Cloud-native 5G experimental platform with over-the-air transmissions and end-to-end monitoring

Sergio Barrachina-Muñoz, Miquel Payaró, and Josep Mangues-Bafalluy
Services as Networks (SaS)
Centre Tecnològic Telecomunicacions Catalunya (CTTC/CERCA)
Barcelona, Spain
{sergio.barrachina, miquel.payaro, josep.mangues}@cttc.cat

***Abstract*—5G represents a revolutionary shift with respect to previous generations given its design centered on network softwarization. Within such a change of paradigm, cloud-native solutions are widely regarded as the future of vertical application development because of their enhanced flexibility and adaptability to complex and dynamic scenarios. In this context, we present an experimental framework with over-the-air transmissions that tackles two critical aspects for enhancing the lifecycle management of 5G and beyond networks: cloud-native deployments of 5G core network functions (NFs) and end-to-end monitoring. First, we deploy Open5GS and Prometheus-based monitoring as containerized network functions (CNFs) in a Kubernetes cluster spanning a multi-tier network with a multi-access edge computing (MEC) host. We then demonstrate the end-to-end monitoring system by showcasing via Grafana dashboards both infrastructure resources and radio metrics of two scenarios; one devoted to user plane function (UPF) re-selection and the other to user mobility.**

***Index Terms*—5G, cloud-native, Open5GS, Kubernetes, monitoring, MEC, experimental platform**

I. Introduction

5G and beyond (B5G) networks are set to address the demands of a fully connected and mobile society, enabling a wide variety of services and applications over the same infrastructure. Further, 5G adopts edge computing as a key paradigm evolving from centralized architectures towards multiple points-of-presence (PoPs) of edge nodes. In turn, this enables multi-access edge computing (MEC) applications for low-latency and high bandwidth like virtual and augmented reality (VR/AR). In this context, projects like MARSAL [1], [2] propose a new paradigm of elastic virtual infrastructures that integrate transparently a variety of novel radio access, networking, management, and security technologies to deliver end-to-end transfer, processing, and storage services in an efficient and secured way.

To materialize the MARSAL vision in 5G and B5G networks, two elements are key: cloud-native deployments within the network function virtualization (NFV) paradigm and end-to-end monitoring. As for the former, cloud-native infrastructures make it possible to share infrastructure resources, enabling their dynamic allocation to meet the service level agreements (SLAs) of existing and future demanding use cases. Cloud-native technologies also reduce time to market, respond sooner to customer demands, and facilitate the lifecycle management and automation of the network. Therefore, they are widely regarded as the future of vertical application development with enhanced flexibility, scalability, and reduced cost.

Monitoring is the second key aspect this paper deals with, as it is critical to manage such complex cloud-native infrastructures and to increase operational efficiency. Indeed, a paramount factor for 5G is end-to-end real-time monitoring, gathering infrastructure metrics (i.e., compute, storage, and network) as well as domain-specific metrics of components such as gNBs or MEC services. Gathering these metrics supports the lifecycle management of services running over the 5G network and favors intelligent reconfiguration and alerting to involved tenants and stakeholders, spanning from infrastructure owners, operators, slice owners, or service/application developers. Certainly, multi-tenant networks must support network slices, which require monitoring key performance indicators (KPIs), commonly belonging to different technological domains and managed by different entities. For instance, a network operator shall focus on the KPIs of the components running the network slice (e.g., RAN, cloud/edge, routers), while the slice owner may consider high-level KPIs (e.g., end-to-end delay) for SLA validation [3]. In both cases, monitoring turns into an imperative aspect.

Even though there are valuable works in the literature on cellular networks devoted to cloud-native deployments and monitoring, only a few of them (partially) treat both aspects together. In contrast, this paper deals with both aspects by presenting a cloud-native experimental platform for MEC-enabled 5G networks endowed with an end-to-end monitoring system. The platform is easily deployable via Helm charts.[1] So, the main contributions are as follows:

- Cloud-native 5G core deployment with Open5GS in a multi-PoP Kubernetes cluster, where each NF runs as a separate containerized NF (CNF).
- End-to-end containerized monitoring gathering both infrastructure and radio/RAN metrics via CNFs.
- Integration of a commercial gNB (Amarisoft Callbox) into the 5G testbed and development of a custom sampling function for pulling gNB metrics (e.g., downlink/uplink bitrate).
- Showcasing of the monitoring system through the visualization of different metrics in Grafana dashboards. Two toy scenarios are considered; one for UPF re-selection and the other for UE mobility.

The rest of the article is structured as follows. Section §II discusses the related work, section §III describes the main components of the experimental platform and the MEC-

[1]All of the source code of our experimental platform [4] is open.

enabled testbed, section §IV depicts the end-to-end monitoring system, and section §V showcases and validates the whole framework. Finally, we our conclusions and future work is collected in section §VI.

II. Related work

There are several works in the literature dealing either with cloud-native 5G deployments or end-to-end monitoring systems for 5G networks. However, none of them fully addresses both aspects together as we discuss next.

As for cloud-native deployments, a method to orchestrate and manage a container-based C-RAN using Kubernetes and OpenAirInterface (OAI) is presented in [5]. Authors in [6] introduce an open-source infrastructure for 5G RAN development where DevOps simplifies the deployment of end-to-end applications to the edge. Also, Kube5G is proposed in [7] for building and packaging a cloud-native telco NF through nested layers, and a 5G cloud-native environment based on Kubernetes and Openshift Operator is introduced in [8]. Recently, an integration of KubeFed for deploying workloads in multiple clusters and Network Service Mesh for providing connectivity across cluster boundaries has been proposed in [9]. The aforementioned works make valuable efforts towards cloud-native deployments. However, none of them realizes a full 5G core, but different variations of 4G's evolved packet core (EPC). Further, they do not focus on monitoring.

As for papers dealing with monitoring, authors in [10] present SONATA, a multi-PoP monitoring framework of NFV services involving both containers and virtual machines. However, monitoring is discussed from an architectural perspective, and no actual 5G core is deployed. Instead, authors in [11], [12] present an approach to deliver monitoring and telemetry mechanisms as a service using Prometheus and Netdata over an Open5GS network, but no containerization is provided. Similarly, a monitoring framework introducing metrics collectors deployed per network slice using Prometheus is devised in [3], but no containerized deployment of the 5G core is provided either. In [13], authors investigate the effect of inter-NF dependencies in terms of resource consumption in a Free5GC network deployed in Kubernetes. However, only limited monitoring (e.g., RAN parameters are not considered) is undertaken through custom Python scripts. Finally, [14] introduces the 5GROWTH service platform with an AI-driven automated 5G end-to-end slicing. However, no 5G core is deployed.

Unlike the previous works, the framework proposed in this paper jointly provides full cloud-native deployment of both 5G core and end-to-end monitoring (including RAN) through CNFs orchestrated via Kubernetes. Besides, the framework is visually demonstrated through two scenarios reflecting a series of events in the context of UPF re-selection and user mobility.

III. Cloud-native 5G experimental platform

The 5G architecture consists of two parts that have remarkably changed from previous generations: the new radio access network (NG-RAN) supporting the new radio (NR) and the 5G Core Network (5GC). In this section, we describe the main components of our experimental platform and propose a MEC-enabled testbed to demonstrate the end-to-end monitoring system.

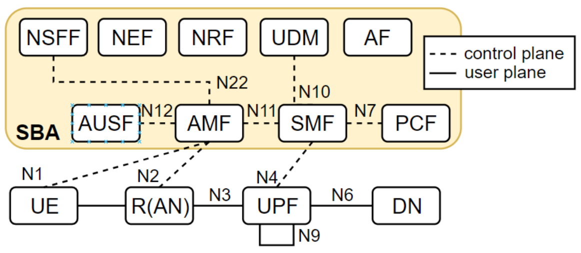

Fig. 1: 5G System Architecture in reference point representation (based on [15]). Some NFs and interfaces are not included for the sake of representation.

A. *Open-source 5G core*

The movement toward softwarization of telecommunication networks has deeply influenced the creation of the 5G core (5GC) [16]. Rather than relying on monolithic elements, 5G adopts a service-based architecture (SBA) composed of NFs that modularize the tasks of the core. As shown in Fig. 1, these NFs interact through Service-Based Interfaces (SBIs), which employ Representational State Transfer (REST) interfaces. A key feature of such SBA modularization is network slicing, which benefits from softwarization and cloudification. In essence, slices represent logical instances of the network that can be tailored to optimize services and thus cope with different service level agreements (SLAs) according to the use case. Further, within the SBA, we may find the NWDAF (Network Data Analytics Function) and the MDAF (Management Data Analytics Function) for generating insights from NFs data and taking actions to enhance performance, including slice selection and control [17].

This new architecture allows 5G stakeholders much more flexibility and openness, paving the way for an environment where open-source perfectly suits. In this regard, some alternatives of open-source 5GC are available, such as OpenAirInterface [18] CN, Free5GC [19], and Open5GS [20]. In our experimental framework, we use Open5GS v2.4.0, since it includes most of the 5GC NFs defined in 3GPP and also allows deploying more than one UPF instance, thus supporting MEC-enabled networks. Nevertheless, our experimental framework is designed to also work with other open-source 5G cores under acceptable adjustments.[2]

B. *Cloud-native deployment of the 5G core*

5G is expected to support use cases that go beyond raw throughput performance, where the focus is to be put on service flexibility and agility. In this regard, the procedure to deploy 5G NFs has a critical impact. In essence, these functions can be instantiated as physical NFs (PNFs), virtual NFs (VNFs), or containerized NFs (CNFs). Naturally, VNFs gained momentum against siloed PNFs since the conceptualization of the modularized 5G SBA because of the virtualization benefits in terms of efficiency, scalability, or cost. Recently,

[2]Preliminary deployments of our containerized framework with OpenAirInterface CN and Free5GC have been already successfully validated.

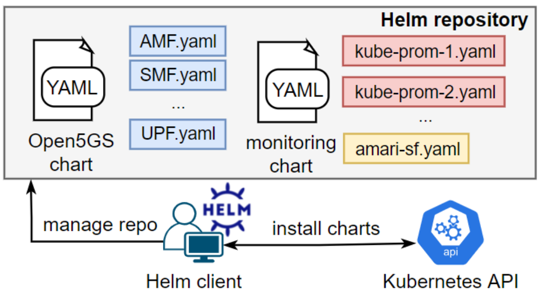

Fig. 2: Helm and Kubernetes flowchart. The Open5GS chart is composed of the NF templates in blue (e.g., AMF), whereas the monitoring chart includes templates for kube-prometheus in red and the Amarisoft sampling function in yellow.

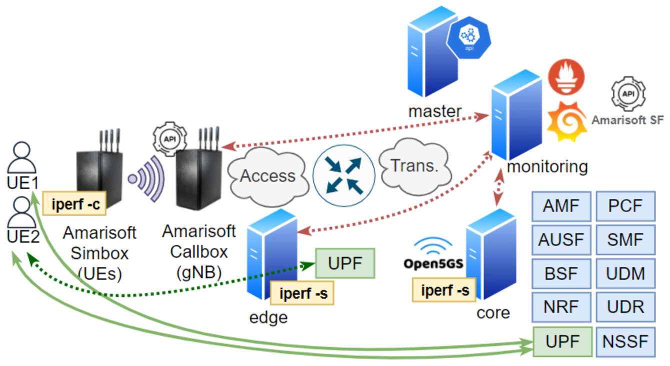

Fig. 3: MEC-enabled 5G testbed. Data planes are shown in green (dashed arrow for the MEC data plane). Monitoring connections are represented with red dashed arrows.

it is the turn of Containerized Network Function (CNFs) to gain momentum among operators [21] against conventional Virtualized Network Function (VNFs) due to their higher degree of scalability, efficiency for operation and management, energy-saving, and suitability for resource-constrained edge applications.

According to Docker [22], a container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another. This makes a container image a lightweight, standalone, executable package of software that includes everything needed to run an application. Container deployments, which may span multiple hosts, are managed with an orchestrator responsible for automating container creation, deletion, and modification without service disruption. Notice that such tasks on containers match the NFV lifecycle management. In this work, we adopt Kubernetes [23] as a container orchestrator, since it is the de facto solution in multiple industries for high-demand services with complex configurations.

As explained in the following subsections, we deploy both the 5GC NFs and monitoring system within the same Kubernetes cluster using Helm charts [24], a collection of files that describe a related set of Kubernetes resources (see Fig. 2).[3] This way, the whole framework is deployed in just two commands, one for installing the monitoring system and the other for installing Open5GS. Once the deployments are instantiated, connectivity among containers and toward external services must be provided. In particular, our Kubernetes cluster relies on Calico [25], a well-known container network interfaces (CNIs) plugin to implement such networking capabilities.

C. Experimental RAN integration

The radio access network (RAN) is another critical component of 5G networks, since it provides individual users with wireless connectivity to the core and external data networks. There are different alternatives when it comes to experimenting with 5G RAN, which can be classified in simulated/emulated (e.g, UERANSIM [26]) and physical/real (e.g., Amarisoft). In this work, in order to provide actual over-the-air-transmissions, we rely on Amarisoft's AMARI Callbox Ultimate [27] acting as a gNB with high-performing NR capabilities. Nevertheless, other RAN alternatives like UERANSIM have been also successfully integrated within the testbed without requiring any configuration changes in the Open5GS Helm chart.

Notice that the RAN in our framework is essentially a physical NF (PNF), whereas the 5GC and monitoring system are built on CNFs. As for how to integrate the AMARI gNB with the Open5GS core, we shall indicate the AMF's IP address in the gNB configuration file to establish the NG Application Protocol (NGAP) connection. In our case, since the AMF runs as a CNF, a custom Kubernetes service exposes the AMF functionality to let the gNB point to the master node IP rather than to the AMF's pod IP. This is a common practice, since pod IPs may change after deletion, while services remain fixed.

D. A MEC-enabled 5G testbed

Testbeds are essential in telco research, as new architectures, techniques, and features can be conveniently assessed and validated in the lab before going into field trial campaigns. In this work, we implement and integrate the testbed shown in Fig 3 to showcase our cloud-native end-to-end 5G experimental platform. In particular, the 5G network is composed (from left to right) of the following elements: two UEs emulated with Amarisoft AMARI UE Simbox Series [28], with UE1 always targeting best-effort services while UE2 may target both best-effort and time-critical MEC applications (e.g., AR/VR); an Amarisoft Callbox acting as a stand-alone 5G gNB; an edge node running the MEC UPF and an iperf [29] server; a core node running the Open5GS CNFs and another iperf server; a monitoring node hosting the monitoring containers; and a master node managing the Kubernetes cluster. In the current testbed implementation, since the focus is on the monitoring of the RAN and core domain components, the access and transport networks have been simplified to Ethernet links in a local area network. As shown in the Kubernetes representation in Fig. 4, the cluster is composed of the aforementioned nodes (master, core, edge, and monitoring). Therefore, the master is responsible for deploying, controlling, deleting, and updating the containers of each of the nodes.

[3]Notice that all the NFs in Open5GS can be compiled and deployed separately, making it a suitable candidate for evaluating the performance of distributed and cloud-native deployments of the 5GC.

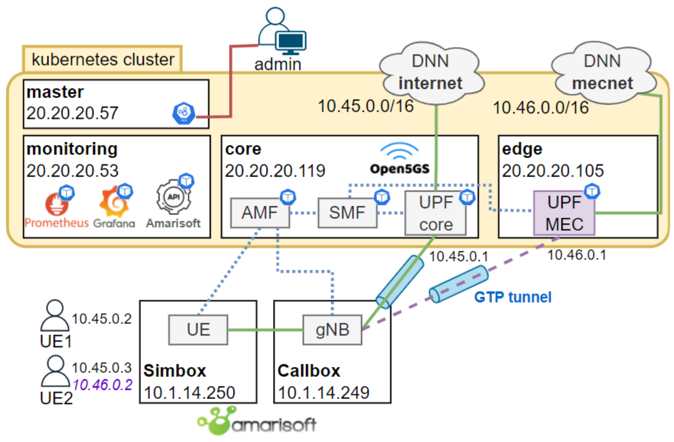

Fig. 4: Kubernetes deployment. Notice that UE2 gets an IP address depending on its assigned UPF, i.e., 10.45.0.3 and 10.46.0.2 for the core UPF and MEC UPF, respectively.

Remarkably, all the 5GC NFs are deployed in one click (through Helm) in the core and edge host. In particular, the core runs all the Open5GS NFs available in v2.4.0, including both critical (e.g., AMF or SMF) and secondary NFs (BSF). Besides, for the sake of enabling MEC platforms, the presented cloud-native deployment provides two UPFs: one located at the core and the other, at the edge node. This way, each UPF serves as a PDU session anchor and provides a connection point to different access networks, one being the conventional Internet, and the other any data network that can benefit from MEC processing capabilities, like AR/VR application components.

Finally, the monitoring node is in charge of running all the containers related to the end-to-end monitoring of the components. To that aim, and as explained in §IV, kube-prometheus [30] is deployed for monitoring Kubernetes elements, while a custom sampling function is developed to pull metrics from the Amarisoft Callbox API. So, these monitoring CNFs can be viewed as 5G's application functions (AFs) within 5G's SBA.

IV. End-to-end monitoring: from core to RAN

We depict below the monitoring system designed for our 5G experimental framework. We shall emphasize two main characteristics that make it a valuable asset: it is cloud-native, meaning that specific CNFs are deployed for monitoring purposes, and it is end-to-end, meaning that deployed CNFs pull metrics both from the core and the RAN domains. The data is gathered into a centralized database where metrics are then plotted in dashboards.

A. *Monitoring with Prometheus*

Monitoring is the practice of examining behavioral data from infrastructure, network events, and user interactions. Thus, in order to let network administrators gather metrics of interest and manage unexpected events, a proper monitoring system must be able to collect data from multiple sources.

In this work, we primarily rely on Prometheus [31], since it is an open-source monitoring and alerting toolkit that can be easily integrated with Kubernetes to support automatic deployment, letting agents be automatically discovered via service discovery. The Prometheus server also opens interfaces to third-party applications, like web UI or Grafana. We refer the reader to [32] for a comparison on monitoring tools focused on 5G networks. In particular, we use Prometheus in two different approaches to tackle infrastructure and RAN monitoring. For the former, we use kube-prometheus [30] and rely on a custom sampling function for the latter.

1) Kube-prometheus for infrastructure: To monitor the usage of infrastructure resources, such as compute, storage, and network, we rely on kube-prometeus stack, an easy to operate end-to-end Kubernetes cluster monitoring stack that uses Prometheus Operator. The stack is pre-configured to collect metrics from all Kubernetes components – meaning resources are measured at different levels, such as pod, workspace, Kubernetes node, or host – and it also delivers a default set of dashboards and alerting rules. Therefore, we can deploy a full off-the-shelf cluster monitoring tool with a single Helm command. Among the metrics gathered with kube-prometheus, we find CPU, memory, transmit/receive networking, etc. Some of them are presented at node level in §V-A.

2) Custom sampling function for RAN: For our monitoring purposes, the Amarisoft Callbox can be accessed through a remote API using the WebSocket protocol, which establishes a persistent connection between the client (Amarisoft sampling CNF in monitoring node) and the server (Amarisoft Callbox itself). This API exposes different metrics at gNB/radio level, including (per user and cell id) uplink and downlink bitrate, modulation coding scheme (MCS), channel quality indicator (CQI), or signal-to-noise-ratio (SNR). The custom sampling function we developed, available at the referred repository [4], is a containerized Python script that opens a WebSocket against the Callbox API and exposes some metrics of interest to the Prometheus scraper. We showcase some of these RAN metrics in §V-B.

B. *Visualizing metrics with Grafana*

Once the data is being gathered, it is usually convenient to visualize it to quickly grasp the behavior of the network at different domains. For such a task, we use Grafana [33], a multi-platform open-source analytics and interactive visualization web application that is also included in the kube-prometheus stack. We modified the corresponding chart with the inclusion of two Grafana dashboard descriptors in JSON format for the experiments in §V-A and §V-B, respectively.

V. Use case evaluation

This section showcases and validates the presented end-to-end monitoring framework by displaying different metrics, including both infrastructure and RAN parameters. It does so through two use case experiments based on the testbed displayed in Fig. 3 and Fig. 4. The first experiment deals with UPF re-selection in MEC-enabled 5G networks, and the second focuses on RAN (gNB) measurements under UE mobility. The considered events are listed in Table I.

A. *UPF re-selection in MEC platforms*

As for Experiment #1, we trigger the following series of events: first (1), the 5GC CNFs are deployed through the installation of Helm charts, and the session management

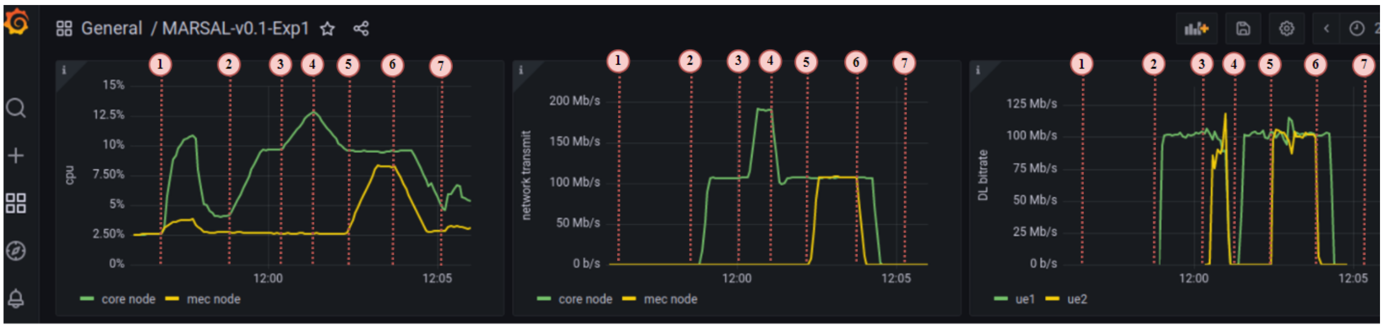

Fig. 5: Grafana dashboard for experiment #1 (UPF re-selection). Events are numerated within red circles.

function (SMF) assigns both UEs to the core UPF. Second (2), UE1 starts a 100 Mbps UDP downlink iperf connection from an iperf server located in the core node. Then (3), UE2 initiates an iperf of the same characteristics until (4), where the iperf stops and the SMF re-assigns UE2 to the MEC UPF. A new iperf connection is then started by UE2 pointing to the edge node iperf server in (5) until (6), the moment at which both UEs stop their corresponding iperf connections. Finally, the whole deployment (5GC CNFs) is terminated, and the Open5GS containers are deleted in the core and edge nodes.

Fig. 5 shows the Grafana dashboard used for Experiment #1 consisting of three panels: two for infrastructure metrics (node CPU and networking transmit) measured at the core and edge nodes, and one for a RAN metric (downlink bitrate) measured at the Amarisoft Callbox acting as gNB. The temporal events are highlighted with red circles. In (1), we observe a peak of CPU caused by the deployment of the 5GC CNFs in the nodes. In (2), CPU and network transmit increase due to the iperf traffic triggered by UE1. In (3), UE2's traffic raises the CPU and transmit networking of the core node, since UE2 is also assigned to the core UPF. Instead, when UE2 is assigned to the MEC UPF (5), CPU and networking resources are shared between the core and edge node. Finally, we observe a peak on CPU in (7) corresponding to the CNFs termination.

This use case shows the potential value of the presented monitoring framework in 5G deployments, where important metrics from different domains (e.g., infrastructure and RAN) can be assessed in an integrated and automated end-to-end manner.

TABLE I: List of events in experiments #1 (UPF re-selection) and #2 (UE mobility).

Event	Description
#1.1	5GC CNFs deployed and UEs assigned to core UPF
#1.2	UE1 starts a 100 Mbps UDP downlink iperf connection
#1.3	UE2 starts a 100 Mbps UDP downlink iperf connection
#1.4	UE2 iperf stopped and assigned to MEC UPF
#1.5	UE2 restarts a 100 Mbps UDP downlink iperf connection
#1.6	Both UEs stop their corresponding iperf connections
#1.7	Whole deployment (5GC CNFs) is terminated
#2.1	UE1 starts a 120 Mbps uplink iperf to the core node
#2.2	gNB reduces the receiver gain 4dB (-4 dB aggregated)
#2.3	gNB reduces the receiver gain 4dB (-8 dB aggregated)
#2.4	gNB reduces the receiver gain 4dB (-12 dB aggregated)

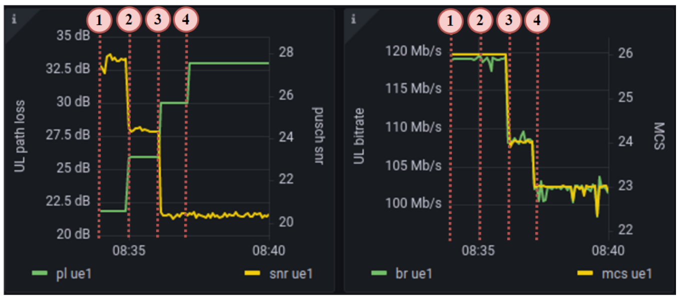

Fig. 6: Grafana dashboard for Experiment #2 (UE mobility). Events are numerated within red circles.

B. *UE mobility*

Finally, to test the containerized Amarisoft sampling function, in Experiment #2 we focus solely on RAN metrics. To do so, we show in Fig. 6 a Grafana dashboard corresponding to a scenario where UE1 moves away from the gNB, resulting in higher path loss (lower SNR) and lower MCS and, consequently, lower bit rates. In particular, the series of events is as follows: (1) UE1 starts a 120 Mbps uplink iperf connection to the core node, and from (2) to (4) we sequentially reduce by 4 dB the receiver gain at the gNB through the Amarisoft API to emulate UE1 moving away. As expected, this results in higher path loss and lower SNR at the gNB, consequently achieving lower MCS and lower bitrate.

Experiment #2 has therefore demonstrated the suitability of the developed Amarisoft sampling function in terms of integrating pure RAN metrics into the whole monitoring framework. Providing such end-to-end monitoring is of critical importance to let 5G network administrators control and manage their services, especially when it comes to end-to-end network slicing.

VI. Conclusions

In this work, we have implemented, tested, and validated a cloud-native 5G framework with containerized end-to-end monitoring. Using a MEC-enabled 5G testbed with over-the-air transmissions, we depict how to integrate a fully operative 5G framework using CNFs in a multi-node Kubernetes cluster, including an open-source 5G core, a commercial RAN, and an end-to-end monitoring system. We demonstrate the multi-UPF and monitoring capabilities of the framework via Grafana dashboards that display different metrics of interest, gathered both from the infrastructure and radio/RAN domains. Therefore, we expect this paper helps the community in easily deploying 5G monitoring through an integrated and automated end-to-end manner.

As for future work, we are endowing our experimental platform with intelligence so that, e.g., different configurations for the placement of NFs can be adopted as a function of certain events or alerts triggered by the monitoring platform. Besides, we plan to provide an in-depth comparison of our framework when deployed with other relevant open-source 5G cores.

Acknowledgment

This work has been partially funded by the MARSAL project from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017171.

References

[1] I. P. Chochliouros, A. Kostopoulos, M. Payaró, C. Verikoukis, S. D. C. d. Vimercati, E. Vinogradov, V. Ranjbar, J. Vardakas, M. A. Rahman, P. Soumplis *et al.*, "Machine learning-based, networking and computing infrastructure resource management," in *IFIP International Conference on Artificial Intelligence Applications and Innovations*. Springer, 2021, pp. 85–94.

[2] J. S. Vardakas, K. Ramantas, E. Datsika, M. Payaró, S. Pollin, E. Vinogradov, M. Varvarigos, P. Kokkinos, R. González-Sánchez, J. J. V. Olmos *et al.*, "Towards machine-learning-based 5g and beyond intelligent networks: The marsal project vision," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2021, pp. 488–493.

[3] M. Mekki, S. Arora, and A. Ksentini, "A scalable monitoring framework for network slicing in 5g and beyond mobile networks," *IEEE Transactions on Network and Service Management*, 2021.

[4] S. Barrachina-Muñoz, "Cloud-native 5G experimental platform with end-to-end monitoring," https://doi.org/10.5281/zenodo.6075481, 2022.

[5] C. Novaes, C. Nahum, I. Trindade, D. Cederholm, G. Patra, and A. Klautau, "Virtualized c-ran orchestration with docker, kubernetes and openairinterface," in *XXXVII SIMPOSIO BRASILEIRO DE TELECOMUNICACOES E PROCESSAMENTO DE SINAIS*, 2020.

[6] J. Haavisto, M. Arif, L. Lovén, T. Leppänen, and J. Riekki, "Open-source RANs in Practice: an Over-the-air Deployment for 5G MEC," in *2019 European Conference on Networks and Communications (EuCNC)*. IEEE, 2019, pp. 495–500.

[7] O. Arouk and N. Nikaein, "Kube5G: A cloud-native 5G service platform," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[8] ——, "5G cloud-native: network management and automation," in *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*. IEEE, 2020, pp. 1–2.

[9] L. Osmani, T. Kauppinen, M. Komu, and S. Tarkoma, "Multi-cloud connectivity for kubernetes in 5g networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 42–47, 2021.

[10] P. Trakadas, P. Karkazis, H.-C. Leligou, T. Zahariadis, W. Tavernier, T. Soenen, S. Van Rossem, and L. Miguel Contreras Murillo, "Scalable monitoring for multiple virtualized infrastructures for 5g services," in *SoftNetworking 2018, The International Symposium on Advances in Software Defined Networking and Network Functions Virtualization*, 2018, pp. 1–4.

[11] D. Giannopoulos, P. Papaioannou, C. Tranoris, and S. Denazis, "Monitoring as a service over a 5g network slice," in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. IEEE, 2021, pp. 329–334.

[12] D. Giannopoulos, P. Papaioannou, L. Ntzogani, C. Tranoris, and S. Denazis, "A holistic approach for 5g network slice monitoring," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*. IEEE, 2021, pp. 240–245.

[13] E. Goshi, M. Jarschel, R. Pries, M. He, and W. Kellerer, "Investigating inter-nf dependencies in cloud-native 5g core networks," in *2021 17th International Conference on Network and Service Management (CNSM)*. IEEE, 2021, pp. 370–374.

[14] X. Li, A. Garcia-Saavedra, X. Costa-Perez, C. J. Bernardos, C. Guimarães, K. Antevski, J. Mangues-Bafalluy, J. Baranda, E. Zeydan, D. Corujo *et al.*, "5growth: An end-to-end service platform for automated deployment and management of vertical services over 5g networks," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 84–90, 2021.

[15] 3GPP, "System architecture for the 5G System (5GS)," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 23.501, 09 2021, v17.2.0.

[16] "Whitepaper: The status of open source for 5g," 5G Americas, Tech. Rep. MSU-CSE-06-2, 2019. [Online]. Available: https://www.5gamericas.org/the-status-of-open-source-for-5g/

[17] E. Pateromichelakis, F. Moggio, C. Mannweiler, P. Arnold, M. Shariat, M. Einhaus, Q. Wei, Ö. Bulakci, and A. De Domenico, "End-to-end data analytics framework for 5G architecture," *IEEE Access*, vol. 7, pp. 40 295–40 312, 2019.

[18] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet, "OpenAirInterface: A flexible platform for 5G research," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 5, pp. 33–38, 2014.

[19] "Free5GC," https://www.free5gc.org/, accessed: 2022-02-07.

[20] "Open5GS," https://open5gs.org/, accessed: 2022-02-07.

[21] B. Chun, J. Ha, S. Oh, H. Cho, and M. Jeong, "Kubernetes enhancement for 5G NFV infrastructure," in *2019 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2019, pp. 1327–1329.

[22] "Use containers to build, share and run your applications," https://www.docker.com/resources/what-container, accessed: 2022-02-07.

[23] "Kubernetes website," https://kubernetes.io/, accessed: 2022-02-14.

[24] "Helm charts," https://helm.sh/docs/topics/charts/, accessed: 2022-02-07.

[25] "What is Calico," https://projectcalico.docs.tigera.io/about/about-calico, accessed: 2022-02-07.

[26] "UERANSIM GitHub repository," https://github.com/aligungr/UERANSIM, accessed: 2022-02-08.

[27] "AMARI Callbox Ultimate," https://www.amarisoft.com/app/uploads/2022/01/AMARI-Callbox-Ultimate.pdf, accessed: 2022-02-08.

[28] "AMARI UE Simbox Series," https://www.amarisoft.com/app/uploads/2021/12/AMARI-UE-Simbox-E-Series.pdf, accessed: 2022-02-08.

[29] "Iperf website," https://iperf.fr/iperf-doc.php, accessed: 2022-02-09.

[30] "Kube-prometheus GitHub repository," https://github.com/prometheus-operator/kube-prometheus, accessed: 2022-02-08.

[31] B. Rabenstein and J. Volz, "Prometheus: A next-generation monitoring system (talk)." Dublin: USENIX Association, May 2015.

[32] Y. Tseng, G. Aravinthan, B. Berde, S. Imadaliz, D. Houatra, and H. Qiu, "Re-think monitoring services for 5G network: challenges and perspectives," in *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*. IEEE, 2019, pp. 34–39.

[33] "Grafana website," https://grafana.com/, accessed: 2022-02-08.