5G Cloud-Native: Network Management & Automation

Osama Arouk and Navid Nikaein
Communications Systems Department - EURECOM, Biot, France - Email: firstname.lastname@eurecom.fr

***Abstract*—In this demo, we present 5G network automation in cloud-native environment. Our proposition is to demonstrate the network automation using Kubernets as container orchestration and automating application deployment, while using Openshift Operator as a tool to manage complex services, such as 5G services. For this purpose, we use the containerized OpenAirInterface (OAI) to deploy the network and demonstrate the automatability, such as dynamic switch of RAN between monolithic base station and disaggregated RAN (i.e. Distributed Unit-DU and Centralized Unit-CU), and auto-configuration.**

***Index Terms*—4G/5G, cloud-native, NFV, Network Management and Automation, kubernetes, OpenAirInterface (OAI), Openshift Operator**

I. Introduction

Fast, access at scale, frequent deployment of services, and quick failure recovery (e.g., service availability >= 99.999) became important features to support current and novel services of 5G networks in a virtualized environment. Such network automation in 5G networks as required for network slicing, as well as the evolution of the services (e.g., service update/upgrade, scaling etc.), can be achieved by softwarization, virtualization, and cloud computing technologies.

Cloud-native approach is a way to build and run applications that fully exploit the cloud computing model. Autoscaling, self-healing, agility, among others, are considered as the core features that are envisioned for 5G in cloud native environment. Note that Cloud Native Computing Foundation (CNCF) is one of the main drivers for the adoption of such paradigm by fostering and sustaining an ecosystem of open source and vendor-neutral projects [1]. Generally, cloud-native applications have three main features. Firstly, they are composed of microservices [2], where a cloud-native application can be composed of many services operating independently of each other. Secondly, the cloud-native applications are packaged in containers, and thus ensuring the context's isolation of microservices. Finally, they run in a continuous delivery model, which ensures fast cycles of building, testing, deploying, releasing, and developing the applications.

One of the main features of microservices and containers is that they have small footprint and fast start times, which is an important feature for 5G in the cloud. There are many technologies for containerization, such as Linux Containers (LXC), containerd, CRI-O, docker. However, it is hard to decide which containerization is better for 5G, since they have comparatively similar performance [3]. However, we use docker due to its maturity and support by large community.

978-1-7281-4973-8/20/$31.00 © 2020 IEEE

In order to cloudify the network, flexible functional split was introduced by 3GPP [TR 38.801] to cope with the ever increasing COPEX/CAPEX with the introduction of new services in 5G. Contrary to monolithic 4G RAN, the RAN of 5G is disaggregated into three main units: the Remote Radio Unit (RRU), the Distributed Unit (DU), and the Centralized Unit (CU). All the necessary components related to signal transmission/reception exist in RRU [4], while DU may contain a set of physical layer functions shifted to the cloud as well as some set of higher layer functions. The rest of higher layer functions will be then treated/processed at the CU. 3GPP defined 8 types of functional split, where their requirements (regarding CPU, latency, and fronthaul rates) are different for every split.

After shifting most of the 5G functions to the cloud (i.e., build them as a cloud-native applications), the management and orchestration of the ecosystem is required in order to achieve the envisioned performance. Several frameworks already exist, such as Kubernetes [5], Docker Swarm, and Apache Mesos. Kubernetes is considered as a good candidate for supporting 5G services, since it can well support the flexibility required in 5G [6], [7], and it is taking the momentum [1]. Moreover, it has bigger community, and thus guaranteeing better support on a long term. For the ease of packaging, deploying and managing a Kubernetes application, the openshift Operator framework [8] was introduced (deailed in the next section).

In this paper, we present our demo on 5G dynamic network automation and management in cloud-native environment. More specifically, we demonstrate the ability to support 5G in cloud native and dynamically automate the network, i.e. upgrade/downgrade services, change the configuration automatically, and dynamically switch between monolithic eNB and functional split (CU-DU).

II. OPENSHIFT OPERATOR & 5G NETWORK AUTOMATION

In order to fully automate the deployment and managing the lifecycle of applications under Kubernetes, we use openshift Operator, or simply Operator. Operator is capable of managing complex services, which is managed by administrators and their operational scripts, e.g. ansible. Since Operator can be run inside Kubernetes as a service, it's thus more portable and has better integration. The Operator utilizes what is called Controller to achieve its ability. Considered as one of the core concepts of Kubernetes, a Controller is an entity that continuously loops on the master node of Kubernetes. With

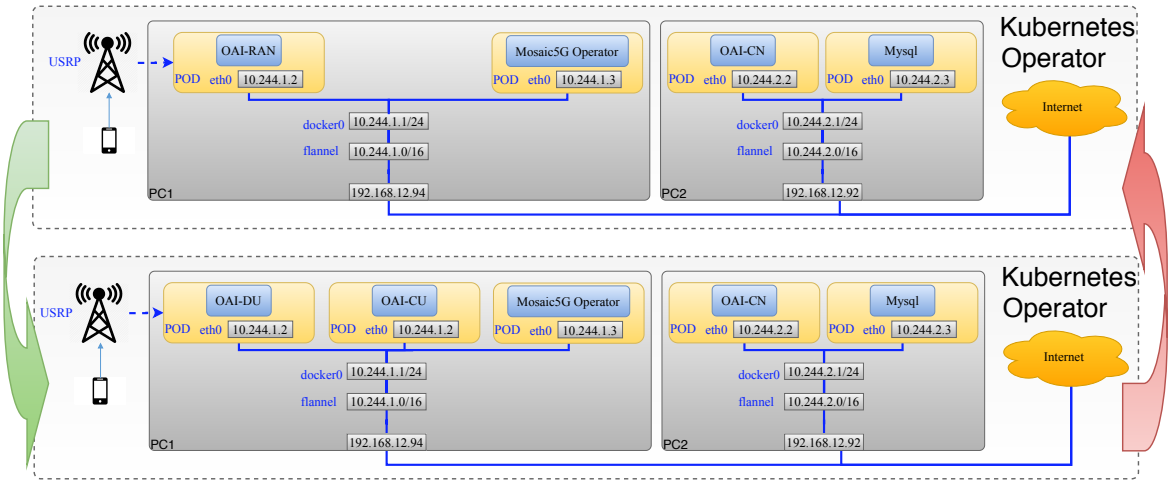

Fig. 1. Network slice lifecycle and resource monitoring

the help of client-go[1][2], this entity will listen to the cluster for any changes in the resources. Whenever these chanegs are detected, the controller will perform the corresponding action(s). For example, a Deployment controller will listen to any changes to its pods, and it will make changes if the state of that pods is not correct. For the objective of managing complex services like 5G, we implemented an openshift operator, dubbed as Mosaic5G Operator, using Openshift Operator SDK[3]. Currently, this Mosaic5G Operator supports the following: i) service deployment, ii) (re)configuration, iii) upgrading/downgrading of service's version. Note that this operator is implemented using golang, since it has more capability than the other two types of operator (namely helm and Ansible) like auto-pilot and deep insights [9].

III. Demo Reproducibility and Potential Impact

All the tools that are used by this demo are open source. Therefore, by setting up correctly kubernetes and openshift operator, the research community can reproduce the current demo. This demo can help to deploy 5G network on large scale, while managing it according to the current user traffic (e.g., switch to monolithic RAN in case of high user traffic or to disaggregated RAN when low user traffic is detected).

IV. Proof of Concept and Planned Demo

Fig. 2 illustrates the entities of 4G/5G network that are considered in our demo. These entities are: i) mysql, ii) oai-cn that includes oai-hss, oai-mme, oai-spgw, iii) oai-ran that is oai-enb for monolithic mode and oai-cu/oai-du for disaggregated mode. Note that the USRP and the antenna act as forntend to connect the User Equipment (UE), which can be any commercial mobile phone, to the network. The detailed setup is illustrated in Fig. 1, where a 4G/5G network is deployed using Mosaic5G operator and kubernetes cluster with two nodes: one as master node, and the other as worker node. Moreover, Mosaic5G operator is used for the automation,

[1]client-go is client used to talk to Kubernetes cluster
[2]https://github.com/Kubernetes/client-go
[3]https://github.com/operator-framework/operator-sdk

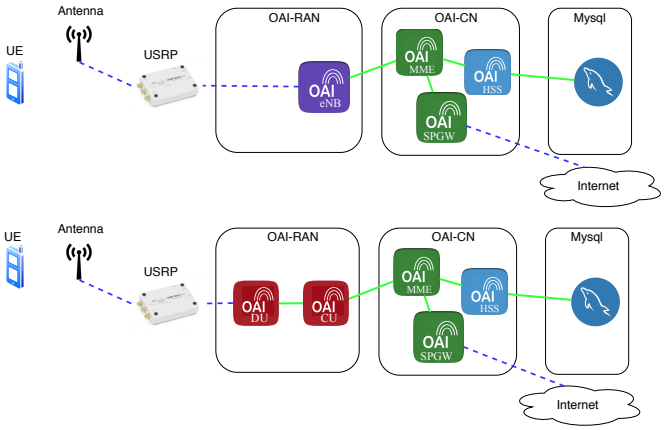

Fig. 2. 4G/5G network illustration: Monolithic RAN (top), versus disaggregated RAN (bottom)

where three principal features are demonstrated: i) change of network configuration, ii) network upgrade (i.e., upgrade one or more network entities), iii) switch between monolithic RAN (top topology in Fig. 1) and disaggregated RAN (bottom topology in Fig. 1) and vice versa according to user traffic.

References

[1] CNCF, "CNCF: Building Sustainable Ecosystems for Cloud Native Software." [Online]. Available: https://www.cncf.io/
[2] M. Fowler, "Microservices: a definition of this new architectural term ." [Online]. Available: https://martinfowler.com/articles/microservices.html
[3] Kubernetes, "Kubernetes Container Runtimes." [Online]. Available: https://kubedex.com/kubernetes-container-runtimes/
[4] N. Nikaein, E. Schiller, R. Favraud, R. Knopp, I. Alyafawi, and T. Braun, *Towards a Cloud-Native Radio Access Network.* Springer International Publishing, 2017, pp. 171–202.
[5] Kubernetes, "Kubernetes: Production-Grade Container Orchestration." [Online]. Available: https://kubernetes.io/
[6] 5G-PPP, "From Webscale to Telco, the Cloud Native Journey," 5G-PPP Software Network Working Group, Tech. Rep., 08 2018.
[7] 5G-PPP, "Cloud-Native and Vertical services," 5G-PPP Software Network Working Group, Tech. Rep., 08 2019.
[8] RedHat, "Introducing the Operator Framework: Building Apps on Kubernetes." [Online]. Available: https://blog.openshift.com/introducing-the-operator-framework/
[9] Redhat, "Understanding Operators." [Online]. Available: https://docs.openshift.com/container-platform/4.2/operators/olm-what-operators-are.htm