Building a 5G Core Network Testbed: Open-Source Solutions, Lessons Learned, and Research Directions

Linh-An Phan, Dirk Pesch, Utz Roedig, Cormac J. Sreenan
School of Computer Science and Information Technology
University College Cork, Cork, Ireland
lphan@ucc.ie,dirk.pesch@ucc.ie,u.roedig@ucc.ie,cjs@cs.ucc.ie

***Abstract*—Building a 5G Core Network (5GC) testbed is a useful exercise to gain a comprehensive understanding of 5G technology and is an important part of 5GC research. However, this task requires significant engineering effort, making it challenging, especially for novice researchers. In this paper, we share insights from constructing an end-to-end software-based 5G testbed, aiming to assist other researchers in overcoming technical challenges in this endeavour. We begin by providing an overview of popular open-source software designed for 5G testbeds, with a focus on the mobile core network. We then present two distinct 5GC testbed designs: one employing two desktop PCs and the other utilising a single high-performance server. Lessons learned during the testbed installation process are noted. Furthermore, we discuss several research opportunities that can be explored with the testbed, ranging from service development to automation in 5G deployments. We believe this work will foster further exploration and experimentation within the 5GC area.**

***Index Terms*—5G, Core Network, 5GC, testbed, open-source**

I. Introduction

Research related to 5G technology can basically be classified into two broad topics: Radio Access Network (RAN) and 5G Core Network (5GC), with the majority of studies concentrating on the first category. In both areas, testbeds play an important role in facilitating empirical research studies. Unlike the RAN, which requires obtaining a spectrum license and specialised radio hardware, a 5GC testbed can be installed on Commercial Off-The-Shelf (COTS) hardware. This is because components of 5GC (i.e., network functions) are fundamentally just applications that are designed to run on commodity servers [1]. There are two main approaches to deploying a 5GC testbed: 1) purchasing commercial 5GC software [2]–[5] or, 2) using open-source 5G solutions [6]–[9]. Although the former option offers better technical support, it is more expensive and offers less flexibility in terms of modifying the source code and sharing testbed implementations with the community [10]. Therefore, a majority of researchers tend to utilise 5G open-source projects for their research, allowing them to build an end-to-end 5G system using open-source RAN simulators and 5GC software without having expensive radio and end-device equipment. However, selecting and combining these solutions into a comprehensive and functional software-based 5G testbed is a non-trivial task that demands significant engineering effort, such as installing deployment environments and configuring networks [11].

Several industry-scale testbeds [12], [13] have been initiated recently in Europe and America, aiming to facilitate advanced research in mobile networks. Conducting research on these testbeds offers significant benefits; however, this opportunity may not be accessible for many researchers. On the contrary, while building a lab-scale testbed may not be a scientific contribution per se; this task is still worthwhile and offers two valuable benefits. First, it helps researchers to deeply understand the operation of the 5GC software system, allowing them to discover a range of practical problems. Secondly, it provides an empirical environment for testing research hypotheses as well as evaluating proposed solutions, ensuring that the findings and the contribution are accurate, reliable, and reproducible. The evolution of the mobile core network towards 5G has made it into a highly complex system. Therefore, building and running a 5GC testbed is a practical approach to start exploring research problems in 5G.

This paper shares our hands-on experience in the journey of building a software 5GC testbed and is aimed especially at researchers who may be new entrants to this field or have limited computational resources. Our goal is to alleviate the intricacies of building a 5G testing environment and, in doing so, facilitate innovative research related to 5G, especially the 5GC. The contributions in this paper are three-fold:

- We first provide a comprehensive survey of open-source software that can be utilised to build a 5GC testbed. These software packages, namely OpenAirInterface 5G Core Network [6], Open5GS [7], free5GC [8], and Aether [9], are still being actively developed and widely used in the research community. We also cover several open-source RAN simulators and Open RAN solutions.
- We showcase two of our testbed designs, serving as a reference for other researchers to set up their own testbeds. The first testbed uses two desktop PCs, while the second one utilises a single, high-performance server. Moreover, we also share setup guidelines and lessons

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 13/RC/2077_P2.

that we learned during the creation of the testbeds, providing insights from our experience, supplemented by our GitHub repository.
- Lastly, we discuss several research directions that can be undertaken with the testbed. These directions include developing 5G services and applications, designing cloud-native 5G systems, developing high-performance data plane functions, and achieving automation in 5G deployments.

The 5GC is at the heart of a 5G system, facilitating 5G standalone (SA) deployments. The 5G SA allows network operators to unleash the full capabilities of 5G technology, including ultra-low latency, network slicing, and network function exposure. Therefore, promoting in-depth research and development of 5GC is essential for its wide adoption. We believe that insights from our experiences can assist other researchers in overcoming technical challenges inherent in 5GC research.

The structure of this paper is as follows: Section II introduces popular open-source solutions that can be used to build a 5GC testbed. Then, we present the two designs of our testbeds and lessons learned during the process of building these in Section III. Section IV outlines potential research avenues that can be conducted using a 5GC testbed. Finally, we conclude the paper and suggest future work in Section V.

II. Open-Source Solution

A. 5G Core Network

A 5G system can be divided into two sub-systems known as RAN and 5GC, as shown in Fig. 1. The 5GC is designed based on a Service-Based Architecture (SBA), with a Control and User Plane Separation strategy to enable agility and scalability in network deployment and operation. The SBA decouples the 5GC into independent services (i.e., network functions) that implement different functionalities [16]. The 5G Control Plane consists of multiple network functions (NFs), such as the Access and Mobility Management Function (AMF), Session Management Function (SMF), Session Management Function (SMF), etc. In contrast, only one NF operates in the data plane, that is the User Plane Function (UPF). Since 5G core NFs are completely software-based applications, several open-source implementations of 5GC have been developed and are widely used in the research community. The characteristics of these open-source 5GC solutions are summarised in Table I.

1) OpenAirInterface [6]: OpenAirInterface 5G Core Network (OAI 5G CN) is an open-source implementation of the 5G Standalone (SA) Core Network based on 3GPP Release 16 specifications. The OAI 5G CN components have been extensively tested by professional testers using commercial 5G base stations (gNBs) with COTS User Equipment (UE), OAI O-RAN solution, and open-source RAN simulators. The OAI 5G CN currently supports fundamental functions such as UE registration, de-registration, and session management. It also supports features such as mobility, paging, and network slicing. A notable feature of the OAI 5G CN is its capability

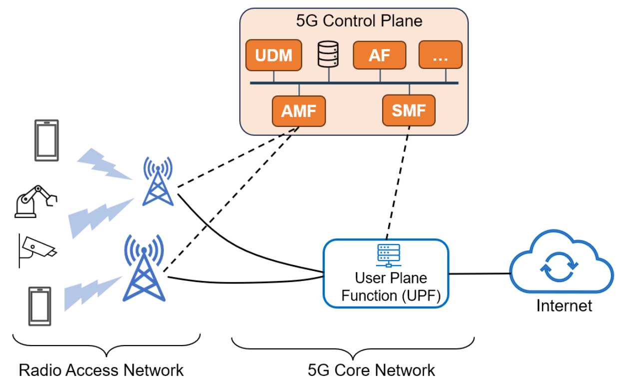


Fig. 1. Overall Architecture of a 5G system.

to manage multiple UEs and PDU sessions concurrently. Regarding deployment options, OAI 5G CN has three deployment options: minimal 5GC (use only AMF, SMF, and UPF), basic 5GC (include authentications and data storage functions on top of the minimal deployment), and slicing 5GC (include all functions of the 5GC). In terms of the UPF, OAI offers a choice between the SPGW-U [14] (developed from 4G with additional features for 5G), VPP-UPF [15] (a high-performance UPF relying on the Vector Packet Processor and Data Plane Development Kit technologies), and production-grade UPF [6]. The OAI 5G CN can be deployed in various environments, such as native installation on bare-metal servers or virtual machines, a Docker-based environment using Docker Compose, and a cloud environment using Helm Chart on Kubernetes (OpenShift cluster). The development of OAI is contributed by members of the OpenAirInterface Software Alliance (OSA). OSA currently has 97 members worldwide, including network operators, companies, universities and research institutes. At the time of writing, the latest version of OAI 5G CN was released in May 2023, accompanied by a roadmap for future releases.

2) Open5GS [7]: Open5GS is an open-source 5GC implementation written in the C language. It comes with a Web application for managing UE subscription information, suitable for testing. The implementation of Open5GS is based on the 3GPP specifications Release 17, with a rich set of supported features such as IPv6 support, the capability to handle multiple PDU sessions, handover functions, and the integration of Voice over LTE and Voice over 5G New Radio. Compared to other 5GC projects, Open5GS benefits from a larger contributor base, resulting in weekly commits to its repository and more frequent software releases. However, Open5GS does not offer official releases of container images for Open5GS's components; thus, users have to build images if they want to deploy them in container-based environments.

3) free5GC [8]: free5GC is another open-source 5GC software developed in Go, adhering to the 3GPP Release 15 specifications. It operates under the Apache 2.0 license, allowing any individual or organization to leverage free5GC for commercial products. One unique network function of

TABLE I
COMPARISON OF OPEN-SOURCE 5GC SOFTWARE.

	OpenAirInterface (OAI) 5G Core Network [6]	**Open5GS** [7]	**free5GC** [8]	**Aether** [9]
3GPP Specification	Release-16	Release-17	Release-15	Release-15
Language	C++	C	Go	Go
Database	MySQL	mongo	mongo	mongo
High-performance UPF	Yes	No	No	Yes
Latest Release	05/2023	10/2023	06/2023	09/2023
Container Image Release	Yes	No	Yes	Yes
Number of Contributors	10+	82	32	20+
Tested RAN	Commercial Open RAN (OAI O-RAN) Simulators (UERANSIM, gnbsim, My-5GRANTester)	Commercial Open RAN (OAI O-RAN) Simulators (UERANSIM)	Commercial Simulators (UERANSIM)	Commercial Open RAN (SD-RAN) Simulators (UERANSIM, gNBSim)
Deployment Environment	Native Application Docker Compose Kubernetes	Native Application Docker Compose Kubernetes	Native Application Docker Compose Kubernetes	Kubernetes
Open-source License	OAI Public License V1.1	GNU AGPL v3.0	Apache 2.0	Apache 2.0

free5GC compared to other open-source 5GC solutions is the non-3GPP Inter-Working Function (N3IWF). N3IWF connects untrusted non-3GPP devices, such as Wi-Fi Access Points and LoRaWAN Gateways, to the 5GC [17]. Free5GC offers container image releases to facilitate its usage and provides a Docker Compose file for automated network function deployment. Additionally, the contributors of free5GC have maintained a list of articles, studies, and tutorials for those new to the platform.

4) Aether [9]: Aether—a project of the Open Networking Foundation (ONF)—is an open-source 5G edge cloud platform aimed at supporting deployments of private 5G networks in enterprise environments. Recently, Aether has been used as the 5G platform for large-scale experimental research projects [9]. The Aether community consists of 44 members, including network operators, companies, and universities. It consists of several components, including SD-RAN (i.e., Open RAN solution), SD-Core (i.e., 5G core network), and an aggregated Aether Management Platform (AMP). The SD-Core project is developed based on Open Mobile Evolved Core and free5GC platforms to support 4G/LTE, 5G non-standalone, and 5G standalone deployments. Aether is highly scalable, spanning from small deployments for research and development purposes to large distributed private networks for Industry 4.0 use cases. Aether provides two automated deployment options: Aether OnRamp for deploying a 5GC on a Kubernetes cluster and Aether-in-a-Box (AiaB) for deploying a 5G system, including a RAN simulator and a 5GC, on a single machine. AiaB provides an automated way to deploy Aether, making it easy to run basic tests and validate the installation of components in the 5GC. However, because Aether employs Kubernetes and Helm as its deployment frameworks, modifying default configurations requires a deep understanding of these complex frameworks.

B. Radio Access Network

The RAN consists of user equipment and base stations. 5G base stations provide wireless communication to UEs and connect them to the 5GC. There are two approaches to setting up a RAN in a 5G testbed: simulation-based and Open RAN-based. The first approach is easy and convenient as it uses simulation software (e.g., UERANSIM [18], my5G-RANTester [20], gnbsim [19], or gNB Simulator [21]) to execute the RAN functions. These RAN simulators usually focus on implementing signalling messages in the control plane and the tunnelling protocol in the data plane, making no difference between simulated and real devices from the 5GC point of view. Open-source RAN simulators are quite suitable for studies focusing on the functionalities of 5GC. Moreover, it is also easy to create a large number of UEs accessing a 5G network for large-scale scenarios. UERANSIM is currently one of the most widely used open-source RAN simulation software.

The Open RAN approach, also known as Software Defined Radio, is to install open-source radio software on Universal Software Radio Peripherals. This approach is preferred to build large-scale, physical 5G testbeds. In the Open RAN architecture, a base station can be split into the Radio Unit (RU), the Distributed Unit (DU), and the Centralized Unit (CU). Open RAN software, such as srsRAN [22], OpenAirInterface [6], and SD-RAN [23], provide implementations of CU and DU that can be deployed on general-purpose computing servers. Open RAN is an important mobile network technology that enables multi-vendor interoperability, intelligence, programmability, and agility in the RAN.

III. TESTBEDS & LESSONS LEARNED

A. Testbed

We chose Aether for the 5GC software and UERANSIM for the RAN simulator to build our testbeds. Aether uses Kubernetes, the de-facto platform for running cloud-native

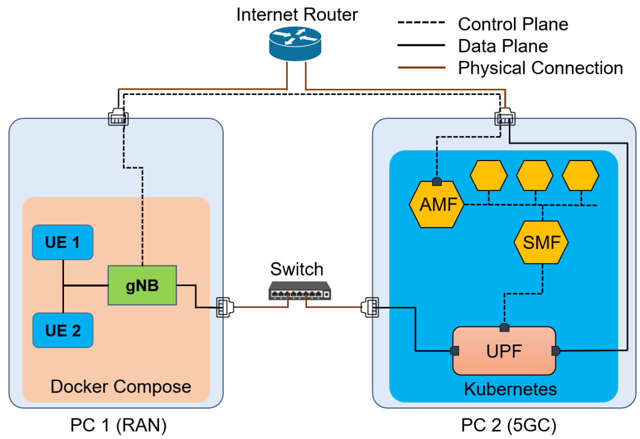


Fig. 2. Configuration of Testbed 1 using two desktop PCs.

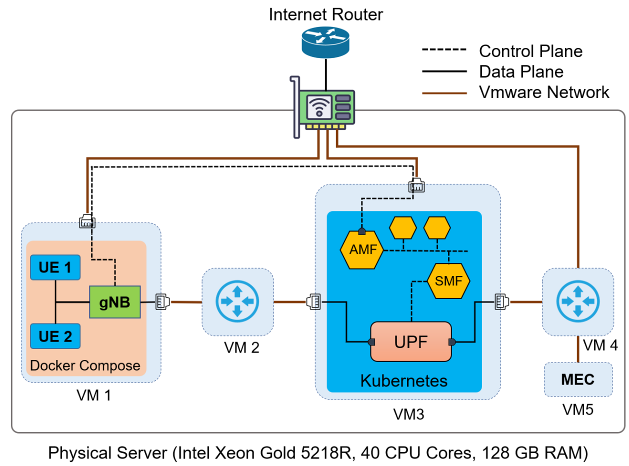
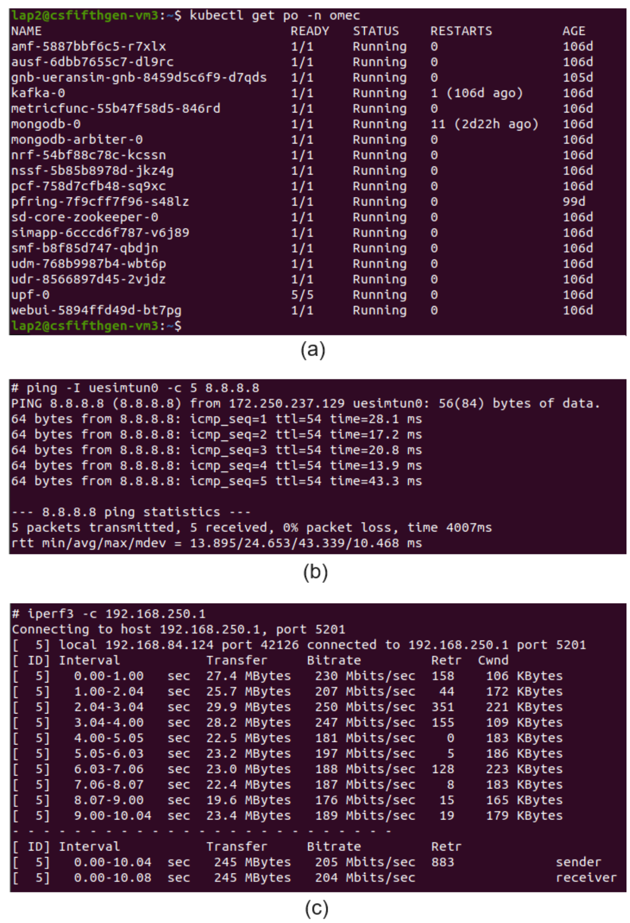


Fig. 3. Configuration of Testbed 2 using five virtual machines on a single powerful server.

systems, as its default deployment environment. Choosing Aether allows us to reuse testbed configurations for future deployments on public cloud services, which can opt for large-scale studies and testing scenarios. For the 5GC, we utilised the AiaB package provided by Aether and applied several modifications to make it operate in a multiple-node environment. As stated, we use UERANSIM to simulate UEs and gNBs. UERANSIM creates a distinctive virtual network interface for each UE, allowing users to transmit data from applications (e.g., ping, iPerf, browser, media streaming) in the 5G data plane under diverse evaluation scenarios. Our setup includes two distinct testbeds, namely Testbed 1 and Testbed 2, which are described below:

1) Testbed 1: Testbed 1 consists of two desktop PCs running Ubuntu 22.04, as illustrated in Fig. 2. The first PC, powered by an Intel Core i3 processor with 8GB RAM, runs UERANSIM within a Docker Compose environment. The second PC is equipped with an Intel Xeon processor and 12 GB of RAM. This PC is used to run the 5GC implementation in the Kubernetes environment. Each PC is furnished with two network interfaces, creating two separate connections between the PCs. One network is configured for internet access and control plane communication between RAN and 5GC, while the other handles data plane communication. As NFs in the 5GC are deployed as a pod in Kubernetes, the communication among NFs is handled by Kubernetes networking. Testbed 1 was initially designed for educational purposes and is used to assist students in understanding 5G systems and 5GC. Despite its setup using modest PCs, the computing capacity of Testbed 1 can be increased by adding more RAN nodes or deploying the 5G control plane and user plane functions in different machines.

2) Testbed 2: Testbed 2 is an enhanced version of Testbed 1, comprising of five virtual machines (VMs) running on a single powerful server, as illustrated in Fig. 3. This testbed is used in our ongoing research to demonstrate the feasibility and advantages of the 5G LAN-type service in industrial applications [24]. The server is equipped with two Intel Xeon processors (40 Core CPU) and 128 GB of RAM. In contrast to Testbed 1, RAN, 5GC, and Multi-access Edge Computing (MEC) servers are deployed in distinct networks (i.e., subnets). Therefore, two virtual routers deployed in VM 2 and VM 4



```
lap2@csfifthgen-vm3:~$ kubectl get po -n omec
NAME                                READY   STATUS    RESTARTS         AGE
amf-5887bbf6c5-r7xlx                1/1     Running   0                106d
ausf-6dbb7655c7-dl9rc               1/1     Running   0                106d
gnb-ueransim-gnb-8459d5c6f9-d7qds   1/1     Running   0                105d
kafka-0                             1/1     Running   1 (106d ago)     106d
metricfunc-55b47f58d5-846rd         1/1     Running   0                106d
mongodb-0                           1/1     Running   11 (2d22h ago)   106d
mongodb-arbiter-0                   1/1     Running   0                106d
nrf-54bf88c78c-kcssn                1/1     Running   0                106d
nssf-5b85b8978d-jkz4g               1/1     Running   0                106d
pcf-758d7cfb48-sq9xc                1/1     Running   0                106d
pfring-7f9cff7f96-s48lz             1/1     Running   0                99d
sd-core-zookeeper-0                 1/1     Running   0                106d
simapp-6cccd6f787-v6j89             1/1     Running   0                106d
smf-b8f85d747-qbdjn                 1/1     Running   0                106d
udm-768b9987b4-wbt6p                1/1     Running   0                106d
udr-8566897d45-2vjdz                1/1     Running   0                106d
upf-0                               5/5     Running   0                106d
webui-5894ffd49d-bt7pg              1/1     Running   0                106d
lap2@csfifthgen-vm3:~$
```

(a)

```
# ping -I uesimtun0 -c 5 8.8.8.8
PING 8.8.8.8 (8.8.8.8) from 172.250.237.129 uesimtun0: 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=54 time=28.1 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=54 time=17.2 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=54 time=20.8 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=54 time=13.9 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=54 time=43.3 ms

--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4007ms
rtt min/avg/max/mdev = 13.895/24.653/43.339/10.468 ms
```

(b)

```
# iperf3 -c 192.168.250.1
Connecting to host 192.168.250.1, port 5201
[  5] local 192.168.84.124 port 42126 connected to 192.168.250.1 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  27.4 MBytes   230 Mbits/sec  158    106 KBytes
[  5]   1.00-2.04   sec  25.7 MBytes   207 Mbits/sec   44    172 KBytes
[  5]   2.04-3.04   sec  29.9 MBytes   250 Mbits/sec  351    221 KBytes
[  5]   3.04-4.00   sec  28.2 MBytes   247 Mbits/sec  155    109 KBytes
[  5]   4.00-5.05   sec  22.5 MBytes   181 Mbits/sec    0    183 KBytes
[  5]   5.05-6.03   sec  23.2 MBytes   197 Mbits/sec    5    186 KBytes
[  5]   6.03-7.06   sec  23.0 MBytes   188 Mbits/sec  128    223 KBytes
[  5]   7.06-8.07   sec  22.4 MBytes   187 Mbits/sec    8    183 KBytes
[  5]   8.07-9.00   sec  19.6 MBytes   176 Mbits/sec   15    165 KBytes
[  5]   9.00-10.04  sec  23.4 MBytes   189 Mbits/sec   19    179 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-10.04  sec   245 MBytes   205 Mbits/sec  883             sender
[  5]   0.00-10.08  sec   245 MBytes   204 Mbits/sec                  receiver
```

(c)

Fig. 4. (a) Network functions of 5GC are deployed in a Kubernetes environment, (b) ping test from UE to a server on the Internet, and (c) throughput test between a UE and a local virtual machine using iPerf3 in Testbed 2.

are used to connect these networks together. Fig. 4(a) shows that network functions of Aether are deployed as pods in a Kubernetes cluster. Meanwhile, Fig. 4(b) and 4(c) show a ping test from a UE to a server on the Internet and a throughput test between a UE and a local VM using the iPerf3 tool in Testbed 2. It should be noted that VMs in Testbed 2 can be deployed on different physical servers. Moreover, adding more VMs to each part of the 5G system will not alter the overall network configuration of the testbed. In other words, the design of Testbed 2 can support both vertical and horizontal scaling. Detailed installation guidelines for both testbeds are available in our GitHub repository [25].

B. Lessons Learned

1) Deployment Platform: As 5G is moving to cloud-native systems, installing a 5G testbed in a container-based environment is needed. Docker Compose and Kubernetes are two popular container orchestration frameworks that can be used for this purpose. Docker Compose is easy to use as it allows deploying the entire system with just a single command using a single configuration file. OpenAirInterface and free5GC are 5GC software implementations that provide a Docker Compose file to install their components quickly. However, Docker Compose is designed to run multiple containers on a single machine (i.e., single-host deployment), which limits the scalability of the system. On the contrary, Kubernetes can deploy and manage containers across multiple nodes (i.e., cluster deployment). Moreover, Kubernetes supports advanced capabilities such as automatic scaling, load balancing, self-healing, and resource management. As a result, Kubernetes is complex and time-consuming to use effectively. It is worth noting that Docker Compose and Kubernetes can be used together in certain environments. For example, in our testbed, we use Docker Compose to simplify the deployment of the RAN while utilising Kubernetes to enable scalability in the deployment of the 5GC.

2) Network Configuration: Understanding the network of a 5G system and how it aligns with the network model of container platforms is crucial to building a 5G testbed. Basically, communications in a 5G testbed can be divided into control plane and data plane paths. It is recommended to use different network interfaces (or even different physical networks) for each plane to avoid conflicting configurations. Moreover, since we do not deploy the RAN in the Kubernetes cluster, the communication between RAN and 5GC (or between Internet and UPF) is between the outside of the Kubernetes cluster and components inside the cluster. This differs from the communication among network functions in the 5GC, which happens inside the Kubernetes cluster. Some NFs, such as AMF, SMF, and UPF, require multiple network interfaces for their operations. Moreover, most open-source 5GC implementations currently provide a single-machine deployment based on a localhost network, which often does not require routing configuration. As a result, these sample deployments may not operate accurately in multiple-node or cloud environments, which require additional network configurations.

IV. Research Directions

A. Cloud-Native 5G System

Using a cloud-native architecture (i.e., SBA) is the major difference between the 5G and 4G core networks [26]. Although this software architecture offers multiple benefits, designing a 5GC cloud-native system is challenging. This task involves refactoring network functions to independent services using container-based technology [27]. In a cloud-native 5G system, the failure of an individual service should not disrupt the operation of other services and cause a system-wide failure. Moreover, adhering to the 3GPP standards for cloud-native design ensures cross-platform compatibility, enabling multi-vendor developments of NFs. In other words, agility, scalability, and dependability are key characteristics of a cloud-native system. However, existing open-source 5GC solutions have not entirely met these cloud-native prerequisites, making the design and development of cloud-native 5G systems an ongoing research question. Employing a 5G testbed enables researchers to implement and validate their solutions regarding this research question.

B. 5G Services & Applications

The advance of 5G technologies has led to the emergence of new services and applications. Many of them are developed on top of the 5GC, such as network slicing [28], 5G LAN-type service [29], edge computing, and traffic steering [30]. The functionalities of fundamental services, such as device authentication and management, mobility management, and session management, are also implemented in the 5GC. A 5G testbed allows researchers to verify their enhancements for existing services or demonstrate the benefits of novel applications in a 5G environment. Another important research topic in 5G is security, which can be conducted based on the actual operation of 5G components in a confined and controlled testbed [31]. Lastly, the 5G testbed allows researchers to generate a large amount of operating data from devices in 5G networks, which can be used as realistic datasets for machine learning studies.

C. High Performance Data Plane

The network performance of 5G requires a UPF to have high throughput and low latency in packet processing. Data plane acceleration solutions have been investigated in the last decade. Several technologies, such as Data Plane Development Kit (DPDK), Vector Packet Processing (VPP) [15], Express Data Path (XDP) [32], and P4 [33], have been utilised to develop high-performance UPFs. However, these technologies require specific hardware and configurations, which are difficult to deploy in a container-based environment [34]. Therefore, supporting high-performance, cloud-native UPF in existing 5GC solutions is crucial for the practical application of open-source 5GC software.

D. Automation

The complexity of 5G systems requires automated deployment solutions to minimise human errors and reduce installation time. Additionally, the geographical distribution and diverse network demands highlight the need for automated deployment of the 5GC [35]. For example, UPFs and application servers can be deployed at the network edge in ultra-low latency communication systems, while control plane functions reside on centralised cloud servers. Providing automated deployment solutions further enables researchers to quickly set up new testbeds and replicate previous research results, facilitating more studies in the 5G domain.

V. Conclusion & Future Work

The 5G Core Network (5GC) is integral to 5G Standalone deployments, allowing network operators to harness the full potential of 5G technology. In this paper, we provided a review of available open-source packages and shared our experience building software-based 5GC testbeds that can be deployed on commodity servers. Notably, these testbeds do not require expensive hardware, making them ideal for researching, developing, and testing with 5GC functionalities. It is important to emphasise that the scale of the testbed can be tailored to adapt to various research scenarios. In future work, we plan to deploy a 5GC testbed on cloud services, actualising a truly cloud-native 5GC deployment. Such an integration would enhance the scalability and flexibility of the testbed and broaden its accessibility to the research community. We believe that our insights and efforts will facilitate further 5GC research and innovations.

Acknowledgment

Testbed 1 was mainly installed by Alban Guilhot (University of Clermont Auvergne, France) under the author's supervision during his internship at University College Cork.

References

[1] W. d. S. Coelho, A. Benhamiche, N. Perrot and S. Secci, "Network Function Mapping: From 3G Entities to 5G Service-Based Functions Decomposition," *IEEE Communications Standards Magazine*, vol. 4, no. 3, pp. 46-52, September 2020.

[2] Amantya, "5G Lab: End-to-end Solution for R&D, Testing, and Use Case Development," https://www.amantyatech.com/5g/5g-use-case-lab-setup (accessed Oct. 23, 2023).

[3] Athonet, "Athonet 5G Core," https://athonet.com/products/athonet-5g-core (accessed Oct. 23, 2023).

[4] Affirmed, "Affirmed UnityCloud—5G Core," https://www.affirmednetworks.com (accessed Oct. 23, 2023).

[5] L. R. Salvador and Z. Rajnai, "5G Vendors, cybersecurity and NESAS," 2023 IEEE 17th International Symposium on Applied Computational Intelligence and Informatics (SACI), Timisoara, Romania, 2023.

[6] OpenAirInterface Software Alliance, "OpenAirInterface," https://openairinterface.org (accessed Oct. 23, 2023).

[7] Sukchan Lee, "Open5GS," https://open5gs.org (accessed Oct. 23, 2023).

[8] free5GC, "free5GC–Open source 5G core network base on 3GPP R15," https://free5gc.org (accessed Oct. 23, 2023).

[9] Open Networking Foundation, "Aether: Private 5G Platform," https://opennetworking.org/aether (accessed Oct. 23, 2023).

[10] D. Pineda, R. Harrilal-Parchment, K. Akkaya, A. Ibrahim and A. Perez-Pons, "Design and Analysis of an Open-Source SDN-based 5G Standalone Testbed," IEEE INFOCOM 2023 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Hoboken, NJ, USA, 2023.

[11] D. Scotece, A. Noor, L. Foschini and A. Corradi, "5G-Kube: Complex Telco Core Infrastructure Deployment Made Low-Cost," *IEEE Communications Magazine*, vol. 61, no. 7, pp. 26–30, July 2023.

[12] S. Fdida *et al.*, "SLICES, a scientific instrument for the networking community," *Computer Communications*, vol. 193, pp. 189–203, September 2022.

[13] T. Chen *et al.*, "A Software-Defined Programmable Testbed for Beyond 5G Optical-Wireless Experimentation at City-Scale," *IEEE Network*, vol. 36, no. 2, pp. 90–99, March/April 2022.

[14] OpenAirInterface Software Alliance, "SPGW-U: An implementation of the Evolved Packet Core network," https://github.com/OPENAIRINTERFACE/openair-spgwu-tiny (accessed Oct. 23, 2023).

[15] travelping, "User Plane Gateway (UPG) based on VPP," https://github.com/travelping/upg-vpp (accessed Oct. 23, 2023).

[16] 3GPP TS 23.501, "System architecture for the 5G System (5GS)," v18.2.2, Jul. 2023.

[17] M. T. Lemes, A. M. Alberti, C. B. Both, A. C. De Oliveira Júnior and K. V. Cardoso, "A Tutorial on Trusted and Untrusted Non-3GPP Accesses in 5G Systems—First Steps Toward a Unified Communications Infrastructure," *IEEE Access*, vol. 10, pp. 116662–116685, Nov. 2022.

[18] UERANSIM, "UERANSIM GitHub Repository," https://github.com/aligungr/UERANSIM (accessed Oct. 23, 2023).

[19] Rohan Kharade, "gnbsim: a 5G SA gNB/UE Simulator for Testing 5G System," https://gitlab.eurecom.fr/kharade/gnbsim (accessed Oct. 23, 2023).

[20] Lucas B.D. Silveira *et al.*, "Tutorial on communication between access networks and the 5G core," *Computer Networks*, vol. 216, pp. 109301, October 2022.

[21] Open Networking Foundation, "gNB Simulator GitHub Repository," https://github.com/omec-project/gnbsim (accessed Oct. 23, 2023).

[22] Software Radio Systems, "srsRAN Project: Open Source RAN," https://www.srsran.com (accessed Oct. 23, 2023).

[23] Open Networking Foundation, "SD-RAN Project," https://opennetworking.org/open-ran/ (accessed Oct. 23, 2023).

[24] L.-A. Phan, D. Pesch, U. Roedig and C. J. Sreenan, "Demonstration of LAN-type Communication for an Industrial 5G Network," In Proc. of IEEE Conference on Computer Communications (INFOCOM), Hoboken, NJ, USA, 2023.

[25] Linh-An Phan, "5G LAN Testbed GitHub Repository," https://github.com/linhanphan/5glan (accessed Oct. 23, 2023).

[26] L. Peterson, O. Sunay, and B. Davie, "Private 5G: A Systems Approach," https://5g.systemsapproach.org (accessed Oct. 23, 2023).

[27] A. Khichane, I. Fajjari, N. Aitsaadi and M. Gueroui, "Cloud Native 5G: an Efficient Orchestration of Cloud Native 5G System," NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, pp. 1–9, June 2022.

[28] S. Zhang, "An Overview of Network Slicing for 5G," *IEEE Wireless Communications*, vol. 26, no. 3, pp. 111–117, June 2019.

[29] X. Zhu and Y. Chi, "5G Release 16 Key Technologies for Vertical Industries Application," *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 1269–1274, Aug. 2021.

[30] A. K. Salkintzis, M. Kühlewind, S. Rommer and R. Liebhart, "Multi-path QUIC for Access Traffic Steering Switching and Splitting in 5G Advanced," *IEEE Communications Standards Magazine*, vol. 7, no. 1, pp. 48–56, March 2023.

[31] M. Repetto, "Service Templates to Emulate Network Attacks in Cloud-Native 5G Infrastructures," 2023 IEEE 9th International Conference on Network Softwarization (NetSoft), Madrid, Spain, 2023.

[32] T. A. N. do Amaral, R. V. Rosa, D. F. C. Moura and C. E. Rothenberg, "An In-Kernel Solution Based on XDP for 5G UPF: Design, Prototype and Performance Evaluation," 2021 17th International Conference on Network and Service Management (CNSM), Izmir, Turkey, 2021.

[33] R. MacDavid *et al.*, "A P4-based 5G User Plane Function," In Proc. of the ACM SIGCOMM Symposium on SDN Research (SOSR), USA, 2021.

[34] S. Sekigawa, C. Sasaki and A. Tagami, "Toward a Cloud-Native Telecom Infrastructure: Analysis and Evaluations of Kubernetes Networking," 2022 IEEE Globecom Workshops (GC Wkshps), Rio de Janeiro, Brazil, pp. 838–843, Dec. 2022

[35] A. Santos, J. Bernardino and N. Correia, "Automated Application Deployment on Multi-Access Edge Computing: A Survey," *IEEE Access*, vol. 11, pp. 89393–89408, 2023.