

A Reference Deployment of a Minimal Open-Source Private Industry and Campus 5G Standalone (SA) System

Fabian John

Technische Hochschule Lübeck –
University of Applied Sciences
Department E&I, Center of
Excellence CoSA
Lübeck, Germany
ORCID: <https://orcid.org/0000-0002-7009-0924>

Jörg Schuljak

Technische Hochschule Lübeck –
University of Applied Sciences
Department E&I, Center of
Excellence CoSA
Lübeck, Germany
joerg.schuljak@th-luebeck.de

Lars B. Vosteen

Technische Hochschule Lübeck –
University of Applied Sciences
Department E&I, Center of
Excellence CoSA
Lübeck, Germany
lars.vosteen@th-luebeck.de

Björn Sievers

Technische Hochschule Lübeck –
University of Applied Sciences
Department E&I, Center of
Excellence CoSA
Lübeck, Germany
bjoern.sievers@th-luebeck.de

Andreas Hanemann

Technische Hochschule Lübeck –
University of Applied Sciences
Department E&I, CoSA
Lübeck, Germany
andreas.hanemann@th-luebeck.de

Horst Hellbrück

Technische Hochschule Lübeck –
University of Applied Sciences
Department E&I, Center of
Excellence CoSA
Lübeck, Germany
horst.hellbrueck@th-luebeck.de

Abstract— With the fifth-generation new radio (5G NR) being ready for deployment and with the possibility of acquiring private licenses, 5G is ready for campus deployment in industry and academia. As industry and academia do not have their own LTE networks, especially 5G standalone (SA) implementations gain importance. However, SA deployments include all components from the user equipment to radio units and core functionality in a large distributed system to work properly. Since commercial systems are costly, it is worth considering open-source implementations. We face the following challenges: many open-source projects are yet in the development phase, it is challenging to deploy 5G systems because hardware and software continuously evolve, documentation is incomplete, and 5G SA is a new technology. A small community runs a continuously changing deployment in a development branch. From the lessons learned, our contribution to the community is to provide a well-documented deployment based on existing tutorials and sometimes in code research and configuration file tests like try and error. For evaluation and proof of concept, we built a 5G system at the Technische Hochschule Lübeck - University of Applied Sciences with the provision of all information to deploy the system step by step and perform integration tests. We provide a complete solution for low costs of 10k EUR with off-the-shelf components and provide basic connectivity and iperf measurement results for latency and throughput as a reference for own deployments.

Keywords— 5G, Campus Network, OpenAirInterface, Private Network, Standalone, SA.

I. INTRODUCTION

Telecommunication or cellular systems have evolved in the last twenty years from dedicated real-time systems with specialized

hardware, software, and protocols to loosely-coupled large flexible distributed systems. The distributed 5G system consists mainly of two segments (1) radio access network and (2) core network. The new generation radio access network (NG-RAN) consists of several new generation Node Bs (gNBs), which are radio base stations connected to the 5G core network (5GC) and each other. The gNB has three main parts: the Centralized Unit (CU), the Distributed Unit (DU), and the Radio Unit (RU), which are deployed as a part of the distributed system. Radio Units are named remote radio heads (RRH) in some publications.

In the past, the core network was structured as a mesh network with dedicated devices and links, whereas nowadays, an ALL-IP-Network with function virtualization is deployed. The core network is based on cloud technology with software-defined networking stacks or a service-based architecture. Although not explicitly standardized in the 5G releases, service management and orchestration (SMO) is a recommended functionality of the 5G system to maintain and update software components and services and support extending RAN by deploying new base stations [14]. Today's open-source systems are mainly based on off-the-shelf hardware and software-defined radios that provide flexibility by software and allow open innovation on all levels in this market segment.

Our work focuses on the 5G standalone (SA) system because SA networks outperform non-standalone (NSA) networks in terms of decreased device power consumption, less deployment complexity, and lower costs [1], [2]. We describe

a 5G SA system deployment (according to 3GPP Release 16) that covers minimal basic functionality. The proposed system is a functional prototype enabling new 5G users from industry and academia to start running their own private 5G networks. The new community members with industry and academic backgrounds get into private 5G systems without decades of experience and knowledge in mobile communications.

In our experience, there are currently few private 5G industrial systems in Germany. In the Schleswig-Holstein region, we currently operate the first private 5G SA system at the Technische Hochschule Lübeck. To the best of our knowledge, the deployment of a 5G SA open-source end-to-end system has not yet been investigated and described. Private 5G systems are either NSA or not open-source [13] [15].

Why do we provide that guide? An open-source 5G SA system deployment is still challenging because development and documentation are in progress, and detailed knowledge of mobile networks operation and architecture is required. In this work, we refer to deployment examples conducted to the OpenAirInterface open-source project, but the overall structure and components are not limited to this solution and can be used for other open-source projects. Our contributions are:

- We introduce a minimal deployment setup of a low cost 5G SA system for a campus network
- We list a complete set of suitable hardware components and open-source software
- We explain necessary configuration settings to get started with a private 5G SA system with Dos and Don'ts
- We provide dedicated test instructions to perform component, integration, system tests, and reference data during the deployment.

The paper is structured as follows. Section II presents the structure of a 5G SA system and its mandatory components. Section III introduces the selected hardware and gives further recommendations. Section IV describes the steps for deployment from 5G SA core, RRU/gNB, SIM card, and user equipment. The evaluation and reference data of our deployment are given in Section V. Section VI concludes the paper and gives directions to future work.

II. 5G SA STRUCTURE

This section presents the structure of a 5G SA system in general and the components deployed in our minimal private 5G SA system at the Technische Hochschule Lübeck. The structure is shown in Figure 1. A complete 5G SA system consists of the four major parts SMO, 5G Core, gNB, and UE. Each of the parts covers several 5G components organized as a service-based architecture (SBA).

The minimal 5G SA system we deployed at the Technische Hochschule Lübeck consists of the three lower parts 5G core, gNB, and UE. The SMO is not required to provide 5G base functionality. Still, the SMO deployment is planned to integrate

AI algorithms, manage policies, and manage resources in our 5G system in a future extension. The SBA enables distributed and centralized components deployment, especially in the 5G core. In our 5G system, we deployed the 5G core and gNBs to dedicated physical machines (see specification in Table I in Section III).

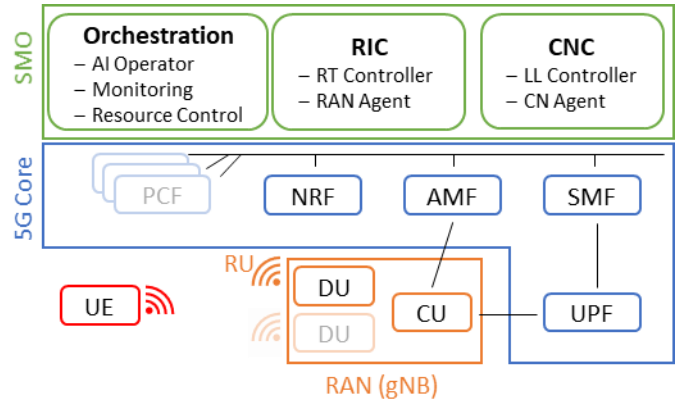


Fig. 1. Structure of a 5G network without CU/DU split.

To cover 5G base functionality, the 5G core services Access and Mobility Function (AMF), Session Management Function (SMF), User Plane Function (UPF), and Network Repository Function (NRF) are essential. For private 5G network cells operated in industry and academic environments, advanced 5G functionalities covered by additional core services (for instance, PCF), the 4G migration aspects [3], [4], handover procedures, and dual connectivity aspects [5] are not essential. Therefore, we focus on the essential services and functionality described in the following.

The Access and Mobility Function (AMF) is one of the main services in the 5G core and is mandatory for every 5G system. Its functionalities are authentication, authorization, ciphering and AS security control, and idle mobility handling.

The Session Management Function (SMF) covers the data plane control, Packet Data Unit (PDU) session control with the User Plane Function (UPF). The SMF establishes, maintains, and releases a PDU session and controls UE IP address allocation for routing data packets within the 5G system and supporting data reception and forwarding to outside networks (external packet data networks PDNs).

In the Network Repository Function (NRF), services like AMF register themselves. The NRF provides service registration and discovery for all services and functions.

As shown in Figure 1 the gNB is connected to the 5G core. The gNB covers radio access network functionality and builds the physical radio connection to the UE. Multiple gNBs connected to a single 5G core extend the system to a multi-cell setup to cover larger network areas. In the 5G specification, the gNB components are split into centralized and distributed units to increase the modularity and flexibility of the network design. Our private 5G SA system deployment consists of two gNBs,

each comprising CU and DU. We use low transmission power and serve two indoor cells with our 5G SA system.

Software-defined radios (SDR) are selected as radio units (RU) in our open 5G system to provide maximum flexibility for the 5G NR in the future. In our minimal deployment, we choose a basic SDR without MIMO and beamforming, which results in moderate throughput for a 5G system (refer Section V).

UEs connected to the 5G SA system must be compatible with the new technology. As of today, there are just a few UEs available that are compatible with 5G SA. Most of the 5G UEs are compatible with the 5G NSA standard at the moment. In our 5G SA system, we integrated the Quectel RM500Q-GL successfully.

The components of a 5G system are organized in two separate IP networks, the network for the 5G operator components (SMO, 5G Core, and gNBs) and the user network (UEs). Figure 2 shows the network structure of our 5G SA system deployment.

We apply a RaspberryPi as DHCP- and DNS-server, and Gateway to connect to the Internet for our operator network. This is a temporary solution, and we will switch to a dedicated backbone subnet to increase the system security for the operator network components. The UPF in the 5G core manages the UE network, which is also connected to the Internet in our implementation. For some industry applications with high requirements for security, the UE network is also protected as a local subnet without an internet connection.

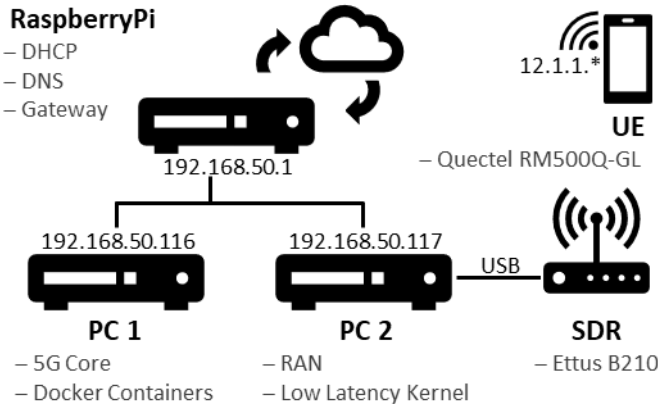


Fig. 2. Structure of the 5G private campus network at the CoSA, Technische Hochschule Lübeck.

The address of our proposed subnet for the 5G system's core and RAN components is 192.168.50.0/24. As described later in Section IV-B the docker deployed core services run in the subnet 192.168.70.128/26, and IP rules are applied to connect the gNBs with the core network. The parts of our proposed 5G SA system (with one of the two gNBs) shown in Figure 2 are further specified in the following Section III.

III. HARDWARE

This section describes the hardware components for our private 5G campus network deployment. Since our deployment

is a functional prototype, low-cost off-the-shelf system components were selected first. To tune the proposed private 5G SA system towards high performance with maximum throughput and ultra-low latency, further adaptations need to be performed, which are not focused and therefore not part of this paper.

Table I lists all components, and Figure 3 shows the main components to deploy our private 5G SA network. The Raspberry Pi serves as a gateway to connect the components to the Internet and provides a DHCP/DNS server. To increase the network's performance for a real 5G SA network, the Raspberry Pi gateway is considered to be replaced by a highperformance router. We selected two desktop computers for this reference design: the 5G core services and the remote radio unit (RRU) that are shown in Figure 3a.

TABLE I. HARDWARE USED TO DEPLOY THE PRIVATE 5G SA NETWORK.

Type/Device	Description
Gateway	
Platform	Raspberry Pi Model B Plus Rev 1.2
CPU	32 Bit ARMv6-compatible processor rev 7 (v6l) (700 MHz)
Memory	430 MB
OS	Raspbian GNU/Linux 10 (buster)
Second Ethernet Interface	Rankie USB Network adapter USB3.0
5G Core (PC 1 in Figure 3a)	
Computer	HP Omen 875-1002 ng
CPU	Intel Core i7 9700k (3.6 GHz)
Memory	16 GB
Storage	1 TB HDD, 256 GB SSD
OS	Ubuntu 18.04.6 LTS (Server)
Kernel	14.15.0-166-generic
5G RRU (PC 2 in Figure 3a)	
Computer	Fujitsu Celsius M7010
CPU	Intel Xeon W-2235 6C (3.8 GHz)
Memory	32 GB
Storage	480 GB SSD
OS	Ubuntu 18.04.6 LTS (Server)
Kernel	5.4.0-96-lowlatency
Software Defined Radio (SDR) in Figure 3b	
Vendor	Ettus Research
Type	USRP B210 (Rev. 4)
Connectivity	USB 3.0
Frequency Range	70 MHz
	70 MHz – 6 GHz
Bandwidth	56 MHz [6]
Ethernet Switch	



a) 5G Core Computer (PC1) and 5G-RRU (PC2)
Fig. 3. Private 5G SA system hardware.



b) USRP B210 SDR



c) Quectel RM500Q-GL

Vendor	Netgear
Type	GS116GE Switch
Ports	16
Max. Datarate	1 GBit

The 5G core services computer is a standard PC with Intel Core i7 CPU and 16 GB RAM. For the RRU, we selected a more powerful PC with a six-core CPU and 32 GB RAM to avoid performance issues with the connected USRP B210 SDR. The USRP B210 SDR builds the physical interface radio interface of our 5G system and is shown in Figure 3b. In Section IV-C we present tests to check the capability of the RRU (PC 2), which is very important for a properly working 5G SA system deployment.

IV. PRIVATE 5G SA DEPLOYMENT

This section presents the deployment for a private 5G SA system. We describe the respective components network procedure, 5G SA core, RRU/gNB, SIM card, and user equipment. The component descriptions are interrelated and include exemplary values of the private 5G SA system deployed at the Technische Hochschule Lübeck. Our 5G SA system is implemented with the open-source project OpenAirInterface. However, the descriptions and components are also applicable to other 5G implementations due to the standardized structure for 5G-SA systems.

A. Network

1) Deployment

Our 5G-SA system components are connected in subnet 192.168.50.0/24 via an off-the-shelf switch. We use a RaspberryPi as the gateway, DHCP, and DNS server. New components, such as additional gNBs, can thus be integrated with ease during the deployment process. In the subnet, fixed IP addresses are assigned in the range 192.168.50.2 - 192.168.50.99 and dynamic addresses in the range 192.168.50.100 - 192.168.50.199. UPF manages the network for the UEs with dynamic addresses in the range 12.1.1.0/25. In our 5G-SA deployment, the RaspberryPi is

also used as a gateway for the UE network to connect to the Internet.

The AMF runs on the 5G core (PC 1), so its IP address must be known in the gNB to connect to the services. The address of the RaspberryPi is part of the IP default gateway configuration via DHCP.

2) Tests

When integrating new components, they automatically receive an IP address configuration in the subnet 192.168.50.0/24 from the DHCP server. The list of leases can be queried in the RaspberryPi with the command `sudo cat /var/lib/misc/dnsmasq.leases` to check the registered components. Connectivity to the Internet can be checked with the `ping www.th-luebeck.de` and `traceroute www.th-luebeck.de` commands.

Before testing UEs connectivity, the PDU session must first be established successfully. The PDU session is confirmed with log outputs in the SMF, including the assigned IP address, as we will show later. For testing, the UPF container can also ping UE devices directly.

B. 5G SA Core Network

1) Deployment

To deploy the service-based architecture of the core network, we followed the tutorial available on the official OpenAirInterface GitLab repository. The components of the core network are deployed as docker containers. Each docker container in the deployment serves one 5G core functionality and is connected to a virtual network in PC 1 within the IP network 192.168.70.128/26. The 5G basic functionality is provided by the services AMF (access and mobility functions), SMF (session management functions), NRF (network repository functions), and UPF (user plane functions). For further details on the core services, we refer to [7] - [11].

To ensure IP packet forwarding between the physical network 192.168.50.0/24 and the virtual network 192.168.70.128/26, we apply IP forwarding with `sudo sysctl net.ipv4.conf.all.forwarding=1` and `sudo iptables -P FORWARD ACCEPT`. For persistent IP packet

forwarding after system reboot, we also add `net.ipv4.ip_forward=1` to the file `/etc/sysctl.conf` and we recommend using the package `iptables-persistent` to save the `iptables` configuration.

After the configuration of the network and proper installation of `docker` and `docker-compose`, we clone the federation of the OpenAirInterface core network 5G repositories at commit hash `8deb4743dc281d6d7d6a67b47afa2cff917381b3`.

In the next step, we build the `docker` images as described in the official tutorial of OpenAirInterface for all the 5G core network services. Therefore, we summarized all build commands in a shell script ¹(located in the `oai-cn5g-fed` directory), that is called with:

```
sudo chmod u+x build-containers.sh ./build-containers.sh
```

After building the images of the OpenAirInterface core network services, we adapt the `docker-compose` configuration for the basic deployment with NRF (`docker-compose-basic-nrf.yaml`) as follows.

```
oai-amf:
environment:
- MCC=262
- SERVED_GUAMI_MCC_0=262
- PLMN_SUPPORT_MCC=262
- SST_0=1
oai-smf:
environment:
- DEFAULT_DNS_IPV4_ADDRESS=192.168.50.1
- DEFAULT_DNS_SEC_IPV4_ADDRESS=192.168.50.1
- NSSAI_SST0=1
- DNN_RANGE0=12.2.1.2 - 12.2.1.128
- DNN_RANGE1=12.1.1.2 - 12.1.1.128
- DNN_RANGE2=12.1.1.129 - 12.1.1.224
- DNN_RANGE3=12.2.1.129 - 12.2.1.224
oai-spgwu:
environment:
- MCC=262
- NSSAI_SST_0=1
```

These modifications are needed to configure the data network name (DNN) and the properties to establish the PDU session for UEs user plane connectivity. We configure the DNN default in the configurations as a packet data network (PDN). The settings for this PDN are set in AMF, SMF, and UPF (SPGW-U) with `SST=1` and `SD=123`. To configure the PDN default in SMF, there are two lists to be changed. The first list (`DNN_LIST`) in the `smf.conf` configures the IP range for DNN default at index 1 and we therefore, set `DNN_RANGE1=12.1.1.2-12.1.1.128`. In the second list (`SESSION_MANAGEMENT_SUBSCRIPTION_LIST`), the DNN default is configured at index 0 and we therefore, set `NSSAI_SST0=1`. A working `docker-compose` configuration file is also provided in our extended deployment documentation¹.

Due to the modification of the PLMN (MCC: 262, mobile country code for Germany, and MNC: 95, not reserved by a mobile operator in Germany), we also adapted the authentication credentials of the UE in the MySQL database table `AuthenticationSubscription`. Therefore, we change the appropriate line in the file `oai_db2.sql` and set the `ueid` and `supi` to `262950000000031`. The corresponding configuration of the SIM card is presented in Section IV-D. The `docker` containers are started and stopped with `docker-compose` commands.

2) Tests

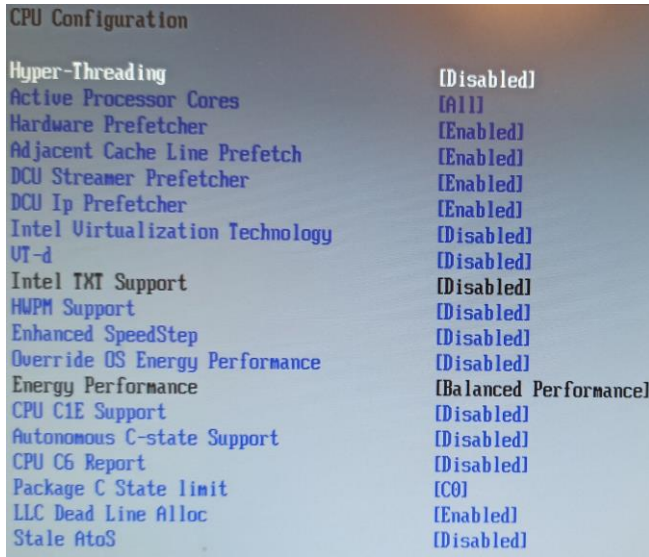
The log outputs of the core services provide information about their interaction and events, such as the dial-in and dial-out of new gNBs and UEs. During the initialization of the services, the parameters defined in the `docker-compose` file for the respective services are output, and we check them for correctness. We also detect errors when starting the services by the error codes and previous log output. When using `docker-compose`, the services are automatically started in the correct order, which must be considered when starting them manually. AMF regularly outputs two tables listing registered gNBs and UEs. If errors occur, these tables are not shown in most cases, and a restart of the 5G-SA core services is required. We also recommend observing the health state of the containers after they were started (`docker ps -a`).

C. RRU/gNB

1) Deployment

This section describes the deployment of a gNB using the OpenAirInterface 5G software. As shown in Table I the computer is set up with Ubuntu 18.04.6. To reach the performance requirements at the RRU computer that are needed with the usage of a software-defined radio as a physical radio interface, we follow the prerequisites of the OpenAirInterface documentation. First, we configure the UEFI to reduce the latency of the CPU by disabling HyperThreading, activating all cores, disabling virtualization technology, disabling VT-d, disabling C1E support, and disabling C-state support. The complete UEFI CPU configuration is shown in Figure 4.

¹ <https://git.mylab.th-luebeck.de/5g/cosa-5g-private-campus-networkdeployment-oai>



CPU Configuration	
Hyper-Threading	[Disabled]
Active Processor Cores	[All]
Hardware Prefetcher	[Enabled]
Adjacent Cache Line Prefetch	[Enabled]
DCU Streamer Prefetcher	[Enabled]
DCU Ip Prefetcher	[Enabled]
Intel Virtualization Technology	[Disabled]
VT-d	[Disabled]
Intel TXT Support	[Disabled]
HWPN Support	[Disabled]
Enhanced SpeedStep	[Disabled]
Override OS Energy Performance	[Disabled]
Energy Performance	[Balanced Performance]
CPU C1E Support	[Disabled]
Autonomous C-state Support	[Disabled]
CPU C6 Report	[Disabled]
Package C State limit	[C0]
LLC Dead Line Alloc	[Enabled]
State AtoS	[Disabled]

Fig. 4. Structure of the 5G private campus network at the CoSA, Technische Hochschule Lübeck.

In the next step, we install the low latency kernel with `sudo apt-get install linux-lowlatency-hwe-18.04`. This kernel is needed to reach the SDR's performance requirements for reliable communication. Once the kernel is installed, we append the line `blacklist intel_powerclamp` to the end of file `/etc/modprobe.d/blacklist.conf`. Then, we install the package `cpufrequtils` with `apt` and set `GOVERNOR="performance"` in the file `/etc/init.d/cpufrequtils`. Also, the file `/etc/default/cpufrequtils` is to be changed or created if it doesn't exist and the line `GOVERNOR="performance"` is appended. In the next step, we disable the `ondemand` daemon with `sudo systemctl disable ondemand` and restart the `cpufrequtils` with `/etc/init.d/cpufrequtils restart`. Finally, we perform the following changes in the boot loader `grub` by editing the file `/etc/default/grub`.

```
GRUB_DEFAULT=saved
GRUB_SAVEDEFAULT=true
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash
intel_pstate=disable processor.max_cstate=1
intel_idle.max_cstate=0 idle=poll"
```

The changes in `grub` are saved with `update-grub`, and the system is rebooted.

Furthermore, we configure a static route to reach the virtual network `192.168.70.128/26` on the core computer with `sudo ip route add 192.168.70.128/26 via 192.168.50.116`. This route is used on the gNB to resolve addresses of the core services, like `192.168.70.130` of the AMF on the core computer with the local IP address `192.168.50.116`.

With the system setup fulfilling the prerequisites, we start the deployment of the OpenAirInterface 5G with the commit hash `142451e741d005a353ebbc9ca47790f0c2811602` and build the software from source as described in the repositories

documentation. We recommend building and running the `softmodem` with the UHD in version 3.15 when using the USRP B210 SDR device with the following commands:

```
source oaienv
cd cmake_targets/
export BUILD_UHD_FROM_SOURCE=True
export UHD_VERSION=3.15.0.0
./build_oai -I -w USRP --gNB
cd cmake_targets/ran_build/build
```

In the future, we will also deploy a gNB with the Ettus N310 and X310 SDRs and provide documentation in our GitLab repository¹. The repository will be permanently updated with experiences corresponding to new deployment setups.

After a successful build of the gNB (`softmodem`), we modify the `config` file `gnb.sa.band78.fr1.106PRB.usrbp210.conf`, that is shipped with the source code. The following changes are applied to run the gNB with the 5G SA core network.

```
tracking_area_code = 0x00a000;
plmn_list = ({
    mcc = 262;
    mnc = 95;
    mnc_length = 2;
    snssaiList = ({ sst = 1; sd = 123;});
});
servingCellConfigCommon = ({
    absoluteFrequencySSB=648000;
    dl_absoluteFrequencyPointA=646728;});
GNB_IPV4_ADDRESS_FOR_NGU="192.168.50.168/24";
```

The MCC, MNC, SST, and SD are configured corresponding to the core settings. Note that these PLMN values shown above are example values, and you need to check the values to be free and not reserved by a mobile operator before running your network. We also configured the carrier frequency to 3.72 GHz to operate in the frequency range, for which we received a license from the German Federal Network Agency (Bundesnetzagentur, assignment number: 06354363).

Finally, we start the gNB with parameters `--sa`, which sets the gNB into standalone mode, and the parameter `-E` is needed for the USRP B210 caused by the reduced sampling rate compared to the N310 SDR. The 5G core services need to be started before the gNB is started.

2) Tests

The selected Linux kernel (low-latency) on the gNB PC is to be checked with `uname -r`, and the CPU speed must constantly be the maximum and the same for all cores and is observed with `watch grep \"cpu MHz\"/proc/cpuinfo`.

After starting the gNB, we observe the connection to the gNB in the AMF docker container on the core computer (PC 1). The AMF periodically shows a gNB table in the log output. Issues for missing connection between the gNB and core are caused by missed IP route at gNB machine or different configuration parameters at gNB and core (such as TAC or PLMN). We test the connectivity between gNB and core with `ping 192.168.70.129` from the gNB.

After initializing the gNB software, it should display periodic status information (e.g., number of bad PUCCH). Error messages, as `received n Bytes expected ...`, indicate performance issues on the USRP interface or gNB machine. From our experience, gNB machines should be operated headless, and configuration, debugging and maintenance via SSH is preferable. Keyboards and especially screens should be avoided for performance reasons. Be aware that gNBs with SDRs have hard real-time requirements.

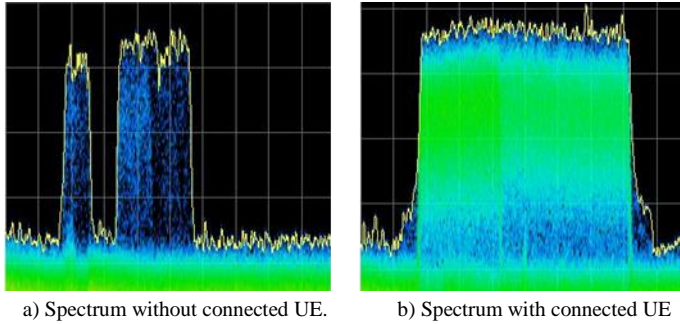


Fig. 5. Spectra recorded when the 5G SA system is running.

We analyzed the physical layer of our 5G SA system for the correct configuration of frequency bands and operation of the gNB. Therefore, we placed antennas of a spectrum analyzer next to the USRP B210 device to perform signal measurements. Figure 5 shows the spectral patterns we observed for one gNB with a Tektronix RSA306 spectrum analyzer. The center frequency of the signals is the expected frequency $f_c = 3.72$ GHz that is configured with `absoluteFrequencySSB` in the gNB.

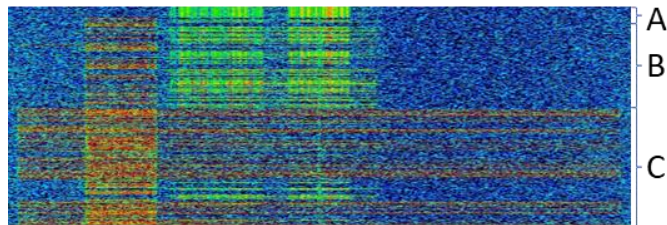


Fig. 6. Waterfall spectrum plot for: (A) no UE connected; (B) UE connecting; (C) UE downloads video stream with interrupt.

The spectrum for the 5G SA system without a connected UE in Figure 5a shows the periodical transmission of the synchronization signal block (SSB) within the configured frequency band (part of the full bandwidth). When UEs stream a large amount of data (in Figure 5b) the rf signal of the gNB covers the whole bandwidth of 40 MHz.

Figure 6 shows the waterfall plot of our 5G SA system for the scenario, first without UE (A), second when a UE connects to the network (B) and streams a video (C). Again, we see the spectrum usage increasing in signal strength and bandwidth with the UE connected and streaming data (indicated by red color).

D. SIM Card Configuration

A suitable SIM card is required to authenticate and register the UE with the 5G core network. The SIM cards must be configurable and support the Mileage confidentiality algorithm the AMF uses in OpenAirInterface core.

After the general deployment of the core network, the UE subscription data entered in the core MySQL database has to be configured onto the SIM card. In the core network branch of the OpenAirInterface, the subscription data of one demo UE is automatically stored in the database. To register additional UEs in the core network, the database may be accessed and edited using the following file `oai_db2.sql`.

To flash a new SIM card, we applied the reader/writer device OYEITIMES MCR3516 with the software GRSIMWRITE v4.1.5. The SIM parameters are obtained from the core configuration and database, where the IMSI (262950000000031) is generated by the PLMN (in our example 26295), followed by an incremental ID (0000000031 in our example). The permanent encryption key (KI) and operator key (OPC) are 32 hex format keys that are configured in the docker-compose configuration in the AMF section and the `oai_db2.sql` file as `insert` into the table `AuthenticationSubscription`. We also set the PLMN codes in the `PLMNwAct`, `OPLMNwAct`, `HPLMNwAct`, and `EHPLMN` to the configured PLMN settings (MCC: 262, MNC: 95, PLMN: 26295).

Fig. 7. Settings applied to program a test SIM card.

Figure 7 shows the settings used to program a SIM card. For a successful authentication of the UE on the AMF and session establishment on the SMF, the KI, OPC, IMSI, and PLMN information must be consistent between SIM card, database, core configuration, and gNB configuration.

E. User Equipment (UE)

Our UE is a Quectel RM500Q-GL, which is recommended as UE in the OpenAirInterface tutorials, mounted on a USB3.0 carrier board connected to a Laptop with Windows or Linux

operating system (OS). Mainly when the modem is run with Windows OS, the installation of the correct driver has to be checked. Suppose the modem is connected and not recognized as a cellular interface on Windows 10 OS. In that case, we recommend uninstalling the modem driver, rebooting the computer, and then reinstalling the driver again. It is essential to use the latest Windows driver, but at least version 2.2.4 or higher. Check the Device Manager in Windows that all modem components are installed with the correct driver.

Some necessary and helpful AT commands are listed below to connect the UE to the 5G SA network with the presented deployment. It is necessary to configure the DNN (formerly APN) default to the modem with:

```
AT+CGDCONT=1,"IP","default"
AT+CGACT=1,1
```

We solved issues when activation of the DNN with `AT+CGACT=1,1` failed by deactivating both the modem and gNB and configuring the modem with deactivated gNB first. Then, we started the gNB again and perform `AT+CFUN=0` and `AT+CFUN=1` to restart the modem. From our experience, the modem connection may also be refused at first connection tries. Then, we observe the UE connected to gNB (shown in the output of the gNB, that UE0 is connected) and disappearing ≈ 30 s after connecting. Setting the modem into flight mode with `AT+CFUN=0` and back into operating mode with `AT+CFUN=1` helps to establish a connection. We also established a connection to the 5G SA network for UE machines with Linux OS, following the instructions to connect a Quectel modem with Linux drivers.

V. EVALUATION

We evaluate the functionality of the 5G system by first checking the UE to Internet connection and, secondly, measuring latency within the system.

First, we check for a successful connection on the modem with the AT commands:

```
AT+C5GREG?
AT+CGPADDR
AT+QPING=1,"www.th-luebeck.de"
```

A PDU session is successfully established when `AT+CGPADDR` respond with an IP address in the estimated UE IP range 12.1.1.2 - 12.1.1.128. The UE has access to the Internet and other devices registered to the UE network with the established PDU session.

Note, make sure that the DNS server configuration in the SMF section in the docker-compose file is correct so that the Internet connection from UE client devices works as expected by using names and not IP addresses only.

For local latency tests, we send ping messages between the oai-ext-dn container on the 5G core machine and the UE. We measured an average round trip time (RTT) of 11 ms, a minimum RTT of 7 ms, and a maximum RTT of 16 ms. The resulting RTT inside our proposed private 5G SA network outperforms the typical RTT in LTE networks with

~ 100 ms [12]. To reach ultra-low latency down to 1 ms, further adaptations of the proposed 5G system are needed in the future. Furthermore, we measured the network performance with iperf3 between the oai-ext-dn container on the 5G core machine (acting as iperf server) and the UE (acting as iperf client). Results are shown in Figure 8.

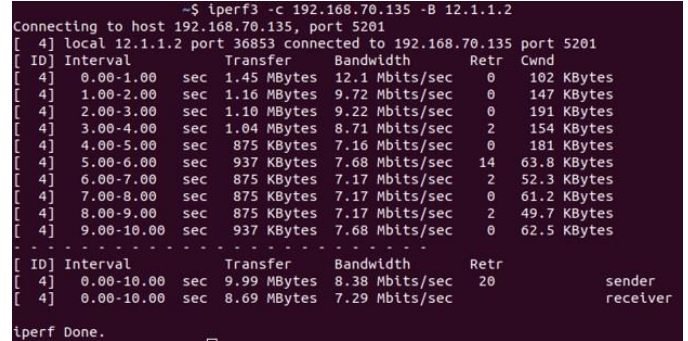


Fig. 8. Recorded performance measurement results with iperf.

The tool iperf measures data rates of 8.38 Mbits/sec for the network, with our basic RF configuration without any optimizations. In a 5G network with a bandwidth of 40 MHz in FR1, we calculated a maximum data rate up to 226.85 Mbit/s according to [9]. The throughput is limited with the USRP B210 devices without amplification and MIMO. We will test performance with USRP N310 or X310, that provide higher bandwidth, with optimized configuration regarding to the modulation and coding scheme in the future. Although the performance is far from being optimal, the performance tests are valuable. They serve as a reference for own deployments. The latency and throughput measurement values are good indicators for correct or comparable implementation.

VI. CONCLUSION AND FUTURE WORK

This work presents a step-by-step private 5G SA network deployment using the open-source software from the OpenAirInterface project. The proposed prototype deployment is minimal and fully functional. We contribute our deployment results to interested 5G operators with just basic network experience, especially with industry and academic backgrounds or system administrators. Our setup consists of low-cost components off-the-shelf and enables a 5G SA deployment with an investment of less than 10k EUR. To share our experience with the community, we provide a repository¹ containing detailed deployment instructions that we will update with new knowledge in the future. The performance evaluation of the proposed private 5G SA network shows our basic results for latency and throughput as a reference for own deployments. Additionally, we will extend the system to four rooms indoor multi-cell 5G testbed in the CoSA 5G-Lab at the Technische Hochschule in Lübeck. Therefore, we will optimize the core components, network structure, and system configuration. We will evaluate additional SDRs, such as Ettus USRP N310 and X310, to increase the system's performance. In the future, we will apply performance modifications to the deployed system

with proposals to change the configuration of the core network and RRU.

ACKNOWLEDGEMENTS

This publication results from the research of the Center of Excellence CoSA at the Technische Hochschule Lübeck. It is funded by the Federal State Schleswig-Holstein, Germany (Id 22021005, Project KI-5G, Project Management Agency: WTSH), Federal Ministry of Transport and Digital Infrastructure of the Federal Republic of Germany (Id 165GU135L, Project 5G-TELK-NF), and Federal Ministry of Transport and Digital Infrastructure of the Federal Republic of Germany (Id 165GU056F, Project BalticFuturePort). Horst Hellbrück is an adjunct professor at the Institute of Telematics of the University of Lübeck.

REFERENCES

- [1] G. Liu, Y. Huang, Z. Chen, L. Liu, Q. Wang, and N. Li, "5g deployment: Standalone vs. non-standalone from the operator perspective," *IEEE Communications Magazine*, vol. 58, no. 11, pp. 83–89, 2020.
- [2] S. Teral, "5g best choice architecture," IHS Markit, 2019.
- [3] M. Agiwal, H. Kwon, S. Park, and H. Jin, "A survey on 4g-5g dual connectivity: Road to 5g implementation," *IEEE Access*, vol. 9, pp. 16193–16210, 2021.
- [4] A. EL RHAYOUR and T. MAZRI, "5g architecture: Deployment scenarios and options," in *2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*. IEEE, 2019, pp. 1–6.
- [5] F. Salah and M. Rinne, "Performance analysis of user plane connectivity in the 5g non-standalone deployment," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.
- [6] "USRP B210 - Datasheet," https://www.ettus.com/wp-content/uploads/2019/01/b200-b210_spec_sheet.pdf, accessed: 2022-01-31.
- [7] S. Ahmadi, *5G NR: Architecture, technology, implementation, and operation of 3GPP new radio standards*. Academic Press, 2019.
- [8] E. Dahlman, S. Parkvall, and J. Skold, *5G NR: The next generation wireless access technology*. Academic Press, 2020.
- [9] M. Kottkamp, A. Pandey, D. Raddino, A. Rosessler, and R. Stuhlfauth, *5G New Radio: Fundamentals, Procedures, Testing Aspects*, 4th ed. Rohde & Schwarz GmbH & Company KG, 2019.
- [10] 3rd Generation Partnership Project, "Release 16 Description; Summary of Rel-16 Work Items," 3GPP, Tech. Rep., 2019.
- [11] "GitLab OpenAirInterface," <https://gitlab.eurecom.fr/oai>, accessed: 2022-01-31.
- [12] Y. Zhu, M. Halpern, and V. J. Reddi, "The role of the cpu in energy-efficient mobile web browsing," *IEEE Micro*, vol. 35, no. 1, pp. 26–33, 2015.
- [13] A. Gabilondo, Z. Fernandez, A. Martín, R. Viola, M. Zorrilla, P. Angueira and J. Montalbán, "5G SA Multi-vendor Network Interoperability Assessment," *2021 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2021, pp. 1–6, doi: 10.1109/BMSB53066.2021.9547167.
- [14] J. Ordóñez-Lucena, J. F. Chavarría, L. M. Contreras and A. Pastor, "The use of 5G Non-Public Networks to support Industry 4.0 scenarios," *2019 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2019, pp. 1-7, doi: 10.1109/CSCN.2019.8931325.
- [15] C. Guimarães et al., "Public and Non-Public Network Integration for 5Growth Industry 4.0 Use Cases," in *IEEE Communications Magazine*, vol. 59, no. 7, pp. 108-114, July 2021, doi: 10.1109/MCOM.001.2000853.