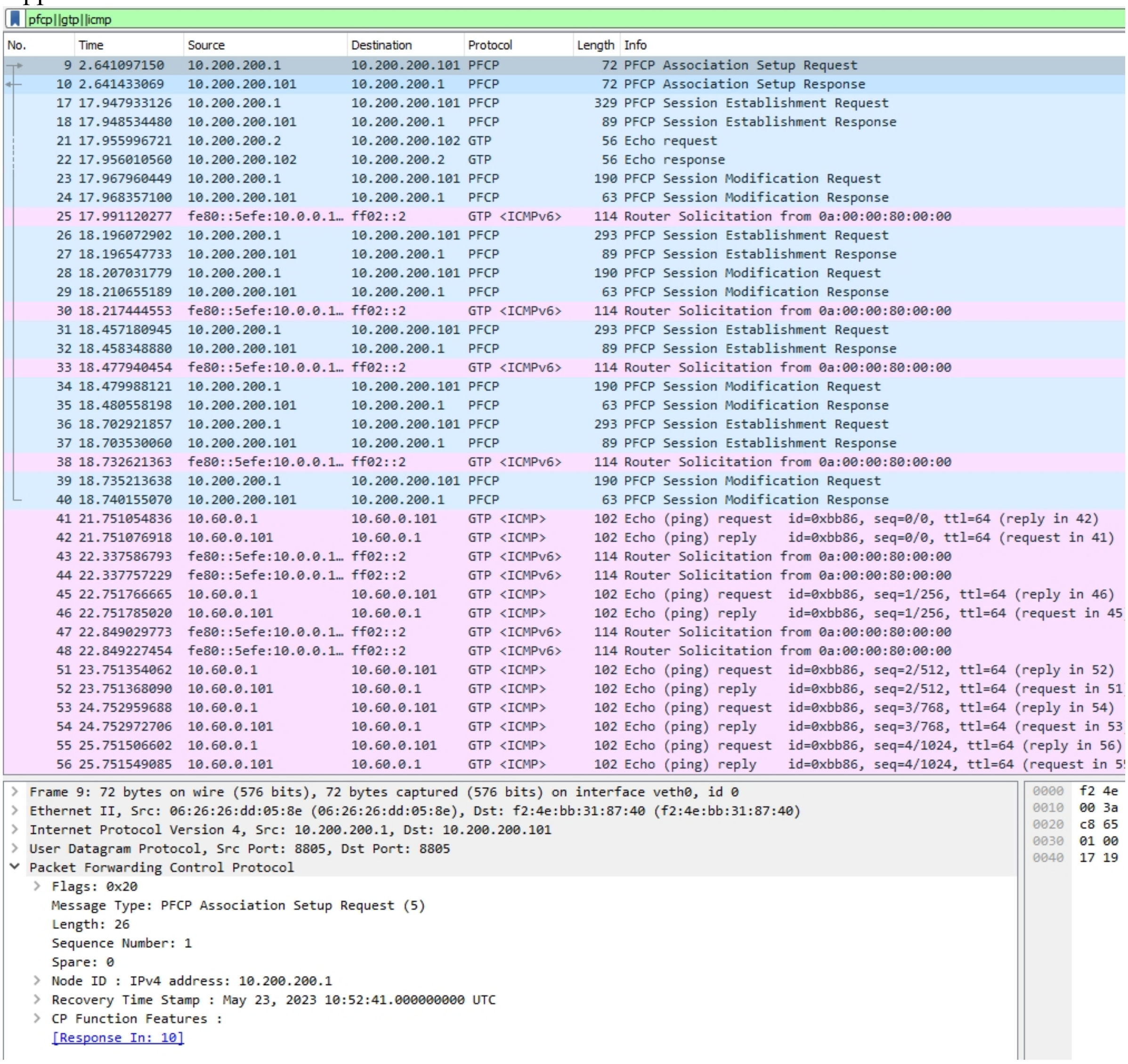
pfcp||gtp||icmp

No.	Time	Source	Destination	Protocol	Length	Info
11	3.339334708	10.200.200.1	10.200.200.101	PFCP	72	PFCP Association Setup Request
12	3.339719915	10.200.200.101	10.200.200.1	PFCP	72	PFCP Association Setup Response
14	4.907739228	10.200.200.1	10.200.200.101	PFCP	329	PFCP Session Establishment Request
15	4.908700474	10.200.200.101	10.200.200.1	PFCP	89	PFCP Session Establishment Response
16	4.914055218	10.200.200.1	10.200.200.101	PFCP	190	PFCP Session Modification Request
17	4.914796391	10.200.200.101	10.200.200.1	PFCP	63	PFCP Session Modification Response
21	5.913819594	10.60.0.1	10.60.0.101	GTP <ICMP>	130	Echo (ping) request id=0x306a, seq=1/256, ttl=64 (reply in 22)
22	5.913974425	10.60.0.101	10.60.0.1	GTP <ICMP>	106	Echo (ping) reply id=0x306a, seq=1/256, ttl=64 (request in 21
23	7.354920250	10.200.200.1	10.200.200.101	PFCP	190	PFCP Session Modification Request
24	7.355502230	10.200.200.101	10.200.200.1	PFCP	63	PFCP Session Modification Response
25	8.351720380	10.60.0.1	10.60.0.101	GTP <ICMP>	130	Echo (ping) request id=0x306a, seq=1/256, ttl=64 (reply in 28)
28	8.351757037	10.60.0.101	10.60.0.1	GTP <ICMP>	106	Echo (ping) reply id=0x306a, seq=1/256, ttl=64 (request in 25

```
> Frame 11: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface veth0, id 0
> Ethernet II, Src: 2e:ef:53:3d:5d:01 (2e:ef:53:3d:5d:01), Dst: 3e:bd:7b:fa:78:85 (3e:bd:7b:fa:78:85)
> Internet Protocol Version 4, Src: 10.200.200.1, Dst: 10.200.200.101
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
v Packet Forwarding Control Protocol
  > Flags: 0x20
    Message Type: PFCP Association Setup Request (5)
    Length: 26
    Sequence Number: 1
    Spare: 0
  > Node ID : IPv4 address: 10.200.200.1
  > Recovery Time Stamp : May 23, 2023 10:51:19.000000000 UTC
  > CP Function Features :
    [Response In: 12]
```

```
0000  3e bd
0010  00 3a
0020  c8 65
0030  01 00
0040  17 19
```

Appendix 9: Wireshark cut test non-3GPP



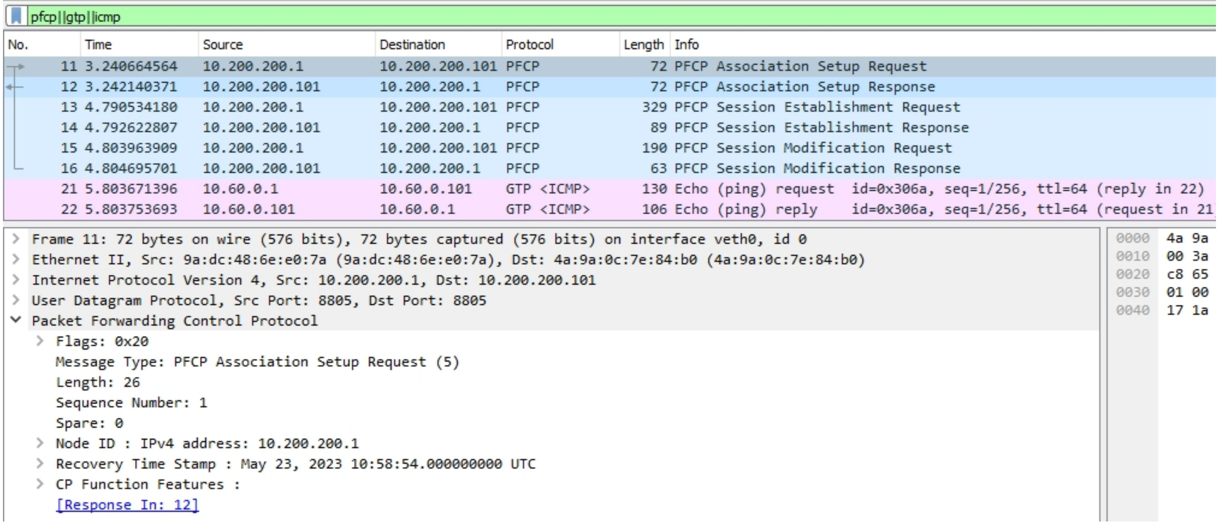
pfcp||gtp||icmp

No.	Time	Source	Destination	Protocol	Length	Info
9	2.641097150	10.200.200.1	10.200.200.101	PFCP	72	PFCP Association Setup Request
10	2.641433069	10.200.200.101	10.200.200.1	PFCP	72	PFCP Association Setup Response
17	17.947933126	10.200.200.1	10.200.200.101	PFCP	329	PFCP Session Establishment Request
18	17.948534480	10.200.200.101	10.200.200.1	PFCP	89	PFCP Session Establishment Response
21	17.955996721	10.200.200.2	10.200.200.102	GTP	56	Echo request
22	17.956010560	10.200.200.102	10.200.200.2	GTP	56	Echo response
23	17.967960449	10.200.200.1	10.200.200.101	PFCP	190	PFCP Session Modification Request
24	17.968357100	10.200.200.101	10.200.200.1	PFCP	63	PFCP Session Modification Response
25	17.991120277	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
26	18.196072902	10.200.200.1	10.200.200.101	PFCP	293	PFCP Session Establishment Request
27	18.196547733	10.200.200.101	10.200.200.1	PFCP	89	PFCP Session Establishment Response
28	18.207031779	10.200.200.1	10.200.200.101	PFCP	190	PFCP Session Modification Request
29	18.210655189	10.200.200.101	10.200.200.1	PFCP	63	PFCP Session Modification Response
30	18.217444553	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
31	18.457180945	10.200.200.1	10.200.200.101	PFCP	293	PFCP Session Establishment Request
32	18.458348880	10.200.200.101	10.200.200.1	PFCP	89	PFCP Session Establishment Response
33	18.477940454	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
34	18.479988121	10.200.200.1	10.200.200.101	PFCP	190	PFCP Session Modification Request
35	18.480558198	10.200.200.101	10.200.200.1	PFCP	63	PFCP Session Modification Response
36	18.702921857	10.200.200.1	10.200.200.101	PFCP	293	PFCP Session Establishment Request
37	18.703530060	10.200.200.101	10.200.200.1	PFCP	89	PFCP Session Establishment Response
38	18.732621363	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
39	18.735213638	10.200.200.1	10.200.200.101	PFCP	190	PFCP Session Modification Request
40	18.740155070	10.200.200.101	10.200.200.1	PFCP	63	PFCP Session Modification Response
41	21.751054836	10.60.0.1	10.60.0.101	GTP <ICMP>	102	Echo (ping) request id=0xbb86, seq=0/0, ttl=64 (reply in 42)
42	21.751076918	10.60.0.101	10.60.0.1	GTP <ICMP>	102	Echo (ping) reply id=0xbb86, seq=0/0, ttl=64 (request in 41)
43	22.337586793	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
44	22.337757229	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
45	22.751766665	10.60.0.1	10.60.0.101	GTP <ICMP>	102	Echo (ping) request id=0xbb86, seq=1/256, ttl=64 (reply in 46)
46	22.751785020	10.60.0.101	10.60.0.1	GTP <ICMP>	102	Echo (ping) reply id=0xbb86, seq=1/256, ttl=64 (request in 45
47	22.849029773	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
48	22.849227454	fe80::5efe:10.0.0.1…	ff02::2	GTP <ICMPv6>	114	Router Solicitation from 0a:00:00:80:00:00
51	23.751354062	10.60.0.1	10.60.0.101	GTP <ICMP>	102	Echo (ping) request id=0xbb86, seq=2/512, ttl=64 (reply in 52)
52	23.751368090	10.60.0.101	10.60.0.1	GTP <ICMP>	102	Echo (ping) reply id=0xbb86, seq=2/512, ttl=64 (request in 51
53	24.752959688	10.60.0.1	10.60.0.101	GTP <ICMP>	102	Echo (ping) request id=0xbb86, seq=3/768, ttl=64 (reply in 54)
54	24.752972706	10.60.0.101	10.60.0.1	GTP <ICMP>	102	Echo (ping) reply id=0xbb86, seq=3/768, ttl=64 (request in 53
55	25.751506602	10.60.0.1	10.60.0.101	GTP <ICMP>	102	Echo (ping) request id=0xbb86, seq=4/1024, ttl=64 (reply in 56)
56	25.751549085	10.60.0.101	10.60.0.1	GTP <ICMP>	102	Echo (ping) reply id=0xbb86, seq=4/1024, ttl=64 (request in 5

```
> Frame 9: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface veth0, id 0
> Ethernet II, Src: 06:26:26:dd:05:8e (06:26:26:dd:05:8e), Dst: f2:4e:bb:31:87:40 (f2:4e:bb:31:87:40)
> Internet Protocol Version 4, Src: 10.200.200.1, Dst: 10.200.200.101
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
v Packet Forwarding Control Protocol
  > Flags: 0x20
    Message Type: PFCP Association Setup Request (5)
    Length: 26
    Sequence Number: 1
    Spare: 0
  > Node ID : IPv4 address: 10.200.200.1
  > Recovery Time Stamp : May 23, 2023 10:52:41.000000000 UTC
  > CP Function Features :
    [Response In: 10]
```

```
0000  f2 4e
0010  00 3a
0020  c8 65
0030  01 00
0040  17 19
```

Appendix 10: Wireshark cut test re-synchronization



Appendix 11: Wireshark Cut Test Request Two PDU Sessions