Financiado por la Unión Europea NextGenerationEU

MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL

Plan de Recuperación, Transformación y Resiliencia

uc3m

6G-CLARION-SI Entregable E8

Use cases definition

PROGRAMA DE UNIVERSALIZACIÓN DE INFRAESTRUCTURAS DIGITALES PARA LA COHESIÓN UNICO I+D 5G 2021

UNICO I+D

Fecha: 31/7/2022

Versión: 1.0

Propiedades del documento

Id del documento	E8			
Título	Use cases definition			
Responsable	UC3M			
Editor	Albert Banchs			
Equipo editorial	**Partner**	**Name**	**Surname**	**Sections**
	UC3M	Pablo	Serrano	All
Nivel de diseminación	Público			
Estado del documento	Final			
Versión	1.0			

Historial

Revisión	Fecha	Por	Descripción
1.0	31/07/22	Editor	Final version

Revisor

Equipo revisor	**Partner**	**Name**	**Surname**	**Sections**
	UC3M	Marco	Gramaglia	All

Descargo de responsabilidad

Table of Contents

Lista de acrónimos

AI: Artificial Intelligence

IoT: Internet of Things

ML: Machine Learning

CPU: Central Processing Unit

GPU: Graphics Processing Unit

RAM: Random Access Memory

API: Application Programming Interface

HTTP: Hypertext Transfer Protocol

DNS: Domain Name System

VPN: Virtual Private Network

OAI: Open Air Interface

RAN: Radio Access Network

NFV: Network Functions Virtualization

UE: User Equipment

VIM: Virtual Infrastructure Manager

LTE: Long-Term Evolution

5G: Fifth Generation of Mobile Networks

Resumen ejecutivo

Las tecnologías de red están adoptando el paradigma nativo de la nube, siguiendo las mejores prácticas actuales en la informática en la nube. Las tecnologías nativas de la nube podrían aplicarse a diferentes tipos de funciones de red en una red móvil, pero son particularmente relevantes en la actualidad para las funciones de red centrales, ya que el estándar reciente introduce la Arquitectura Basada en Servicios que se adapta a tecnologías nativas de la nube modernas como Docker o Kubernetes. Al mismo tiempo, varios proyectos de software de código abierto ya proporcionan a los investigadores y profesionales software utilizable que implementa la funcionalidad clave de una red móvil (tanto para LTE como para 5G). Sin embargo, estas soluciones de software son monolíticas y no están integradas en marcos nativos de nube de última generación. En este documento, cubrimos esta brecha describiendo la implementación de una red móvil nativa de la nube, que admite la emulación de canal y proporciona una forma económica y escalable de probar algoritmos de orquestación con interfaces VIM estandarizadas.

Abstract

Network technologies are embracing the cloud-native paradigm, following the current best practices in cloud computing. Cloud-native technologies might be applied to different types of network functions in a mobile network, but they are particularly relevant nowadays for core network functions, as the recent standard introduces the Service-Based Architecture that matches modern cloud-native technologies such as Docker or Kubernetes. In parallel, a number of open source software initiatives already provide researchers and practitioners with usable software that implements the key functionality of a mobile network (both for LTE and 5G). These software solutions, however, are monolithic and not integrated into state-of-the-art cloud native frameworks. In this paper, we fill this gap by describing the implementation of a cloud-native mobile network, which supports channel emulation and provides an affordable and scalable way of testing orchestration algorithms with standardized VIM interfaces.

1. Introduction

The emergence of 5G networks and their diverse characteristics have created a need for higher levels of network flexibility, programmability, and automation. Network Function Virtualization (NFV) is a technology that addresses these needs and is considered one of the most important drivers for 5G networks and beyond[1]. NFV allows for the softwarization of the network, enabling the deployment of services on commodity off-the-shelf servers (COTS) traditionally used for cloud computing. This results in reduced operational expenses, as services can be multiplexed over the same physical infrastructure, resources can be shared between functions, and services can be deployed on-demand per user load.

However, virtualization technologies based on virtual machines are not always efficient for fast timescale operations, and the tight dependency between software and hardware can hinder the deployment of NFV technology[2]. Cloud Network Functions (CNF)[3] have emerged as a potential solution as they are already widely used in cloud computing environments and are tested for their robustness. CNFs rely on a thin virtualization layer and have a minimal footprint on the infrastructure, enabling modern software architectures such as microservices and providing a high degree of configurability. However, CNFs lack the maturity required for adoption in production environments, particularly in the telecommunications domain.

To address this issue, the authors of this paper present an end-to-end mobile network with channel emulation capabilities, monitored and managed by a CNF orchestrator that is released as open source. The paper describes the different building blocks required to build a cloud-native mobile network, including the channel emulation module, data shippers to monitor network functions, and configuration blueprints to automate orchestration processes. Two use cases are developed to illustrate the features of the solution and validate the behavior of the tools and modules developed.

The proposed cloud-native mobile network is available in open source, and interested practitioners can experiment with the solution. The codebase is split into core and orchestration, available at https://github.com/kaposnick/k8s-open5gs, and access available at https://github.com/kaposnick/srslte.

2. State of the art

Various studies have investigated the deployment of open source software for end-to-end mobile networks, but most of these works have focused on large-scale deployments. For example, some studies have used Open Air Interface (OAI) as the RAN solution with hardware-based radio interfaces, limiting the evaluation of more complex channel conditions and mobility scenarios. Others have proposed extensions to Kubernetes resources' specification to support NFV loads but lack control and user plane separation at the NF granularity. While some experiments have focused on the RAN component, they also face limitations in extensibility to more UEs or different radio conditions due to their reliance on real radio front-ends. Meanwhile, large-scale radio testbeds have less focus on the core part, which is a critical aspect for testing end-to-end solutions. To address this gap, we have designed and developed an end-to-end cloud-native open source 4G/5G mobile

---

[1] D. Bega et al., "Towards the Network of the Future: From Enabling Technologies to 5G Concepts", Trans. Emerging Telecommun. Tech., vol. 28, no. 8, pp. e3205, 2017.

[2] A. Morris, The Long Voyage to NFV: Vodafone and Orange Outline the Challenges Ahead, [online] Available: fiercewireless.com.

[3] S. R. Chowdhury et al., "Re- Architecting NFV Ecosystem with Microservices: State of the Art and Research Challenges", IEEE Network, vol. 33, no. 3, pp. 168-76, May/June 2019.

network using only conventional hosts and no RF hardware front-end. Our solution allows for rapid, easy, and cost-efficient deployment of small experimental deployments for research and educational purposes, providing control and user plane separation at the NF granularity and channel emulation capabilities without reliance on real radio front-ends.

3. Design of a Cloud Native Ecosystem

The softwarization of mobile network functions has allowed network operators to reduce their operational costs by using open software. It has also enabled researchers and practitioners to experiment with new network functionalities. With the introduction of cloud-native architectures, this software-driven transition is being accelerated, which greatly increases the flexibility and scalability of the software.

In this document, we present the different components of our cloud-native mobile network design, several of which have been chosen from the set of open-source solutions that have become available during the last few years.

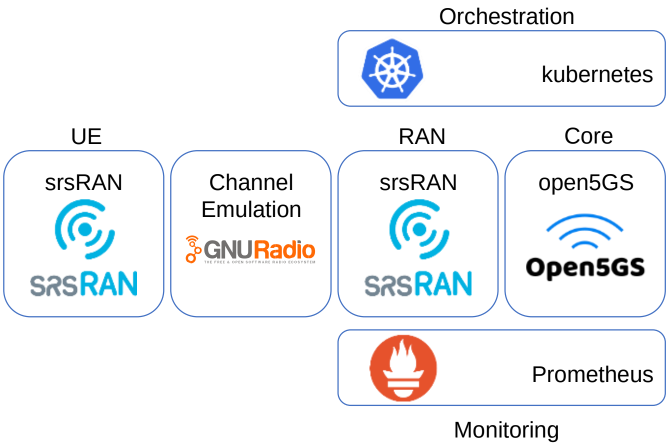

Figure 1 The solutions that build the end-to-end architecture presented in this document

3.1. Overview

Figure 1 illustrates the solutions that build the end-to-end architecture presented in this paper. For orchestration, we chose Kubernetes, an open-source system for deploying and managing containerized applications. For the core network implementation, we selected Open5Gs, which is adaptable to the cloud-native paradigm. For the UE and the Access network, we rely on srsRAN, an open-source software that provides a full mobile network stack implementation for these elements. We developed a simple channel emulator to simplify the deployment and support repeatability. Finally, we chose Prometheus as the monitoring framework to collect the status of the whole system. In what follows, we detail these components.

3.2. Management and Orchestration

Kubernetes is a container orchestration platform that has been widely adopted in the cloud computing domain for over five years. A typical cloud-native application consists of a set of container images that are deployed and managed by Kubernetes, using configuration files that describe their functionality and the services they provide. Its scaling and redundancy capabilities have already been demonstrated in production environments. Telco operators are expected to rely on Kubernetes for next-generation networks, motivated by the need to make more efficient use of the heterogeneous virtualized infrastructure, which spans over central and edge data centers.

3.3. Core Network Functions

Open5Gs is a very popular open-source implementation of a mobile network core. It stands as a reference among researchers and mobile telecommunications practitioners for experimentation and future enhancements. Currently supporting up to 3GPP 5G Release 16, it contains the most important components of the 5G Core (5GC) and 4G Evolved Packet Core (EPC) with Control-User Plane Separation (CUPS). Its straightforward build procedure makes it easy to deploy in small-scale private networks, while its modular architecture is a natural candidate for running it in microservices-based cloud-native environments powered by Kubernetes.

3.4. RAN Functions

srsRAN is another open-source software framework that provides the functionality from the PHY up to RRC layer for eNB/gNBs, while also supporting 4G/5G UEs. Its configuration parameters cover a wide range of base station and UE possible configurations. Regarding the RF interface, its contributors have developed drivers for various commercial hardware RF-frontends. Additionally, srsRAN provides a software RF-frontend based on ZeroMQ, an open-source message queuing library written in C. Choosing this driver avoids the need for high expertise in RF channel configuration and facilitates the introduction of researchers who want to simulate a radio access network environment.

3.5. Channel Emulation: The GRC Broker

The use of ZeroMQ enables running the network in a fully softwarized way, including the emulation of complex topologies via programming. We utilized the GNU Radio Companion (GRC), which contains modules for the ZeroMQ library, to code a GRC Broker between the UE and the cells of the eNB(s) implementing the RF interface. This allows the emulation of complex mobility scenarios and the handling of I/Q symbols, e.g., sending low power symbols from the eNB to trigger a Handover Request from the UE.

3.6. Monitoring

Monitoring the mobile network's status is crucial for maintaining its efficiency and reliability. To achieve a unified view of the network's health status, we deployed the cloud-native version of Prometheus (kube-prometheus-stack) in the same Kubernetes cluster used for Open5Gs. In this section, we will describe the steps taken to monitor both Core and RAN through Prometheus.

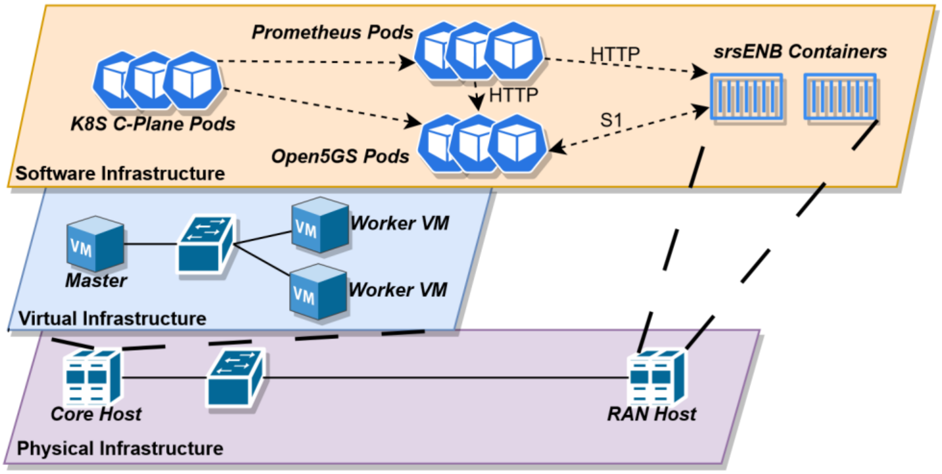

Figure 2 Core and RAN infrastructure setup

Core Network: When kube-prometheus-stack is deployed, exporters are automatically installed in all the master and worker nodes by default. These exporters collect and expose metrics on different granularity levels such as CPU usage, RAM, Networking status, and IOPS for nodes, pods, and containers. By exporting these metrics, we can monitor the virtual infrastructure running the different core network functions, as depicted in Figure 2. The Prometheus operator pods have set these exporters as targets and will start polling them in fixed intervals using HTTP in a well-known path: (/metrics). These metrics are directly accessible through the Prometheus WebUI or Grafana using PromQL, a functional query language.

RAN: srsRAN produces an array of metrics for the lower layers of the networking stack, which can be exported to the console or to a text file. However, srsRAN does not have an agent that acts as a metrics exporter towards third-party software. Therefore, we modified the source code of the srsENB to develop and integrate a new one into the final binary. We integrated the lighttpd project, a lightweight open-source HTTP server written in C programming language, and developed a REST API through which the metrics, in key-value pair format, will be exported to the Prometheus operator. Afterward, we developed Kubernetes manifests that comprise a pod, whose role is to proxy the HTTP requests, received by the Prometheus operator, towards the end srsENB process. These manifests are deployed to the Kubernetes cluster with the IP address of the srsENB as an input parameter. Lastly, we deployed a ServiceMonitor configuration resource, which automatically discovers the 'proxy' pods existing in the cluster and sets them as Prometheus targets, facilitating the discovery capabilities of Prometheus.

Thus, for every srsENB process we instantiate, an autodiscovery service is performed, and once it is completed a few seconds later, the eNB starts getting polled (or scraped, using the Prometheus terminology). This enables us to monitor both Core and RAN through Prometheus and have a consistent view of the mobile network's current status in a unified platform.

4. Conclusion

Telecommunication operators are migrating their infrastructure to cloud environments, replacing statically deployed PNFs with dynamic and flexible CNFs. To ensure predictability, fast fault detection, and scalability of heterogeneous network components, observability is crucial. As these new concepts are adopted, networking researchers require an easy and cost-effective way to deploy end-to-end mobile networks for experimentation and development. In this document, we have designed an end-to-end cloud-native 4G/5G mobile network by leveraging existing open-source software projects and developing new modules and tools. This cost-efficient solution enables the development and validation of novel control and orchestration algorithms without the need for specialized hardware.