# Multi-Access Edge Computing: Performance Optimization via Orchestration and Virtualization

Carlo Centofanti

Department of Information Engineering,
Computer Science and Mathematics

Ph.D. Program in ICT - System Engineering, Telecommunications and HW/SW Platforms
XXXV cycle - SSD ING-INF/03

Università degli Studi dell'Aquila

Advisor: Prof. Fabio Graziosi

Co-Advisor: Prof.ssa Dajana Cassioli

Coordinator: Prof. Vittorio Cortellessa

A thesis submitted for the degree of
*Doctor of Philosophy*

2023

# Abstract

Multi-access Edge Computing is an architectural model proposed by the European Telecommunications Standards Institute to support the traffic growth and latency requirements coming from new applications and use cases. In conjunction with existing access technologies (i.e.: 5G, Passive Optical Networks, etc.) it can enable services of the future which are hungry of high bit-rate and real low round trip time. This thesis is divided in three parts: in the first part, the theoretical background is given along with an overview of the main virtualization and orchestration technologies. In the second part of this thesis, the most relevant publications produced during the PhD period are appended. In the third and last part, conclusions and future research directions are drawn.

The content of this thesis evolves following two main directions: low latency and multimedia. In the low latency direction, a whole 5G enabled Open Radio Access Network (O-RAN) with Optical Access Network backhaul and a Service Based Architecture for the 5G core will be introduced, designed and realized along with a management and orchestration framework that takes decision to commission or decommission network slices, optimizing performances through the network. The multimedia direction will show performance optimization in video streaming and spatial audio scenarios, leveraging the edge capabilities of Multi-access Edge Computing.

Finally, a contribution to O-RAN security is shown as fundamental and transversal discipline. A significant contribution to this topic is proposed to enforce the O-RAN protection against malicious attackers.

# Acknowledgements

I really feel the need to thank a lot of people that are or are used to be in my life. I will try to put all of you on this acknowledgments but you know me, I will forgot some of you for sure.

First I would like to thank you, the one I forgot to mention below. I would also say sorry to you for the inconvenience.

To my family: thank you for letting me do what I want, with your infinite support.

Thank you Desiree, for your immense support during the last few years. I wouldn't have been here without you.

My special thanks to my advisor Prof. Fabio Graziosi, for his priceless help and sustain in time of need and his unlimited patience and his ability to say the right thing at the right time in every kind of situation.

Thank you Prof. Dajana Cassioli, your humanity and sincerity united with your outstanding intuitions and advices guided me through my entire PhD.

I would like to special thank Dr. Andrea Marotta. Thank you for your constant and solid guidance. You have always shown me the right way sincerely and professionally in recent years. Thank you for teaching me how to approach research, how to build bonds and connections with other researchers and how to react to the everyday pleasant and sometimes unpleasant situations in a more professional way than what I was used to. He is one of the most honest and professional people I have ever meet but also a sincere friend of mine.

Thanks to Prof. Koteswararao Kondepu, we met in person for very little time but shared our passion and knowledge and he taught me many things in this short time.

I would like to thank Dr. Venkateshwarlu Gundepu, a precious colleague and a friend of mine.

Thank you Dr. Luigi Pomante, you simply made my PhD possible in the very beginning.

Many thanks to Dr. Marco Santic, you convinced me to start the Ph.D. and gave to me a kick of trust and responsibilities. I will be forever grateful to you.

Thank you ALEX, we shared a lot of time, thoughts and concerns together during our Ph.D. course. We have finally reached the end of this path.

Thank you Mario, Ciro, Riccardo, Federica, Sabrina, Sara, and Fiordi. I do not know why but you do!

Thank you Dr. Leonardo D'Errico for teaching me an outstanding amount of technical and professional knowledge. You have always been one of the most honest friend of mine.

Many thanks to my colleagues from BiTe, WEST Aquila, RadioLabs and exEmerge for your support.

Thank you Dr. Claudia Rinaldi, your happiness and good vibes make the research look better.

Thank you Ing. Aldo Sebastiani and Ing. Umberto Mosca, you gave me a lot and in such a short time.

Thank you, Fabio Franchi. With your "Io questo Carletto lo voglio salvare", you put me on the right track and, in hindsight, that's true.

# List of Publications

**J:** Journal publication; **C:** Conference publication.

## Appended Papers:

C1 C. Centofanti, A. Marotta, C. Rinaldi, F. Franchi, D. Cassioli and F. Graziosi, "Improved DASH Video Streaming Performance by MEC-Enabled Optical Access", 2021 Asia Communications and Photonics Conference (ACP), Shanghai, China, 2021, pp. 1-3.

C2 C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, N. Sambo, L. Valcarenghi, C. Bernard, H. Roberts, "Slice Management in SDN PON Supporting Low-Latency Services", 2022 European Conference on Optical Communication (ECOC), Basel, Switzerland, 2022, pp. 1-4.

C3 C. Centofanti, A. Marotta, V. Gundepu, D. Cassioli, F. Graziosi, H. Roberts, C. Bernard, K. Kondepu, "End-to-end Slicing of RAN based on Next Generation Optical Access Network", IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Gandhinagar, India, 2022.

C4 C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, V. Gundepu, K. Kondepu, "End-to-End Slicing via O-RAN and Software Defined Optical Access", Optical Fiber Communication Conference (OFC), San Diego, California, USA, 2023.

C5 C. Centofanti, W. Tiberti, A. Marotta, F. Graziosi, D. Cassioli, "Latency-Aware Kubernetes Scheduling for Microservices Orchestration at the Edge", SUBMITTED TO IEEE International Conference on Network Softwarization (NETSOFT), Madrid, Spain, 2023.

J1 W. Tiberti, E. Di Fina, C. Centofanti, A. Marotta, D. Cassioli, "Security over the O-RAN Inter-Controller Interface: a Blockchain-based Anti-tampering Scheme for Traffic Policies", SUBMITTED TO Journal on Selected Areas in Communications.

J2 C. Rinaldi, F. Franchi, A. Marotta, F. Graziosi and C. Centofanti, "On the Exploitation of 5G Multi-Access Edge Computing for Spatial Audio in Cultural Heritage Applications," in IEEE Access, vol. 9, pp. 155197-155206, 2021, doi: 10.1109/ACCESS.2021.3128786.

# Contents

# List of Figures

# List of Tables

# Part I

# Thesis Background

# Chapter 1

# Introduction

The exponential growth of internet traffic which started since the inception of the World Wide Web shows no sign of stopping. Cisco's Annual Internet Report [1] states that the global Internet traffic is expected to reach 4.5 zettabytes per year by 2025, up from 1.2 zettabytes in 2016. This growth can be explained with the proliferation of cloud computing, the rise of video streaming services and the increase of connected devices, including smartphones that connect to the network. At the same time, access networks are evolving but not at the expected rate to catch the new traffic demand. Actual mobile access technologies i.e. 5G, stops at 25% worldwide coverage [2]. Traditional network architectures may incur in scalability issues when dealing with a so growing traffic. That is why European Telecommunications Standards Institute (ETSI) is focussing to develop new architectural solutions to face the change. Multi-access Edge Computing (MEC) is a modern network architecture proposed by ETSI that enables running application services closer to the end-users to reduce the experienced End-to-End (E2E) latency, increase bit-rate and enhance Quality of Experience (QoE). This is accomplished through the exploitation of two main factors: the edge, and the multi-access. The idea behind the modern MEC comes from the so called Mobile Edge Computing [3] which was a network architecture mainly oriented to mobile networks to reduce the gap to terrestrial networks. Nowadays, MEC aims to reduce the latency of applications and improve the overall performance of the network, especially for time-sensitive and high-bandwidth applications such as virtual and augmented reality, gaming, and industrial automation. The computing power is brought closer to the end-users by the means of micro data-centers deployed at the edge of the network such as cell towers, Point of Presence (PoP)s and other network access points. The presence of computation nodes at the edge provides many benefits to the overall network as it is possible to fully or partially process data at the edge,

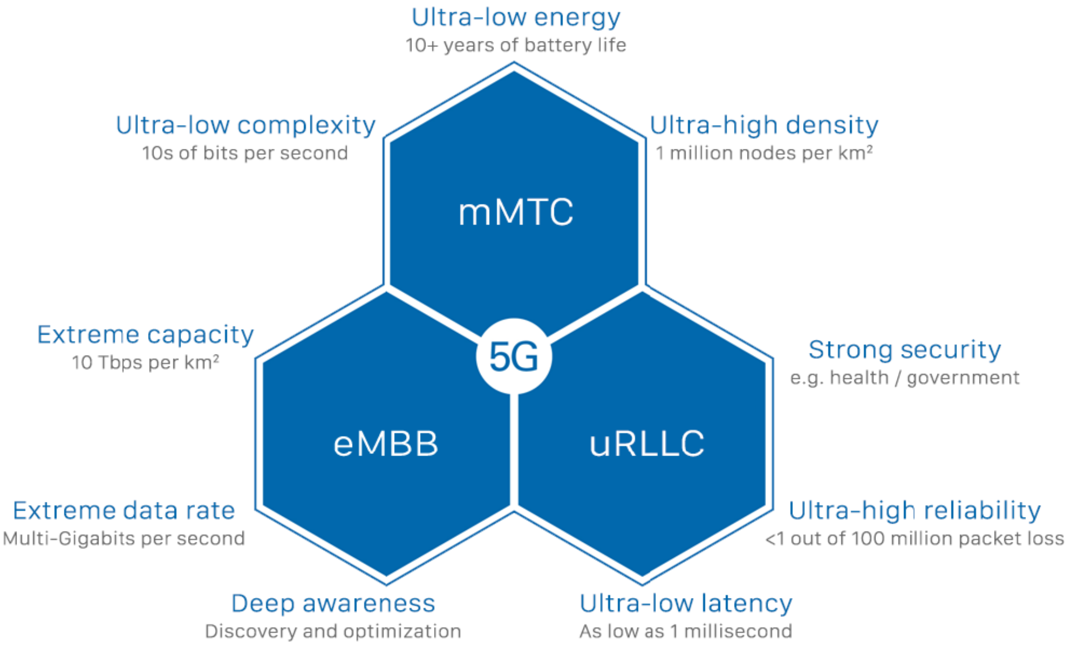


Figure 1.1: 5G Objectives.

extracting features and aggregating high level data at the edge while forwarding only important information to the central cloud. This may be very helpful in high density deployments of Internet of Things (IoT) network sensors to reduce the network load at backbone side. MEC is expected to be a technological enabler to the networks of the future [4] [5]. MEC architecture works well in all use cases where network Key Performance Indicator (KPI)s play a crucial role to the application's business.

5G is the fifth generation of mobile network architecture, designed to deliver lower latency, faster data speed, more reliable connectivity, higher energy optimization, and higher density in terms of connected devices while enabling low deployment time for new network services. The 5G core concepts are based around three core directions which are: Massive Machine-Type Communications (mMTC), Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low Latency Communications (uRLLC). Around those directions, 5G use cases are shown in Fig. 1.1. To support those new technologies and enable the technological progress beyond 5G, all the networking layers need to be optimized too. To ship low latency and high bandwidth requirements needed by the applications, new paradigms need to be implemented into the network architecture.

## 1.1 Basic concepts

All new architectures have an impact on previous generation architectures and how they should transit into the new ones. The period of coexistence may be shorter or longer depending on the complexity of the change and on economical drivers. The higher economics pushes changes, the faster is the adoption of new technologies but there is a technical upper-bound limit to this change that depends on technology. When ETSI started the Mobile Edge Computing standardization process, they began from mobile architectures, defining how to deliver computation and storage facilities to the mobile edge. The next step has been to generalize that approach extending the concept to all kind of access networks. This led the change from Mobile to Multi-access in the MEC acronym. The following section will give an overview on basic concepts needed to understand this thesis.

### 1.1.1 Multi-access

The multi-access part of MEC refers to the ability of the edge computing to be connected to many access networks so providing low latency to different users. This part of the MEC architecture introduces new challenges to the telecommunication operators as it means they should open in some manner their core networks to external MEC nodes or allow their internal ones to communicate with external world. One more beneficial effect coming from the introduction of such architecture is that in a Metropolitan Area Network (MAN) would be possible to have very low latencies, so enabling uRLLC scenarios.

### 1.1.2 Edge Computing

The term ”edge” refers to a model that aims to bring computational and storage resources at the edge of an Autonomous System (AS) to provide different improvements both to the perceived QoE, to the Quality of Service (QoS), and to network performance optimization in terms of forwarded packets. The QoE grows as a phenomenon directly influenced by QoS and the optimization of forwarded traffic inside the AS ensures a reduction of forwarding queue into the network nodes and a better optimization of the used network resources by the owner of the AS. Mobile Edge Computing, as a precursor of the MEC, estabilished in the perimeter of the Telco Mobile Core Network.

### 1.1.3 Economical drivers

Today we are used to live in a cloud centered world where all the computation and storage resources are centralized into hyper converged and large scale data-centers. This facilities are usually owned by the Over The Top (OTT) i.e. Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), Alibaba Cloud. The OTTs share an exponentially growing market that was estimated in $217 billion over 12 months. In this market, AWS, Azure and GCP account for two thirds of total revenues and the first eight OTT control the 80% of the market [6]. Latency in content distribution has its own cost in terms of business. A high latency reflects in end-user dissatisfaction and, finally, brings to revenue losses. Ookla shows that every 100 ms of additional latency on Amazon.com, cost the company 1% of revenue [7].

Over $700 billion in cumulative CAPital EXpenditure (CAPEX) will be spent within the next decade on edge IT infrastructure and data center facilities. The COVID-19 pandemic acted as an accelerator and a catalyst for national and international founds issued to cover edge and related networking infrastructure problems. In 2021, Congress authorized the Emergency Connectivity Fund (ECF) with $7,17 billion budget under the American Rescue Plan Act. at the same time, the The bipartisan Infrastructure Investment and Jobs Act included a $65 billion investment in broadband access in the USA [8]. The European Commission is financing Edge Computing through the Horizon Europe cluster 4 program with a first drop of €64 million [9].

### 1.1.4 Quality of Service Vs Quality of Experience

QoS and QoE are two metrics used to evaluate the quality of services at different layers. The metrics are defined by International Telecommunication Union (ITU) by the ITU Telecommunication Standardization Sector (ITU-T) group. ITU-T defines QoS and QoE as follows:

- **QoS defined by ITU-T E.800 (09/08):** *"totality of characteristics of a telecommunications service that bear on its ability to satisfy stated and implied needs of the user of the service."*

- **QoE defined by ITU-T G.1080 (12/08):** *"the overall acceptability of an application or service, as perceived subjectively by the end-user. It includes the complete end-to-end system effects (client, terminal, network, services infrastructure, etc.) and maybe influenced by user expectations and context."*

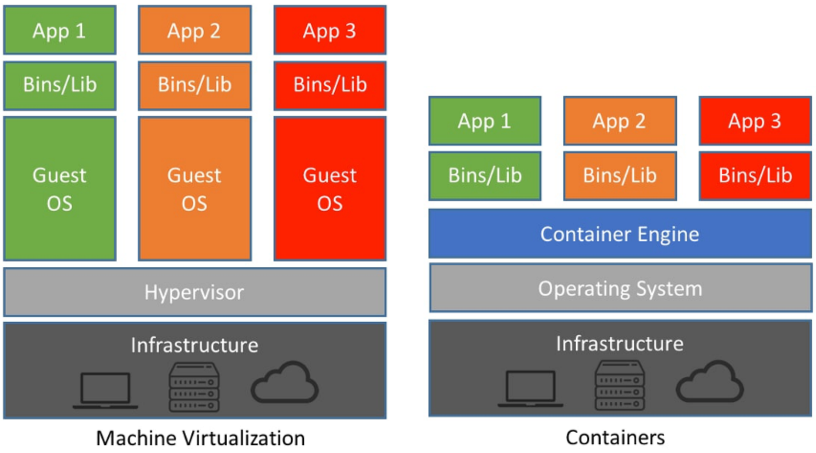


Figure 1.2: Virtual Machine Vs Container.

### 1.1.5 Virtualization

One of the most important concepts when dealing with cloud and edge computing is the virtualization. Virtual Machine (VM) are widely used to optimize hardware resource usage and are a consolidated concept. One limit of VM is that they virtualize every single aspect of a real machine, including the whole Operating System (OS), the kernel and virtual hardware associated to the machine. The virtualization technology has its own impact on the software running on the top of the virtualization stack [10] [11] [12] [13]. VM rely on an hypervisor. Hypervisors can be divided into 2 different families, known as type 1 and type 2. The former is installed directly on the bare metal hardware and directly access and manage hardware resources. It behave like a regular OS and usually exposes a web Graphical User Interface (GUI) or Application Programming Interface (API)s. The latter is a software running on an host operating system that acting as a middleware that translates OS level calls coming from the guest virtualized kernel to OS level calls to the host kernel. In opposition to the VM concept, containers take advantage of the reduced stack needed to run. A comparison between the two technologies is shown in Fig. 1.2. Isolated environments, called containers, share the same kernel of the host system leveraging the so called container engine to translate system calls into real calls to the underlying hardware. Containers have a faster access to Input/Output (I/O) operations and need less time to be run. The footprint in terms of storage memory needed to run a container is significantly less than the one needed to store the virtual hard drive of a virtual machine. Containers also represent a viable way to distribute Service-Based Architecture (SBA) and, in particular, are suitable to run the so called microservice architectures and cloud native architectures. In the above mentioned cases, orchestrating groups of containers is crucial to provide resiliency and auto-healing capabilities to the software being deployed. Kubernetes (K8s) is the defacto standard for container orchestration in industries and research contexts. Many container engines exist today. Docker is the most known container engine but does not implement the Container Runtime Interface (CRI). This pushed K8s to drop Docker as the default container engine.

### 1.1.6 Network softwarization and virtualization

The network softwarization process refers to the replacing of legacy network hardware with software-based solutions that may be run on general purpose Commercial off-the-shelf (COTS) hardware. One of the most important objectives of network softwarization is automatized and programmable network orchestration and management. Softwarization and virtualization they go together towards the network of the future, bringing a set of new functionalities and possibilities to the network operators. Everything is fluid and programmable, down to the lower layers with the aim to reconfigure complex network segments and data flows in seconds. This opens to new reliability and restoration scenarios in case of disasters or during scheduled maintenance time frames that may bring to zero-downtime. Network management tasks can be automated so reducing operational costs and improving network agility. Network scalability takes benefit from network softwarization and virtualization as Virtualized Network Function (VNF) replace bare metal hardware. Horizontal scaling in legacy hardware mean to buy new devices, to configure protocols, to let network converge and to test it. With Network Function Virtualization (NFV), scaling horizontally ideally means to instantiate a new piece of software that will run the work. Also the security gets benefit from network softwarization and virtualization as the update process of an existing VNF is a matter of distributing a normal software update, instead of installing new pieces of hardware. The network softwarization and virtualization enables fast deployment of new services and protocol stacks down to one day from ten years required in previous technologies due to hardware constraints (Fig. 1.3). All the mobile generations from the 1G to the 4G were not able to adapt to changes that involve architectural aspects. 5G aims to start the transition to the software. Having the most of network functions implemented on software, allows elasticity and should enable deployment of new network services in days instead of years needed by previous technologies.

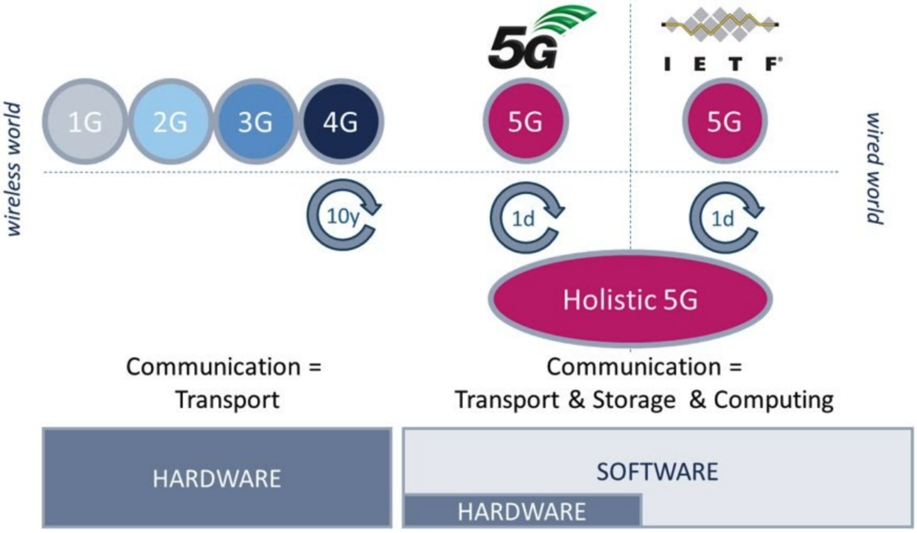


Figure 1.3: Evolution of communication networks.

### 1.1.7 Network Slicing

Network slicing is a central concept in networking as it support a huge variety of networks with different requirements sharing the same underlying resources. Network requirements usually come from application needs. A time-sensitive application may call for milliseconds or even under-millisecond Round Trip Time (RTT) to close the control loop and work properly. A high resilient application may call for very high resiliency and push the network to provide some kind of protection to the data plane traffic. Other application requires high peak throughput and so on. Luckly, the vast majority of such application does not need all those requirements at the same time. As an example, an automated arm robot may require very low latency at low data-rate while a video streaming service may require a very high throughput but is more tolerant to little delays and packet loss. The natural solution to this problem is to create many different physical networks which provide different QoS to different needs. It is easy to show that this solution does not scale well with the number of services that need different requirements. The network slicing solves this problem, enhancing the overall network efficiency through the resource reuse and sharing. The so created networks are logically isolated and using the same shared physical infrastructure. Software Defined Networking (SDN) and NFV play a crucial role in realizing an efficient network slicing in modern systems.

### 1.1.8 Multi-access Edge Computing

ETSI MEC is an ecosystem providing cloud-computing capabilities and infrastructure service and environments at the edge of the network. The aim of MEC is to enable uRLLC, real-time access to the access network information and high bandwidth to the end users. The main use cases for MEC are:

- Vehicle-to-everything (V2X)
- Video analytics
- Location services
- IoT
- Augmented and virtual reality
- Data caching

MEC is suitable for mobile, fixed, and Wireless Local Area Network (WLAN) access networks and produces an increment in network's KPIs for all of them. Currently, MEC is on its 'Phase 3' stage, where standards about MEC security, cloud and NFV Life Cycle Management (LCM), mobile and not reliable connected components and consumer-owned cloud resources [14]. The scope of ETSI is to create open and standardized service environments to host and support third party applications at the edge. MEC and NFV are concepts that can cooperate together. ETSI designed the MEC architecture to be deployed in many different configurations. One of the designed architecture variant allows MEC applications and ETSI NFV Management and Orchestration (MANO) components to be instantiated on the same Virtual Infrastructure Manager (VIM), reusing part of the MANO to orchestrate MEC itself. The MEC architecture reference architecture variant in Fig. 1.4 shows the relation betwen those components and the reference points used to communicate, divided into NFV, MEC, and MEC-NFV reference points.

## 1.2 State of the art

### 1.2.1 Motivation

The MEC architecture defined by ETSI poses a series of new challenges to the networking world. Legacy network architectures need complete re-designed or adaptation

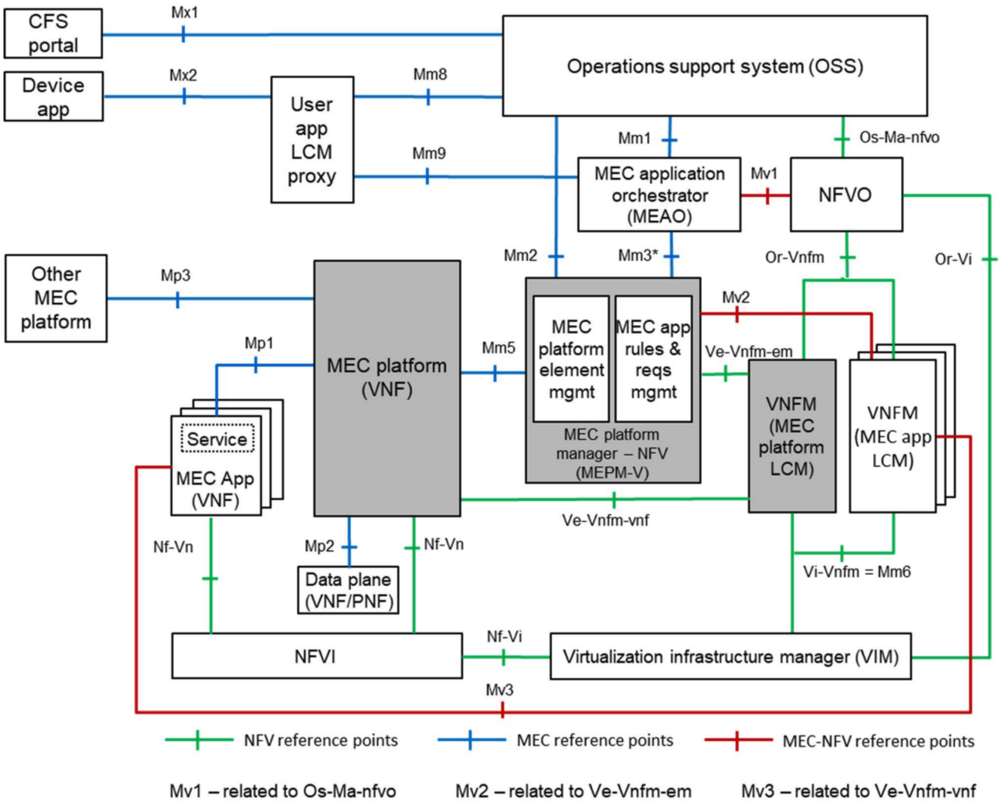


Figure 1.4: Multi-access edge system reference architecture variant for MEC in NFV.

of the existing architectural paradigms to match new objectives. Current transmission technologies (i.e., optical fibers) are able to transmit light at about 204 (m/µs), as shown in Table 1.1. This means that it is theoretically possible to cover around 204Km within 1ms one-way transmission through optical fibers. In a transmission originated by the end-user, a delay introduced by the Radio Access Network (RAN) or another user-side access technology should be added. Ignoring the RAN delay to transmit data, the 204 (m/µs) value is just an upper-bound for a real data transmission using optical fibers. Application-layer data transmission may be delayed at different Open Systems Interconnection (OSI) layers while traversing the network. Fig. 1.5 shows data communication from *Host A* to *Host B* following the International Organization for Standardization (ISO) OSI model. Each network device (i.e. Switch, Router, Server) needs to decapsulate and encapsulate back data into the proper protocol layer. Furthermore, Layer 3 (L3) packets may be queued at switch and router devices or they can be dropped generating a huge application layer delay. Optimizing network performances alone is not a viable way to reduce latency. Under a certain value, the only way to reduce application layer latency is to move phisically

Table 1.1: Optical Fiber Characteristics. Data from [15].

| Optical Fiber Type | Wavelength (nm) | Refractive Index | Distance (m/µs) |
|---|---|---|---|
| Brand A (G.652) | 1310 | 1.4677 | 204.260 |
| | 1550 | 1.4682 | 204.191 |
| Brand A (G.655) | 1550 | 1.468 | 204.218 |
| | 1625 | 1.469 | 204.079 |
| Brand B (G.652) | 1310 | 1.467 | 204.357 |
| | 1550 | 1.468 | 204.220 |
| Brand B (G.655) | 1550 | 1.470 | 203.940 |
| | 1625 | 1.470 | 203.940 |

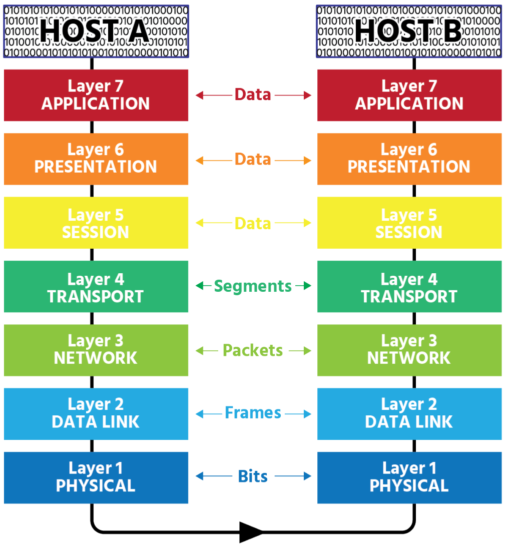


Figure 1.5: OSI Model.

the application near to the end user. This is exactly what ETSI MEC promises.

### 1.2.2 Telecommunication access network architecture

The current telecommunication access network architecture is composed by several components that interact together to provide communication between users or between users and requested services. The legacy access network architecture can not provide IP connectivity to the end users, so the IP packets generated at the edge by the users are encapsulated into a Layer 2 (L2) tunnel to reach centralized data-centers that provide higher level networking capabilities. This is a limitation to how deep an edge server can be deployed into the access network and it is something that may

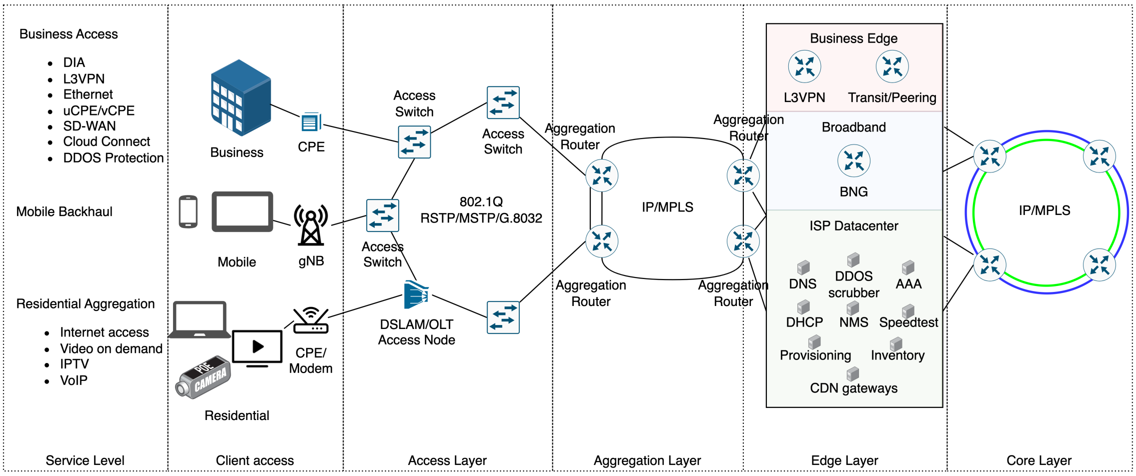


Figure 1.6: Telco Network architecture.

be addressed in the future as the latency requirements continue to push for lower latency. Also, a communication that involves two users accessing to internet from different telecommunication providers must traverse all the layers up to the core and then go back down to the client access layer. This limit is due to the current access network architecture.

Fig. 1.6 depicts a current deployment of a telecommunication operator's access network from its core to the client access layer. This architecture is replicated by each Internet Service Provider (ISP) operating in every geographic area. This means the shortest path between two users connected from different telecommunication operators need to go up to the Edge Layer and then go down again to the access layer of both operator's networks.

### 1.2.3 Software Defined Networking

SDN is an approach to networking that uses software-based controllers to communicate with underlying hardware infrastructure and control data-plane traffic on a network. The fundamental idea behind SDN is to decouple the control plane from the data plane and place an application layer that holds the entire control logic [16] [17]. Enterprises and carriers gain network programmability, automation, and network control. This enables them to build highly scalable and flexible networks able to react to changes in minutes instead of days required by legacy architectures. The aim of the control plane is to manage forwarding rules and is the intelligent part of the architecture, where decision are taken. A centralized SDN controller will get information from the bare metal SDN devices and applies the logic to take decision on each data flow. The aim of the data plane is to apply decision coming from the centralized controller into the network to properly forward packets. In SDN, network features like switching, routing and firewall that historically are shipped within hardware devices are now placed outside SDN devices. This is crucial in the SDN architecture as it gives more elasticity to the network itself and enable network programmability. Fig. 1.7 shows the three fundamental layers of SDN. There are two important interfaces: Northbound Interface (NBI) and Southbound Interface (SBI). The two interfaces enable respectively communications from the application plane and the control plane and communication from control plane and data plane. The SBI uses standardized protocols (i.e.: OpenFlow, NETCONF) to communicate to the underlying hardware layer. The SDN domain is a very huge topic, where a lot of research effort has been spent over the last years. One of the new direction in SDN is to control the PON to unlock the full potential of the deployed networks. It is worth noting that PON are currently yet in place in many countries all around the world and optimizing efficiency of such access networks may be crucial to provide a better QoE to the end users. There are some works in literature that use SDN to control the PON and provide Service Level Agreement (SLA) accordingly to requirements. In [18] the authors provide an innovative access network for Business-to-Business (B2B) use case where requirements may change over time. In [19] authors show how it is possible to apply fast protection exploiting FPGA-based Optical Line Termination (OLT) in Long-Reach Passive Optical Network (LR-PON). In [20] authors investigate with an experimental demonstration how it is possible to use an SDN controller to dynamically adapt the wireless bandwidth depending on cell load in a Long Term Evolution (LTE) over PON fronthaul with a sub-second E2E reconfiguration time. In literature

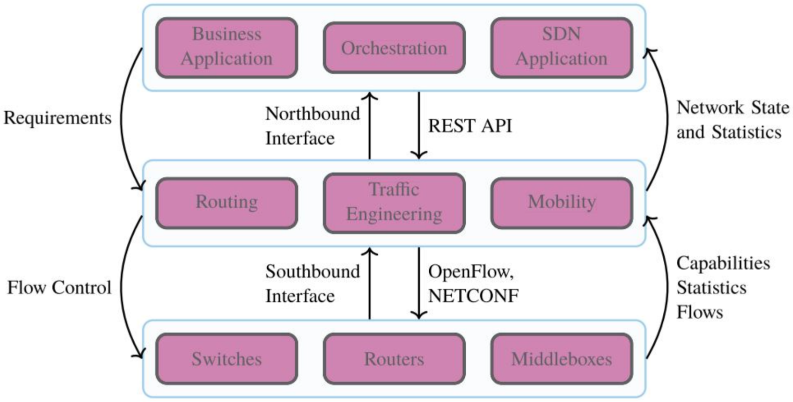


Figure 1.7: SDN Architecture.

there are very few works addressing real end-to-end SDN from the end user to the service. In this contexts new kind of interoperability problems need to be addressed. The operation time needed to bring up required resources in heterogeneous scenario is something that needs to be studied.

### 1.2.4 Network Function Virtualization

NFV is a paradigm that aims to decouple network functionalities from physical devices where they run. Each VNF should run on general purpose hardware [21] to provide system interoperability. This paradigm allows for flexible deployment of network functions, enabling simpler management and allocation of network resources for specific services. It will be possible to dynamically and flexibly adapt network resources according to service requirements. No longer having to worry about configuring specific network devices, i.e., what they are or are not capable of doing (for example, based on the manufacturer or the version of the operating system installed on them), provided by different vendors, increases the degree of freedom in creating, implementing, and managing network services. A precise service can be obtained by combining multiple VNFs, a commonly known operation called VNFs chaining. The network functionalities, being now implemented in software and executable on generic hardware, can be easily updated and modified to keep up with the evolving service requirements. It is interesting to note that with this strategy, it is possible to execute the VNF that implements a service closer to the user.

# Chapter 2

# Thesis contribution

This thesis investigates performance optimization focusing on orchestration and virtualization aspects to provide guaranteed E2E low latency and an increase of end-user QoE. The problem has been studied from the perspective of two main use cases that are low-latency and multimedia. On the head of those two scenarios, three main technologies have been investigated to achieve performance optimizations in terms of network KPIs or in terms of QoE and they are: PON, RAN, and Orchestration. Network slicing and orchestration have been used to support the technologies and to guarantee in the end an E2E QoE in access networks, serving multimedia or low latency use cases.

First, the ETSI MEC architecture has been studied. Then, networking aspects have been selected and the problem of optimizing performances on the access network has been studied by many perspectives. Video streaming and spatial audio performances as well as providing low latency to the end users have lead the research topics. The problem of resource orchestration has been studied and different solution for PON and heterogeneous access networks has been formulated and demonstrated. At the end, a fully functional access network composed by Open Radio Access Network (O-RAN) and PON providing connectivity from the end user to the 5G SBA has been deployed. Studies on orchestration, commissioning and decommissioning of dedicated network slices and network performance optimization have been studied on such complex network. The study on K8s container orchestration evidenced a lack of literature on latency concerns while scheduling pod resources on K8s clusters. Part of the work has been guided to close this gap and provide a seamless migration of resources to the network edge, per user. This is one of the first works in literature addressing latency in K8s and the first one providing a latency-aware scheduler for K8s.

Table 2.1: Technologies and use cases addressed by appended works. A check mark shows that a technology or a use case is considered into the respective paper.

| | **Technology** | | | **Use Case** | | |
|---|---|---|---|---|---|---|
| | **PON** | **RAN** | **Orchestration** | **Multimedia** | **Low-latency** | **Security** |
| **C1** | ✓ | | | ✓ | | |
| **C2** | ✓ | | ✓ | | ✓ | |
| **C3** | ✓ | ✓ | ✓ | | ✓ | |
| **C4** | ✓ | ✓ | ✓ | | ✓ | |
| **C5** | | | ✓ | | ✓ | |
| **J1** | | ✓ | ✓ | | | ✓ |
| **J2** | | ✓ | | ✓ | | |

Table 2.1 summarizes technologies and the use cases taken into consideration in the works appended to this thesis.

A first phase consisted in setting up an experimental testbed to accommodate experiments. Contribution to hardware and software stack decisions have been pushed to design and build the physical infrastructure and related virtualization layers hosting the laboratory. A network KPIs measurement process has been run over a commercial PON to collect data related to network's KPIs. It is worth noting that, during the COVID-19 pandemic we assisted an enormous degradation of network's KPIs in residential networks worldwide, especially during peak hours. That degradation was so huge during the very first year of pandemic that it was almost impossible to perform normal user's network activities. Watching movies on Netflix, Amazon Prime Video and other major video streaming platforms was not as expected. Even if it was an edge and sporadic case for networks, it raises a series of problems that should be addressed sooner or later to guarantee the future growth of the networks. Fig. 2.1 again clearly shows the problem from the end-user perspective. From the network side, the problem was that there was not enough bit-rate to sustain an acceptable video streaming flow for the end-users. The problem was not only related to video streaming but affected all the aspects of surfing on Internet (including browsing, gaming, video conferences, etc.). This issue has been studied and characterized in terms of measurable network's KPIs (i.e.: latency, throughput) over a time window of 24 hours. Data have been collected and stored for analysis and reuse. A software tool has been developed to read and reproduce the whole data set or a part of it. This allows us to make reproducible experimentation over time. Fig. C1.1(b) shows the trend line of the available bit-rate measured over the 24h time frame. The peak hours from 20:00 to 23:30 bring a huge drop in available bit-rate and an huge increase

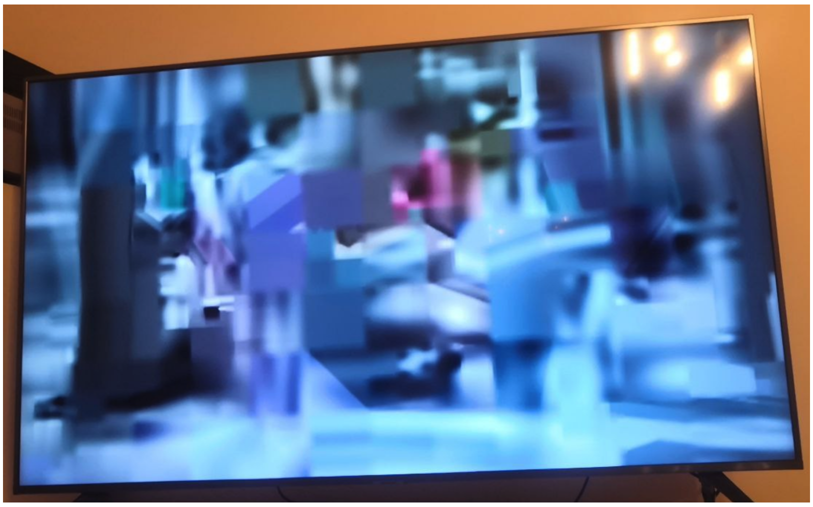


Figure 2.1: Video streaming (4K at the origin) reproduced through a 1Gbps (Downlink) 200Mbps(Uplink) commercial PON during the COVID-19 pandemic peak hours.

of the experienced RTT (up to 350ms). The measured values, accordingly to J2.1 do not allow even one single video streaming at an acceptable QoS and explain the image shown in Fig. 2.1.

Starting from this idea experimentation has been set up to show how different MEC node deployments can mitigate the problem.

Different aspects are considered spanning from video streaming performance evaluation on MEC to the orchestration of dedicated Virtual Local Area Network (VLAN)s to optimize network efficiency while granting required network KPIs for different access networks. Those concepts are studied in many different and incremental, really deployed scenarios up to reach a full-functional E2E slicing exploiting the O-RAN capabilities and the SDN principles to provide low latency slice commissioning and decommissioning on demand. Security aspects are studied as transversal and important topic and also a 5G MEC enabled scenario for a spatial audio in cultural heritage application scenario.

## 2.1 Publications appended to the thesis

This section presents the main submitted papers, appended to this thesis in Part II. A brief presentation of each contribution is provided per each appended paper.[1].

C1 C. Centofanti, A. Marotta, C. Rinaldi, F. Franchi, D. Cassioli and F. Graziosi, "Improved DASH Video Streaming Performance by MEC-Enabled Optical Ac-

[1] **J:**Journal,**C:**Conference, **B:**Book chapter.

cess", 2021 Asia Communications and Photonics Conference (ACP), Shanghai, China, 2021, pp. 1-3.

In this work, the priority is to improve Dynamic Adaptive Streaming over HTTP (DASH) performances in an access network composed by a PON connected to the cloud through internet. The link between the PON and the cloud constantly changes following the data set collected in a real environment over 24 hours. Different MEC node deployments are placed in the loop and the QoE is measured through a Quality Index parameter.

The three MEC deployments considered are: MEC at Cloud; MEC at OLT; MEC at customer premise. The MEC at customer premise is used as baseline for the optimal solution, not yet reachable from current technologies, that places a MEC server inside the customer Local Area Network (LAN).

Results show that we are able to improve DASH performances up to 5%, using MEC at OLT with respect to MEC at cloud, with the same network conditions.

***Author's contribution:*** The author came up with the idea, developed the data measurement software, the network emulator to reproduce data from the data acquired, deployed the DASH stack, performed the analysis of the results and wrote the manuscript.

C2 C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, N. Sambo, L. Valcarenghi, C. Bernard, H. Roberts, "Slice Management in SDN PON Supporting Low-Latency Services", 2022 European Conference on Optical Communication (ECOC), Basel, Switzerland, 2022, pp. 1-4.

In this paper, SDN is exploited to orchestrate a PON aiming to provide slices on demand. The AXOS E7-2 10-Gigabit-capable Symmetric Passive Optical Network (XGS-PON) solution from Calix has been adopted to provide a real physical infrastructure. The Calix OLT is capable to provide different class of services at the optical segment from the Optical Network Terminal (ONT) to the OLT. Through the implementation of a NETCONF SDN controller and a Low-Latency Service Manager it is possible to trigger the commissioning and the decommissioning of a dedicated network slice. This is really helpful to fulfill the SLA requirements of services that need very low latency while in function but can tolerate a little start-up time. To get affordable measurements on data-plane, an ad-hoc software tool has been developed together with a communication protocol.

Results show that reactive slice deployment can help in all scenarios where the application can afford a start-up time with sub-optimal performances. The deployment of a new slice through NETCONF requires about one tenth of a second to be forwarded to the Calix device and then the OLT requires about nine tenth of a second to start using the new rules at data plane. This means we are able to close the control loop in about 1 second from the application request to the moment the network slice has been created and the network starts forwarding packets within the requested low latency slice.

***Author's contribution:*** The author came up with the idea, designed and developed the glssdn controller, designed and developed the latency measurement protocol and related software, designed the overall architecture, contributed to the PON and OLT set-up, performed the analysis of the final results and wrote the manuscript.

C3 C. Centofanti, A. Marotta, V. Gundepu, D. Cassioli, F. Graziosi, H. Roberts, C. Bernard, K. Kondepu, "End-to-end Slicing of RAN based on Next Generation Optical Access Network", IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Gandhinagar, India, 2022.

In this work we moved the focus on heterogeneous network joint orchestration supporting 5G technologies. The 5G RAN has been integrated with the PON and the 5G SBA virtualized containers are placed back to the PON itself. Slicing mechanisms coming from previous works have been adapted to provide a low latency network slice and a best effort one. The management and orchestration framework has been used to orchestrate the mobile network controller, the optical access controller, and the network orchestrator. Measurements on network KPIs have been taken from user equipments connected to the 5G Next Generation Node B (gNB) to the SBA core component deployed on a general purpose server behind the PON.

Results show that our presented E2E dynamic slicing can work on a real implementation of RAN based on optical access network. The proposed slicing mechanism is able to reduce the mean experienced latency by 33% and significantly reduce the jitter so increasing network quality.

***Author's contribution:*** The author contributed to the idea, jointly set-up the experimentation with co-authors, collected results, performed the analysis of the results, and wrote the manuscript.

C4 C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, V. Gundepu, K. Kondepu, "End-to-End Slicing via O-RAN and Software Defined Optical Access", Optical Fiber Communication Conference (OFC), San Diego, California, USA, 2023.

In this paper an E2E strategy operating in O-RAN and PON context is shown. The Software Defined PON access controller, the O-RAN controller and the network orchestrator are managed and orchestrated by a slice manager component that takes decision based on real time metrics coming from O-RAN monitoring xApps. The control loop is closed on the slice manager that can take decision and commission low latency slices when the measured network's KPIs does not respect the SLA. This dynamism allows to re-distribute unused resources to other users in the PON, reserving them with fixed bandwidth allocation contention free mechanisms only when really needed.

Results show that in a real implementation of O-RAN, PON and 5G SBA it is possible to feed a slice manager module to take decision on the heterogeneous network and reserve E2E resources on demand. This helps increasing optical network efficiency keeping low to zero the packet loss ratio through a bandwidth adaptation mechanism.

***Author's contribution:*** The author contributed to the idea, developed the SDN controllers, contribute to general network design and configurations, collected data, perform analysis of the results, and wrote the manuscript.

C5 C. Centofanti, W. Tiberti, A. Marotta, F. Graziosi, D. Cassioli, "Latency-Aware Kubernetes Scheduling for Microservices Orchestration at the Edge", SUBMITTED TO IEEE International Conference on Network Softwarization (NETSOFT), Madrid, Spain, 2023.

In this work, the default orchestration mechanisms in a virtualized and container-based K8s environment have been studied. In literature, a gap in latency awareness inside the scheduling mechanism has been identified. When the K8s cluster is wide and composed by nodes that are not geographically close to each other, the experienced latency experienced by the end user accessing services may vary. Default scheduler does not take into account user's latency while performing scheduling decision of pods. This may generate network inefficiency and degradation of measured application latency. A custom K8s latency aware scheduler and de-scheduler system has been designed and developed to assure seamless pod migration to the nearest-most location to the end user, in terms of latency.

Many software component have been developed following the micro-service design patterns to implement desired functionality and their performance have been evaluated.

Results show that our latency aware scheduler outperform the default one and is able to move the pod containing the target service to the node with the lowest latency, with no service interruption exploiting K8s and custom scheduling procedures.

***Author's contribution:*** The author came up with the idea, designed the system, the latency aware scheduling algorithm, implemented the code, perform analysis of the results, and wrote the paper.

J1 W. Tiberti, E. Di Fina, C. Centofanti, A. Marotta, D. Cassioli, "Security over the O-RAN Inter-Controller Interface: a Blockchain-based Anti-tampering Scheme for Traffic Policies", SUBMITTED TO Journal on Selected Areas in Communications.

This work analyzes the current state of the art of the O-RAN alliance standards to find cybersecurity critical aspects and possible vulnerabilities and an anti-tampering scheme for traffic policies based on block-chain. A man in the middle attack is demonstrated to show that communication from non RT Radio Intelligent controller (RIC) and near RT may be compromised. Our proposed approach is evaluated through three experiments featuring two different scenarios and we show that security is enforced by the adoption of our presented schema.

The results show that the added complexity introduces a tolerable latency which remains under the limits defined by O-RAN RIC specifications for the A1 interface communications.

***Author's contribution:*** The author contributed with the idea of using blockchain to protect communications, contributed with the state of the art research and contributed on writing the paper.

J2 C. Rinaldi, F. Franchi, A. Marotta, F. Graziosi and C. Centofanti, "On the Exploitation of 5G Multi-Access Edge Computing for Spatial Audio in Cultural Heritage Applications," in IEEE Access, vol. 9, pp. 155197-155206, 2021, doi: 10.1109/ACCESS.2021.3128786.

In this work, a service for the improvement of cultural heritage experiences that leverages the 5G and MEC paradigms is presented. The computation and storage resources at the edge provide binatural spatial audio rendering on demand with reduced latency and jitter. This work demonstrates how 5G, together with MEC architecture can enable new guaranteed services to the end users and, in particular allows the spatial sound application to serve a huge number of users with none or few loss of QoS.

Results show that the MEC deployment at the edge is capable to serve more connected clients, optimizing the network performances and enabling offloading of computation from the end user to the network.

***Author's contribution:*** The author contributed with the idea, with the definition of the overall architecture, with the modeling of the scenario and contributed to the writing of the paper.

# Part II

# Appended Papers

# Conference C1

# Improved DASH Video Streaming Performance by MEC-Enabled Optical Access

C. Centofanti, A. Marotta, C. Rinaldi, F. Franchi, D. Cassioli and F. Graziosi

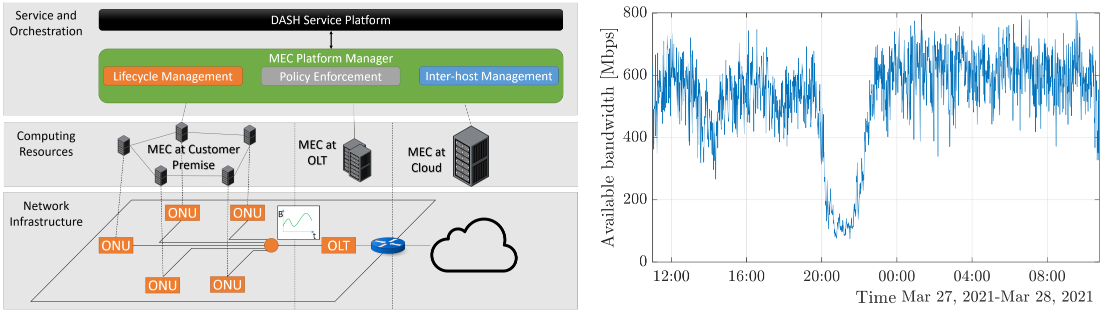
In proceedings of
*2021 Asia Communications and Photonics Conference*

# Abstract

We discuss how different service placement can improve DASH performances on an edge-enabled architecture in a MEC infrastructure. Network experiments emulate a real optical network behaviour allowing to measure improvements with different service placement.

## C1.1 Introduction

The growth of connection speed pulls for new network architectural models. One of the most promising technology is the so called Multi-access Edge Computing (MEC) [22, 23] architecture, which enhances network performance by making storage and computation capacity available at the edge. The reduced physical distance between end users and service location drastically lowers the latency [24]. The MEC technology represents a viable solution also for video delivering, which is the most consistent portion of nowadays Internet traffic. By 2022, video related traffic will be 79% of the whole Mobile data traffic and 82% of all consumer traffic generated in Internet [25]. Dynamic Adaptive Streaming over HTTP (DASH) is an enabling technology to distribute video chunks with different quality formats, finding the best trade-off between audio/video quality, network conditions and user's requests. The DASH, in combination with the MEC technology, can effectively address network-related problems, like, e.g., bandwidth shortage and high latency. In this work, we evaluate the performance improvement provided by a MEC-enabled optical access network on video streaming DASH services with respect to conventional cloud-based architectures. We run controlled network experiments by reproducing a real optical access network. The quality of the video transmission experienced by the users is evaluated and compared under different MEC deployment configurations. Our study evidences the advantages of MEC deployment at the Optical Line Termination (OLT) compared to both the deployment at the customer premise and the traditional deployment in a cloud regional provider.



(a) (b)

Figure C1.1: (a) The network scenario: a MEC-enabled PON architecture; (b) Throughput distribution over 24 hours, one sample per minute

## C1.2 System Model

We consider the reference architecture shown in Fig. C1.1a, composed by a *network infrastructure* and a few *computing resources* managed by a service and orchestration layer. The network infrastructure consists in a Time Division Multiplexed Passive Optical Network (TDM-PON).

MEC architectures are based on two main pillars: *Multi-access* and *Edge Computing*. The *Multi-access* technology connects different network nodes belonging to different access networks, enhancing network performances in the context of regional services. The *Edge Computing* in MEC will drastically lower latency [24] by reducing the physical distance between the end users and the network nodes where the service is running so enabling vertical services with stringent latency requirements.

Three different scenarios are considered in our study, i.e.:

- ***MEC at customer premise*:** the MEC node is inside the customer network. This configuration achieves maximum throughput and minimum latency for the users.
- ***MEC at OLT*:** the MEC node is inside or directly connected to the Internet Service Provider (ISP) access network, i.e., the user shares the optical resources with other active users connected to the Passive Optical Network (PON). This may reduce the available bandwidth while achieving a latency sightly lower than 1 ms. Fig. C1.1 shows the 24-hours trace of available bandwidth collected from a real commercial Gigabit-capable Passive Optical Network (GPON).
- ***MEC at Cloud*:** The MEC node is deployed at the regional *Cloud*, outside the ISP network. This is the worst-case scenario, with a reduced bandwidth and a high Round Trip Time (RTT) in the range of tens of milliseconds.

The Service and Orchestration Layer shown in Fig. C1.1a is composed by the *MEC Platform Manager* and the *DASH Service Platform*.

The *MEC Platform Manager* shown in Fig. C1.1a provides an uniform Application Programming Interface (API) to the upper layers. Its functions include:

- ***Lifecycle Management*:** this process takes care of the entire MEC application lifecycle, treating each MEC application instance as a Virtualized Network Function instance

- ***Policy Enforcement*:** it manages networks and applications connectivity, according to the network policies

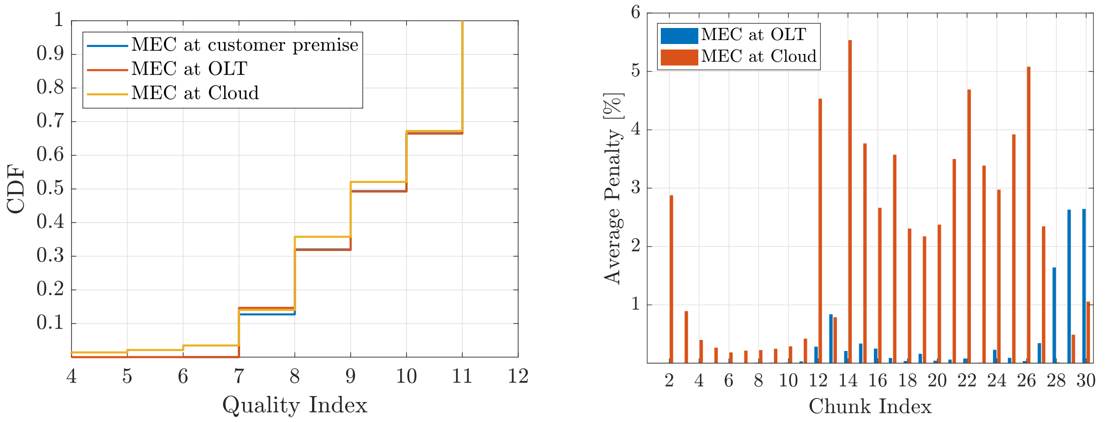
- ***Inter-host Management*:** this block is responsible of enabling networking and communications between different MEC hosts, exposing them to the *Service Layer*.

The *DASH Service Platform* handles the DASH service. The *MPEG-DASH* technique, provides adaptive bit rate streaming over simple Hypertext Transfer Protocol (HTTP) GET calls. Videos are parted into segments or chunks, with a fixed length in time and a fixed bit-rate. Each chunk reproduced by the player at the client side is characterized by a *Quality Index (QI)* associated to a video resolution and achieved bit rate. The QI assumed in our experiment span from 1 which is associated to a resolution of 320x180 px @254 Kbps up to 12 which corresponds to 3840x2160 px @30000 Kbps.

The client makes decisions about the quality of requested video chunks, by using an Adaptive Bit Rate (ABR) algorithm to monitor both the network parameters and the player status so optmizing the Quality of Experience (QoE) according to the current network conditions [26]. Several ABR algorithms have been proposed. The most relevant ones are *Bola* [27], and *Throughput*. *Bola* uses the buffer level indicator to estimate the best bit-rate for the next chunks. The *Throughput* algorithm bases its decisions upon the recent throughput history. We use a dynamic switch between those two to reach the best results [28].

## C1.3 Experiment Setup And Results

We set up three machine: a server, a gateway and a client. The server is responsible to store and serve the video content. The client continuously runs 5 parallel video playbacks for 24 hours. The gateway reproduces the network bandwidth and round trip time associated with the MEC scenarios described in Sec. C4.2, by running a *nodejs* daemon. The daemon fetches network data from our database and apply them to the network. For the purpose of our experiment, we use real data measured over a commercial GPON in the city of L'Aquila, Italy. The recorded network behaviour is then reproduced as follows: for the MEC at cloud scenario we apply recorded bandwidth and RTT; for the MEC at OLT we only apply bandwidth restriction; for the MEC at customer premise we consider 1 Gbps Ethernet connection.



(a) Experienced Quality Comparison

(b) Quality penalty over chunks

Figure C1.2: Results.

Fig. C1.2a shows the Cumulative Distribution Function (CDF) of experienced video *Quality Index*. As expected, the MEC at cloud scenario achieves the worst performance, due to higher latency and bandwidth limitations. This is explainable by the fact that the experienced throughput is inversely related to the RTT, which is maximum for this scenario, and at the same time is directly related to the available bandwidth.The MEC at OLT achieves comparable performance to the MEC at customer premise case which theoretically represents the best solution from the technical view point, achieving maximum bandwidth and minimum latency. This makes MEC at olt configuration an interesting compromise between costs of the infrastructure and performance.

By adopting the MEC at customer premise as the theoretically reachable target for the *Cloud* and *OLT MEC*, we define an *Average Penalty* metric to measure deviations from the adopted baseline. This allows to estimate the loss in terms of *Quality Index* drop. *Cloud* and *OLT MEC* average penalties are expressed as a percentage of the deviation from the baseline reference value. Fig. C1.2b shows that MEC at OLT performs 5% better with respect to the cloud case, in terms of QI.

## C1.4 Conclusion and future work

In this paper we have evaluated the impact of a MEC-enabled OLT in the context of DASH video streaming scenarios. The results show that with a Fiber To The Home (FTTH) GPON, it is possible to improve the overall video quality transmission placing a MEC node in the access network reducing RTT. Results show that the improvements given by the presence of MEC at OLT are comparable with the data obtained with the *MEC at Customer Premise* baseline. Comparing the OLT MEC with the *Cloud* the increase of *QI* indicator shows that it is possible to increase users QI placing resources into the optical access network. The *Cloud* performs significantly worse than the *MEC at Customer Premise* baseline. In future work we plan to investigate the impact of different ABR algorithms and PON dynamic bandwidth assignment strategies on the performance of MEC-based video delivery.

# Acknowledgement

This work has been partially supported by the H2020-MSCA-RISE OPTIMIST project (GA: 872866) and by the Italian Government under CIPE resolution no. 135 (December 21, 2012), project INnovating City Planning through Information and Communication Technologies (INCIPICT).

# Conference C2

# Slice Management in SDN PON Supporting Low-Latency Services

C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, N. Sambo, L. Valcarenghi, C. Bernard, H. Roberts

In proceedings of
*2022 European Conference on Optical Communication (ECOC)*

# Abstract

We study possible slice management strategies in software defined passive optical networks for low latency services. Our results show that reactive slice deployment is able to enforce latency requirements requiring a minimal setup time while increasing network efficiency compared to proactive strategies. ©2022 The Author(s)

## C2.1 Introduction

Software-Defined Networking (SDN) is a control architecture used to manage network operations in a centralized, elastic and efficient way [29]. In Passive Optical Network (PON), SDN can be used for network configuration and resource allocation based on requirements coming from Service Level Agreement (SLA). Moreover, network slicing has emerged as a paradigm to guarantee targeted performance to a variety of services sharing the same physical infrastructure [30].

Due to their high capillarity, PONs represent an interesting supporting infrastructure for a variety of services with heterogeneous requirements, e.g. multimedia [31], IoT, mobile networking, and critical services. Thus, the adoption of network slicing in PONs may be beneficial from both business and network efficiency viewpoint.

Among the protocols that have been proposed for the control of physical network infrastructures, NETCONF [32,33] has found wide application due to its capability to aid automated configuration of heterogeneous network devices, e.g. optical network devices, packet switching nodes, and virtual and physical machines. Such wide range of applicability allows to enforce end-to-end slice performance [34, 35] and enables network automation.

It is worth considering that service requirements and traffic conditions may vary over time leading to inefficient usage of network resources or to violation of SLA. Thus, network automation mechanisms for slice management become crucial in PONs. While static configuration of network settings, such as optical access bandwidth allocation policies, [36–38] is a well known problem in the literature, there is a gap on studying how to automate slice management in SDN-enabled PONs and evaluating the impact of slice management when slice requirements change over time or are dynamically activated/deactivated. This is a common scenario in Low Latency Services (LLS) where ad-hoc slice resource reservation may be triggered by either the occurrence of a critical event (e.g., a seismic wave in earthquake early warning or other alert conditions) or by the initiation of a Extended/Mixed Reality (XR/MR) transmission requiring high throughput and low latency.

In this work we provide an overview of different slice management mechanisms and perform an experimental evaluation of the impact in terms of slice deployment time and user latency of deploying an on-demand slice in a real PON in automated manner via NETCONF.

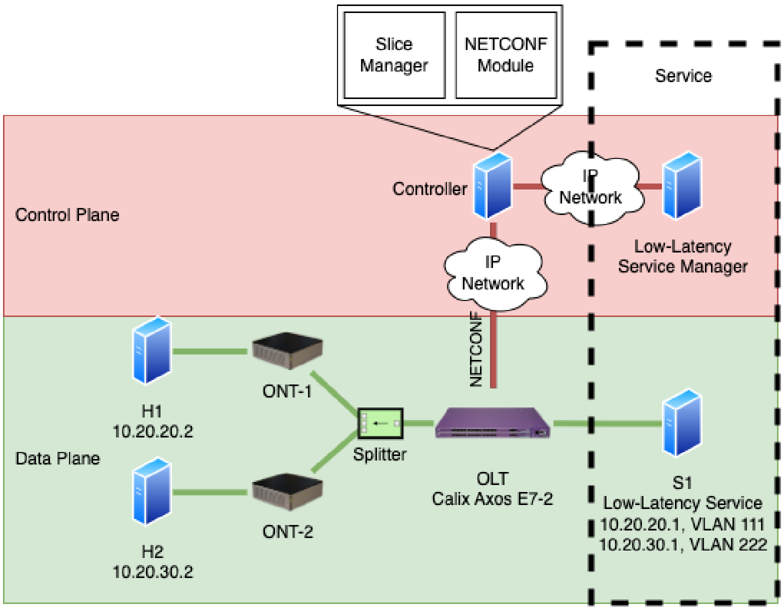


Figure C2.1: System architecture

## C2.2 System Description

We consider the system architecture shown in Fig. C2.1 where multiple users are connected to a data network through a Time Division Multiplexed PON (TDM-PON) at the data plane while the control plane is based on SDN and NETCONF. The hosts at the ONT side may request multiple services with heterogeneous requirements. We introduce a Low Latency Service Manager (LLSM) as a component of a low latency service architecture which is in charge of requesting network slices to the SDN controller. After receiving a slice request, the controller validates the request and then sends the right configuration to the Optical Line Termination (OLT) through NETCONF, to deploy the slice. In PONs, different bandwidth allocation policies can be selected in order to meet service requirements, such as: (i) Expedited Forwarding which gives to a particular ONT the possibility to utilize a reserved amount of bandwidth in a Grant-free fashion in order to reduce experienced latency; (ii) Request-Grant based access which leverages Dynamic Bandwidth Assignment procedures to exploit statistical multiplexing and increases network efficiency at the price of a larger experienced latency. The steps needed for the slice deployment in the considered architecture can be summarized as:

1. The LLSM sends the slice request to the controller.

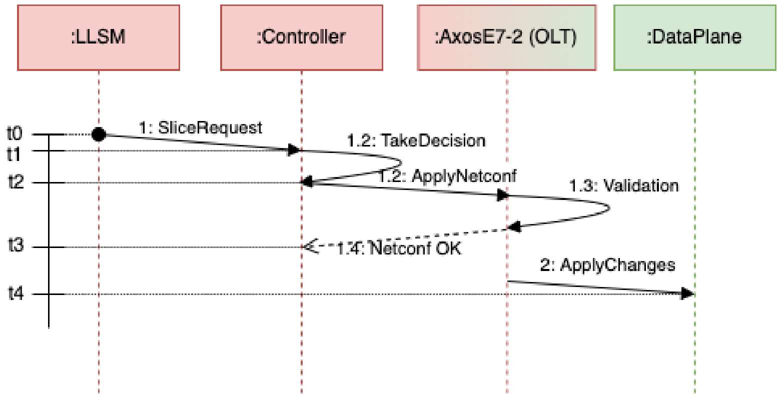


Figure C2.2: Communications between different elements. On the left a timeline shows references for time measurements.

2. The controller validates the incoming message and builds the NETCONF message for the OLT.

3. The controller sends the NETCONF message to the OLT through a Remote Procedure Call (RPC) over a pre-established SSH connection.

4. The controller receives an acknowledgment message from the OLT, i.e. a NETCONF OK message.

5. The configuration is applied at the PON data plane.

We consider the OLT to be directly connected to a Server which represents an Edge node able to host service instances to reduce end-to-end latency experienced by service users.

As latency represents a critical aspect, we consider the following events to identify different time contributions to measure slice deployment delay and perform relative comparisons among slice management strategies (a graphical overview being given in Fig.C2.2):

- $t_0$: The reference starting time, the time at which the slice request is issued, e.g. an alert condition is triggered.

- $t_1$: The time at the which the slice request message reaches the controller, influenced by network delay.

- $t_2$: The time at the which the controller has taken decision and sends a NETCONF message to the OLT.

- $t_3$: The time at which the controller receives a NETCONF OK message from the OLT.
- $t_4$: The time at the which new configuration is applied to the data-plane.

The overall Slice Deployment Time ($T$) is defined as $T = t_4 - t_0$.

## C2.3 Slicing Management Techniques

Different deployment strategies can be adopted to enforce slice requirements at the data plane. We consider the three following slicing management policies and show how they perform in real network conditions.

**Request-Grant Committed Information Rate** (RG-CIR) is a proactive strategy guaranteeing a given bandwidth to the users of the slice. The access to the bandwidth is based on a request-grant mechanism: the slice traffic is buffered and queued at the ONU until a transmission opportunity is granted, so increasing end-to-end latency.

**Proactive Low-Latency Reservation** (PLLR) uses EF to guarantee low latency and traffic prioritization at the OLT's scheduler. As a proactive strategy, resources are statically reserved before $t0$. This leads to an over-provisioning of resources but guarantees low-latency without waiting any slice deployment time.

**Reactive Low-Latency Reservation** (RLLR): when the LLSM requests low latency, the SDN controller dynamically changes the OLT configuration from normal request-grant to EF seamlessly. When the LLS wants to release resources, the controller reverts the configuration seamlessly, dismissing the slice and restoring request-grant conditions. This slice management policy can be easily adapted to a pay-as-you-go [39] model so reducing costs for the LLS provider.

Table C2.1: Mean time measures to instantiate and destroy a RLLR slice. Values in $\mu$s.

| | **Slice request transmission** | **Controller Computation** | **NETCONF communication** | **SDT** |
|---|---|---|---|---|
| | $t_1 - t_0$ | $t_2 - t_1$ | $t_3 - t_2$ | $T$ |
| **Deployment** | 93 | 70 | 125834 | 1032081 |
| **Decommissioning** | 82 | 31 | 87997 | 1001638 |

## C2.4 Results

To perform experimental evaluation we adopted the architecture shown in Fig.C2.1. The physical infrastructure is composed by a Calix E7-2 XGS-PON OLT running AXOS platform, two Calix 801XGS ONTs offering connectivity between two client PCs (H1 and H2) and one edge server (S1). Then, we implemented a Python-based NETCONF controller. Fig. C2.4 shows an excerpt of a NETCONF message sent to the OLT to enforce EF resource reservation.

We implemented a simple service able to perform time measurements at application layer through a time-stamped-based mechanism and deployed an instance at the edge node S1. In order to have a common reference time between the application layer, the LLSM and the SDN controller we deployed all this element on the same physical machine. In the considered scenario H1 generates service data toward the edge instance of the low-latency service under consideration, while H2 is utilized to generate secondary service traffic concurring in optical resource usage in the PON.

First, we perform a comparison of the latency achievable through a request-grant approach (RG-CIR) and expedited forwarding slice reservation (PLLR and RLLR). Fig.C2.3 shows average packet latency and standard deviation experienced by H1 (low latency traffic) and H2 (secondary service traffic) when a RG and EF slice resource reservation are implemented. Results show that while both approaches are able to enforce bandwidth requirements, only EF is able to offer latency level around 0.1 ms which represent more than 85% latency reduction compared to RG baseline.

RLLR and PLLR differ in the fact that in case of RLLR a time is needed to perform slice deployment and enforce slice requirements. Thus, we study the different contributions to the slice deployment time $T$. Table C2.1 defines and shows the different latency contributions. The first row in Table C2.1 is a measurement of time needed to enable RLLR while the second row is the fallback process from RLLR to no reservation for the slice. It is worth mentioning that the slice request transmission time and NETCONF communication time are strictly related to the physical deployment of the LLSM and the SDN-Controller. The considered co-location scenario allows to perform time synchronization among architectural elements. Results show that RLLR requires around $1s$ for slice deployment but compared to PLLR avoids to reserve optical resources for the low-latency slice when traffic is not transmitted, by reducing capacity over-provisioning.

## C2.5 Conclusions

In this paper we presented three different Slice Management strategies in SDN PON supporting Low-Latency Services. We implemented a NETCONF-based SDN controller which realizes slice management over a commercial PON infrastructure. We show that it is possible to use reactive strategies in cases where LLS does not request instantaneous low latency communications. The slice deployment time for reactive case may represent a very low time for most human-centric services but may not be not suitable for extreme safety scenarios and other critical applications.

## C2.6 Acknowledgements

This work was supported by the Italian PRIN 2017 project Fiber Infrastructure for Research on Space-Division Multiplexed Transmission (FIRST).

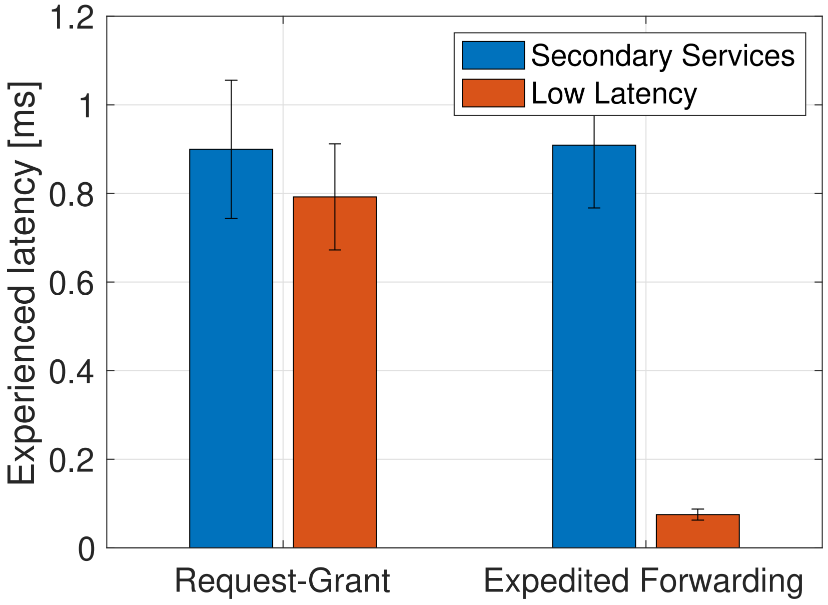
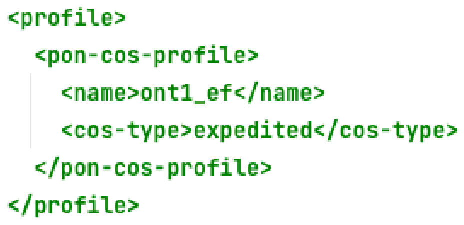


Figure C2.3: Latency differences between RG on the left and EF on the right.



```
<profile>
  <pon-cos-profile>
    <name>ont1_ef</name>
    <cos-type>expedited</cos-type>
  </pon-cos-profile>
</profile>
```

Figure C2.4: Excerpt of XML configuration to enable EF

# Conference C3

# End-to-end Slicing of RAN based on Next Generation Optical Access Network

C. Centofanti, A. Marotta, V. Gundepu, D. Cassioli, F. Graziosi, H. Roberts, C. Bernard, K. Kondepu

In proceedings of
*2022 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*

# Abstract

Due to their large diffusion, optical access networks represent a viable supporting infrastructure for mobile networks and services. The heterogeneity of services supported by mobile networks calls for the implementation of slicing mechanisms able to accommodate resources in all the involved network segments from the mobile user up to the core network.

In this work, we demonstrate a fully-functional and integrated 5G network deployment to satisfy real end-to-end slicing in next-generation access networks. We evaluate the impact of optical access network resources allocation mechanisms on slice performance in terms latency and jitter experienced by mobile users.

## C3.1 Introduction

Telecommunication networks have a distributed, shared architecture that is designed to forward traffic from one location to another. With the well-known mobile generations from first to fifth (1G to 5G), mobile networks have been experiencing a rapid technological advancement for more than 40 years. The current generation (i,e., 5G) is being implemented globally to replace the 4G specification and open up entirely new use cases for mobile networks that were previously restricted to landline connections. The flexibility that 5G technology brings introduces new issues that need to be solved. One of them is the next-generation Node B (gNB) high density required to cover uRLLC and eMMTC use cases.

To satisfy 5G network service demands such as dynamic Quality of service (QoS) requirements, operators must boost network capacity by deploying more RAN components (gNBs), which raises Capital Expenditure (CAPEX) and Operational Expenditure (OPEX), or by sharing the RAN, which allows operators to share CAPEX and OPEX expenditures and also enhance the mobile coverage [40]. RAN sharing can be defined as an approach in which the antenna, tower, power, mast, and backhaul equipment for the access network are shared by more than one network operator to provide different vertical services [41].

Network slicing is among the most essential components of 5G, provides a wide range of applications with different service demands through network virtualization [42]. Network slicing is the establishment of dedicated virtual network functions (VNF) with capabilities unique to the service or client across a shared network infrastructure [43] and it is realized with the support of Network Function Virtualization (NFV). NFV offers these VNFs with a customizable, adaptable, and modular network environment [44–46].

The 5G Core (5GC) exploits its Service Based Architecture in a ”cloud-native” manner. Slices are realized by NFV under the Network Slice Selection Function (NSSF) control. Each 5GC slice can be made up of a group of 5G core VNFs that are linked together to support a particular use case. One of the key features of 5G is Control and User Plane Separation (CUPS). It allows a 5G core system to be split into two parts: the Control Plane (CP) and the User Plane (UP). The CP is used as a common slice, whereas the UP is split into multiple customised slices with different bandwidth and latency requirements. The only UP 5G network function is User Plane Function (UPF) which is in charge of establishing and sustaining Packet Data Unit (PDU) sessions to route and send data packets to different external Data Networks

(DNs). 5G UPFs can be sliced to accommodate different services in order to meet diverse QoS requirements.

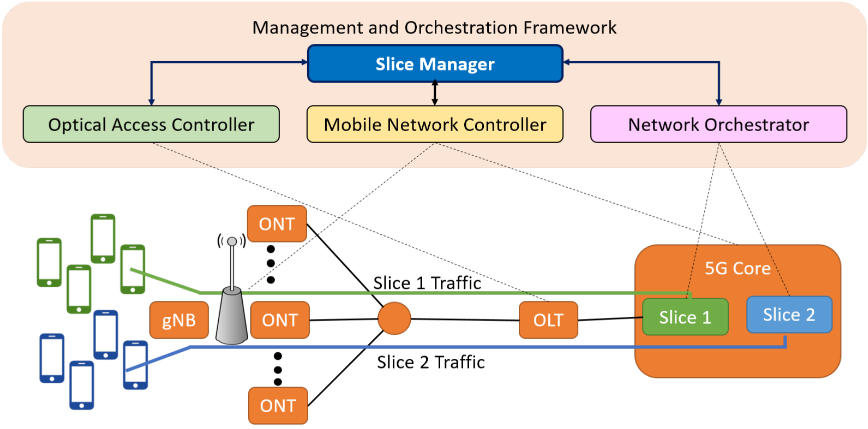


Figure C3.1: Network Architecture

Due to their high capillarity, Passive Optical Networks (PONs) represent an interesting supporting infrastructure for a variety of services with heterogeneous requirements (e.g., multimedia) [31], IoT, critical services, and nonetheless mobile networking. Thus, the adoption of network slicing in PONs may be beneficial from both business and network efficiency viewpoint. In [47], the authors highlight the issues that may occur in a shared access network serving a 5G Radio Access Network (RAN). A method to compute optimal virtual PON (vPON) slice configuration supporting uRLLC scenarios is proposed in [48] while in [49] slice management strategies in PONs via NETCONF are illustrated. In [50] the virtualization and isolation problems in RAN are studied by the authors. They setup a testbed to deploy dynamic and isolated RAN slices to deliver end-to-end slices to the end users, however they only manage to run a LTE testbed and not considered slicing with PON.

Integrated management of radio and optical access networks represents a viable solution to increase network efficiency, especially in slicing scenarios as discussed in [51]. The purpose of this paper is to demonstrate feasibility of a fully-functional and integrated 5G network deployment to satisfy real end-to-end slicing in next-generation access networks.

## C3.2 System Model

We consider the reference architecture shown in Fig. C4.1 composed by three network segments, i.e., the RAN, the Optical Access Network (OAN), and the 5G Core. The

OAN is a PON and is adopted as backhauling infrastructure for next-generation Node Bs (gNBs). The gNB is equipped with an Optical Network Termination (ONT) and the PON is also utilized as Fiber To The X (FTTx) infrastructure for other services such as residential and business connectivity and Internet of Things (IoT).

In order to deploy end-to-end slices, resources in the three considered network segments, have to be jointly allocated. To achieve such integrated resource allocation we consider a Management and Orchestration Framework which includes software defined controllers for the optical access and mobile networks and a network orchestrator. A Slice Manager element is responsible to map requirements of different slices into ad-hoc configurations of the network segments and to coordinate the network controllers and the network orchestrator throughout the slice life-cycle.

The network controllers are responsible to implement resource allocation strategies able to meet the desired slice performance. At the mobile core segment the network orchestrator is responsible to deploy 5G core functions such as Access Management Function (AMF), User Plane Function (UPF), Network Repository Function (NRF), and Session Management Function (SMF) that constitute the slices at mobile core level. This also allows to place UPF in different physical locations, according to the Service Level Agreement or the user's requirements. An UPF deployed at the edge may guarantee very low latency capable to serve ultra reliable and low latency communication (uRLLC) while an UPF deployed at central office or at a cloud provider may help cutting down CAPEX and OPEX at the cost of increased latency.

As the OAN may be shared with other customers (i.e., residential customers) isolation on the optical link is crucial to provide desired SLA for the service running at 5G network. 5G traffic on that segment includes user's data and control traffic to manage the radio network so demanding for isolation and prioritization. Such traffic differentiation is achieved at the PON by adopting different bandwidth allocation strategies such as: (i) Request-Grant based access which leverages Dynamic Bandwidth Assignment procedures to exploit statistical multiplexing and increases network efficiency at the price of a larger experienced latency; (ii) *Expedited Forwarding* which gives to a particular slice the possibility to utilize a reserved amount of bandwidth in a Grant-free fashion in order to reduce experienced latency.

## C3.3 Experimental Setup and Results

Figure C3.2 shows the considered experimental setup primarily consisting of three different parts that are the RAN, OAN and CN.

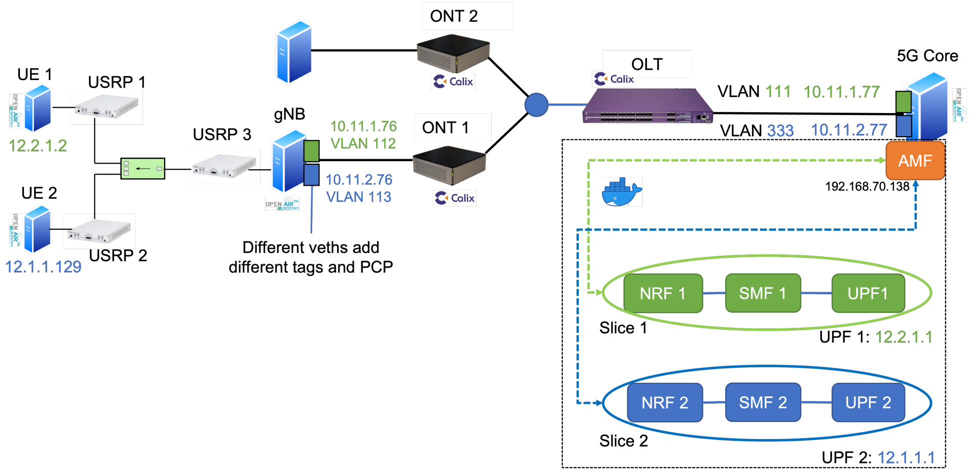


Figure C3.2: Experimental Setup

### Radio Access Network (RAN)

The RAN is deployed by using OpenAirInterface (OAI) open-source RAN software stack [52]. Then we deployed two OAI-based UEs (*UE 1* and *UE 2*) using two different physical machines. The UEs are connected to the RAN by using National Instrument X310 Universal Software Radio Peripherals (USRPs). The UE side USRPs are connected with the gNB side USRP through a two-way signal splitter/combiner with SubMiniature version A (SMA) connectors as shown in Fig. C3.2. This allows to evaluate multi-user connection with the single gNB.

On the other end, the OAI gNB is configured with Single – Network Slice Selection Assistance Information (S-NSSAI) list to support two different Slice/Service Type (SST) and Slice Differentiator (SD) values to evaluate the RAN slicing scenario.

### Optical Access Network (OAN)

The OAN segment in our setup is deployed in between the RAN and CN. It is realized by utilizing commercial Calix Axos E7-2 NG-PON2 that provides a configurable NETCONF API useful to configure isolation and network traffic rules at runtime to implement dynamic slicing. To provide the traffic differentiation and isolation at the forwarding plane we implemented a slicing mechanism based on IEEE 802.1Q Virtual Local Area Networking (VLAN). The 802.1Q standard also specifies a 3 bit Priority Code Point (PCP) to select traffic type. It is worth noting that 802.1Q does not specify how to implement traffic type differentiation based on PCP.

The Calix Axos E7-2 is able to instruct the ONT to differentiate traffic based on VLAN tags and PCP. This way, services that needs high priority and low latency can use VLANs to get their Service-Level Agreements (SLAs) fulfilled. We configured the OAN to treat the traffic belonging to Slice 1 (associated to VLAN 112 and PCP 7) as *Expedited Forwarding* —- to be transmitted from the ONT to the OLT without applying request-grant mechanism in order to reduce latency. The traffic belonging to Slice 2 is associated with VLAN 113 and PCP 3 and is transmitted in a *Best Effort* manner —- via traditional request-grant based on dynamic bandwidth allocation.

The mechanism to ensure low latency to a specific slice through the whole network is as follows:

1. The gNB applies a VLAN tag and PCP to each packet going towards to the 5G CN, for each UE, by utilizing virtual ethernet interfaces.
2. The ONT uses the PCP to apply forwarding policies such as *Expedited Forwarding* or *Best Effort*
3. The traffic flows through the OAN to the 5G CN, where the CN slicing contains two different User Plane Functions (UPF) associated with two different VLAN tags as shown in Fig. C3.2

The forwarding policies at the OLT can be configured in a dynamic way with help of Slice Manager (NETCONF) as shown in Fig. C4.1.

### Core Network (CN)

The OAI-based 5G Core is implemented in a scalable manner that makes it simple to adapt to the requirements of various 5G use-cases [53]. In this, different Network Functions (NFs) are implemented to provide services —- the Control Plane (CP) and User Plane (UP) functions are separated to enable independent scaling. Here, Network Slice Selection Function (NSSF) decides on the acceptable NSSAI, and sets Access and Mobility Management Function (AMF) to serve the UE. As shown in Fig. C3.2, each Slice is associated with three different NFs — Network Repository Function (NRF), Session Management Function (SMF), and User Plane Function (UPF) — and their selection is mainly based on the S-NSSAI with the help of NSSF.

The container (Docker) based 5G Core with all the required NFs are deployed, specifically two UPFs (i.e., Slice 1 and Slice 2) are associated with the two UEs while running the experiments to guarantee isolation and enhance slice performance.

In order to evaluate the impact of different resource allocation strategies of the OAN on the slice performance we perform delay measurements: (i) between gNB and Core (i.e., optical access delay); (ii) between the UEs and Core (i.e., end-to-end

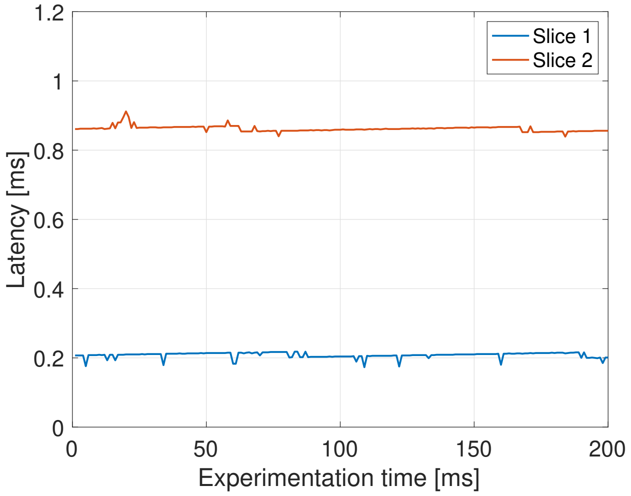


Figure C3.3: PON latency for different slices

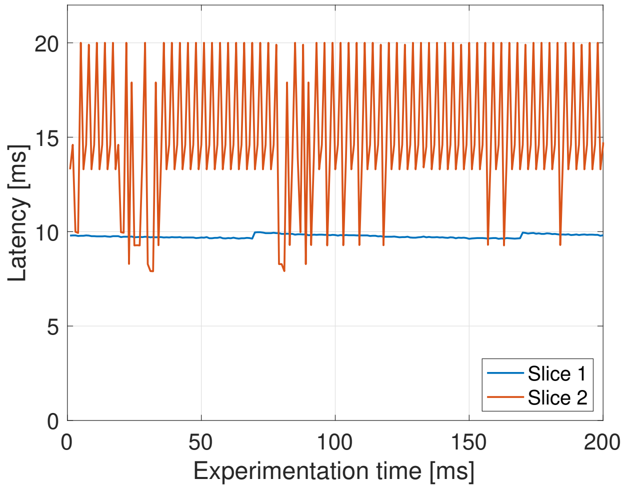


Figure C3.4: UEs to Core (UPFs) latency for different slices

delay). To perform round trip time measurement we send ICMP packets with a ping interval of 10ms.

Fig. C3.3 shows the optical access delay measured between gNB and Core for the two slice under consideration. It can be noticed that *Expedited Forwarding* allocation strategy adopted for Slice 1 allows to reduce the delay in the PON by the 80% compared to the Slice 2 which is served in a *Best Effort* strategy. In fact, the optical access latency measured for Slice 1 is 0.25ms average while the one measured for Slice 2 is around 0.85ms. Such reduction is due to the absence of request-grant procedure for Slice 1 whose transmission opportunities in upstream are reserved in advanced allowing the ONT to transmit in a grant-free fashion, thus, reducing queuing of packets at the ONT.

Fig. C3.4 shows end-to-end latency measurements for the two slices. As can be noticed, the end-to-end delay measured for Slice 1 is significantly reduced compared to Slice 2. Such delays include contribution of the RAN together with the one of the OAN. Fig. C3.4 shows that the improvement of performance introduced by resource reservation in the optical access network can be observed not only in terms of latency reduction but also in terms of end-to-end jitter reduction. This is explainable by the fact that packets belonging to Slice 1 are not queued at the ONT. This significantly reduces delay variations which can be up to 10ms as measured for Slice 2.

## C3.4 Conclusion

In this paper we presented a real implementation of an end-to-end slicing of RAN based on optical access network. We utilize a commercial NG-PON2 for the backhauling of the mobile traffic and deploy container based sliced core network elements. We evaluate the impact of the proposed VLAN and PCP based slicing of the optical access segment. Results show that the proposed slicing mechanism is able to significantly reduce the experienced latency while offering reduced jitter to mobile users belonging to a specific slice.

## Acknowledgment

This work has been partially supported by the H2020-MSCA-RISE OPTIMIST project (GA: 872866) and by the Italian Government under CIPE resolution no. 135 (December 21, 2012), project INnovating City Planning through Information and Communication Technologies (INCIPICT). This work is also partly supported by DST SERB Startup Research Grant (SRG-2021-001522).

# Conference C4

# End-to-End Slicing via O-RAN and Software Defined Optical Access

C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, V. Gundepu, K. Kondepu

In proceedings of
*2022 Optical Fiber Communication Conference (OFC)*

# Abstract

We propose an end-to-end slice management strategy which exploits the programmability of O-RAN and software defined optical access. Advantages in terms of user experienced latency and packet loss are experimentally evaluated.

# C4.1 Introduction

The huge increase of data traffic calls for the densification of high capacity and heterogeneous mobile and optical network infrastructures. To cope with such a heterogeneity, a vendor-agnostic and software-defined approach for the management and control of the network is essential. Open-RAN (O-RAN) architecture represents a key enabler to achieve vendor interoperability and to implement flexible and intelligent control of mobile networks, which is crucial to meet the ambitious requirements of next generation services [54].

The *openness* of O-RAN paves the way towards the cooperation between RAN and optical transport network infrastructures, which has been shown to offer advantages in terms of network efficiency and savings in cost of ownership for operators [55]. In particular, Passive Optical Networks (PONs), due to their large capillarity, represent a good transport technology to support mobile networks densification. Furthermore, the introduction of Software Defined Optical Access Networks (SDOANs) allows to implement flexible bandwidth allocation strategies in a dynamic and programmable way [51].

*Network slicing* has emerged as a key technology to offer dedicated virtual networks with performance tailored to specific service or customer over a common network infrastructure. Slice deployment requires the accommodation of *computational resources*, which are represented by Virtual Network Functions (VNFs), and *network resources* as well. In order to offer the targeted performance to the user belonging to a specific slice, all the involved network segments (e.g., RAN, optical access and metro networks, mobile core network) have to be configured accordingly. Recent efforts addressed resource allocation strategies for sliced PONs [56] and SDN-based slice management via NETCONF in PONs [49] but, to the best of our knowledge, no study investigated so far the utilization of O-RAN and SDOAN for end-to-end slicing.

This paper aims to present an end-to-end slice management strategy which leverages the software controllability of O-RAN and SDOAN. A slice manager element interacts with a purposely built xApp in the O-RAN and dynamically adapts resource allocation in the PON to offer low latency and guaranteed bandwidth for specific slices at the backhaul level via NETCONF. The adaptation of reserved bandwidth at the PON avoids both the over-provisioning and under-provisioning of resources in the optical access and increases the network efficiency.

## C4.2 System Model

We consider the network architecture shown in Fig. C4.1 where two slices are deployed in a TDM-PON backhauled O-RAN, with *Slice 1* requiring low-latency for connected users and *Slice 2* with best-effort latency requirement. The O-RAN is represented by a single O-gNB controlled by a Near Real Time (Near-RT) Radio Intelligent Controller (RIC) which is responsible to apply decisions and perform monitoring on a millisecond time scale via the standardized E2 interface. The Near-RT RIC interacts with a Non-RT RIC which provides policies with a time scale $\geq 1$ s. On top of the near-RT controller, custom-built applications called *xApps* implement the control logic desired by the network operator via Application Programming Interfaces (APIs) exposed by the near-RT RIC.

A network orchestrator is responsible to deploy core network functions composing the slices at core level, i.e. the Access Management Function (AMF), Network Repository Function (NRF), Session Management Function (SMF), and User Plane Function (UPF). A Slice Manager element is responsible to map requirements of different slices into ad-hoc configurations of the network segments and to coordinate the network controllers and the network orchestrator throughout the slice life-cycle.

A PON controlled via SDN and NETCONF is adopted as backhaul infrastructure for the O-RAN; a VLAN-based traffic differentiation mechanism is implemented to identify backhaul traffic belonging to different slices and adopt bandwidth allocation strategies basing on targeted slice performance. We consider the following bandwidth allocation strategies: (i) *Expedited Forwarding* which gives to a particular slice the possibility to utilize a reserved amount of bandwidth in a Grant-free fashion in order to reduce experienced latency (adopted for *Slice 1* traffic); (ii) *Request-Grant based access* which leverages Dynamic Bandwidth Assignment (DBA) procedures to exploit statistical multiplexing, thus increasing the network efficiency at the price of a larger experienced latency (adopted for *Slice 2* traffic).

Although the Expedited Forwarding offers low latency, it requires dedicated bandwidth which may lead to inefficient resources usage when slice traffic varies. To preserve resource efficiency while offering low latency, we design a slice manager element which leverages O-RAN and SDOAN programmability to dynamically adapt the amount of reserved low-latency optical access resources. As shown by the sequence diagram in Fig. C4.2a, the slice manager element is responsible to collect slice traffic statistics from the *Slice Monitor xApp* which runs at the near-RT RIC and obtains the instantaneous slice traffic load at PDCP layer. The slice manager then

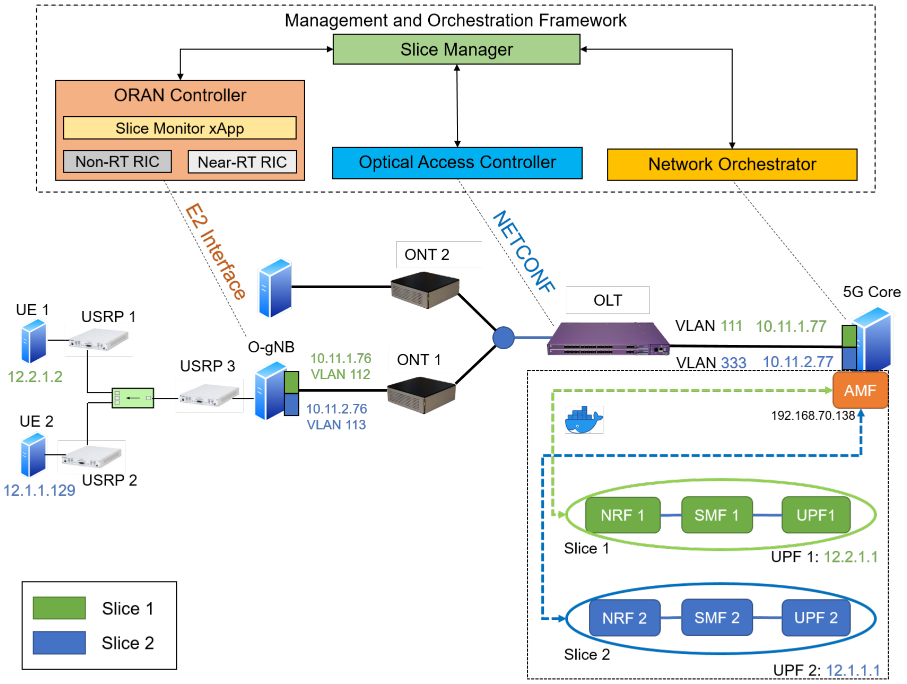


Figure C4.1: Network Architecture.

interacts with the optical access controller to adapt the reserved bandwidth basing on the effective slice conditions with pre-defined thresholds.

## C4.3 Experimental Setup and Results

The considered experimental setup primarily consists of three different parts that are the O-RAN, SDOAN and mobile core network, as shown in Fig. C4.1.

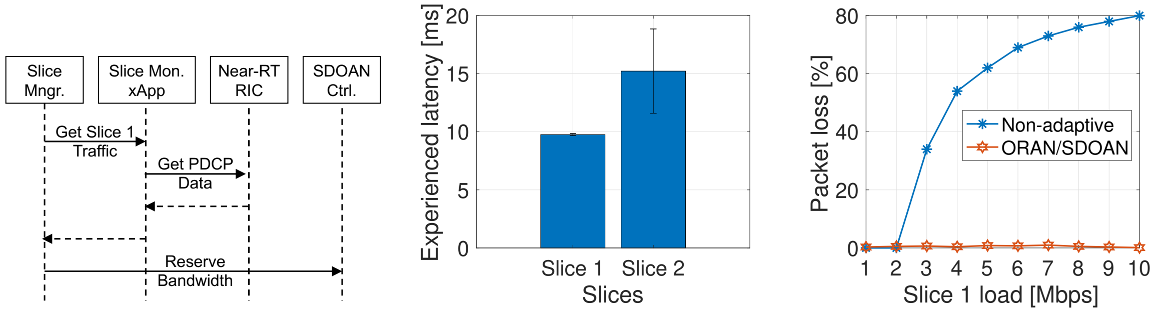
The O-RAN is deployed by using OpenAirInterface (OAI) open-source software stack [52]. We deployed two OAI-based UEs (*UE 1* and *UE 2*) and an O-gNB using National Instrument X310 Universal Software Radio Peripherals (USRPs). The OAI O-gNB is configured with Single – Network Slice Selection Assistance Information (S-NSSAI) list to support two different Slice/Service Type (SST) and Slice Differentiator (SD) values to evaluate the O-RAN slicing scenario. Without loss of generality, we assume that *UE 1* sends traffic over *Slice 1*, i.e. low-latency slice, and *UE 2* sends traffic over *Slice 2*. We adopt flexRIC [57] software as the near-RT RIC. At the mobile core network side, we deploy different sets of functions corresponding to the different slices.

The SDOAN is realized by utilizing a commercial Calix Axos E7-2 XGS-PON that supports NETCONF protocol. Traffic differentiation is based on VLAN tags. The O-gNB applies different VLAN tags to each packet going towards the core network by utilizing virtual ethernet interfaces. *Slice 1* and *Slice 2* are associated with VLANs 112 and 113, and VLANs 111 and 333 at ONT and OLT side, respectively. *Slice 1* is forwarded at the PON using the Expedited Forwarding mechanism while *Slice 2* applies Best Effort. Slice Manager and Slice Monitor xAPP are realized using the *C* language, whereas the Optical Access Controller is realized using *Python* language and is invoked from the Slice Manager to dynamically change the PON bandwidth based on the RAN traffic.

Fig. C4.2b shows a comparison in terms of average experienced latency measured between the UEs and core network. The end-to-end delay measured for *Slice 1* is significantly reduced when compared to *Slice 2*, thanks to the bandwidth reservation and traffic prioritization applied at the PON. It can be noticed that, as an effect of expedited forwarding in PON, delay variation is also reduced. It is also worth mentioning that such delays include the contribution of the O-RAN (which is impacted by OAI software implementation performance). Fig. C4.2c shows a comparison in terms of packet loss between the proposed slice management approach (exploiting O-RAN and SDOAN programmability) and a non-adaptive strategy, where bandwidth reservation at the PON is configured *a priori*. For the non-adaptive case, we assume 1 Mbps reserved bandwidth in the PON for *Slice 1*. We vary the traffic of *Slice 1* and measure packet loss by running *iperf* between *UE 1* and *UPF 1*. As described in Sec. C4.2, to offer low latency to *Slice 1*, bandwidth has to be reserved. When the traffic required by the slice exceeds the bandwidth reserved in the PON, packet loss is experienced since packets are dropped at the O-gNB and ONT without being forwarded. To this purpose, the proposed O-RAN/SDOAN based slice manager dynamically adapts the reserved bandwidth in the PON. As shown in Fig. C4.2c, this allows the network to keep packet loss limited while offering low-latency to a specific slice compared to a non-adaptive approach where the packet loss increases with the traffic load of the slice.

## C4.4 Conclusion

In this paper we propose an end-to-end slice management strategy which exploits O-RAN to retrieve information on radio access network load for a specific slice and dynamically adapts the bandwidth at the optical access network. Results show that the proposed mechanism allows the network to offer low end-to-end latency while reducing experienced packet loss and preserving optical access network efficiency.



(a) (b) (c)

Figure C4.2: (a) Slice management sequence diagram; (b) Slice experienced latency; (c) Packet loss comparison.

# Acknowledgment

This work has been partially supported by the Italian Government through projects FIRST, SICURA and INCIPICT, by the European Commission through the project H2020-MSCA-RISE-2019 OPTIMIST, and by DST SERB Startup Research Grant (SRG-2021-001522).

# Conference C5

# Latency-Aware Kubernetes Scheduling for Microservices Orchestration at the Edge

C. Centofanti, W. Tiberti, A. Marotta, F. Graziosi, D. Cassioli

Submitted to
*2023 IEEE International Conference on Network Softwarization (NET-SOFT)*

# Abstract

Network and computing infrastructures are nowadays challenged to meet the increasingly stringent requirements of novel applications. The most critical aspect appears to be the latency perceived by the end-user accessing the services. New network architectures offer a natural framework for the efficient orchestration of *microservices*. However, how to incorporate accurate latency metrics in the orchestration of microservices still represents an open challenge. In this work we propose a novel architecture to perform scheduling operations in a Kubernetes environment based on the latency effectively perceived at the application layer. Compared to other approaches, the proposed one collects latency data directly at the application layer from the orchestrated service instead of relying on external measurement services. We show the validity of our approach by adopting an iterative discovery strategy able to dynamically determine which node operates with the lowest latency for the Kubernetes pod placement.

## C5.1 Introduction

Next generation services pose novel challenges in the management and orchestration of network and computing infrastructures. Envisioned requirements in terms of latency, bandwidth, and reliability can only be met with a tight coupling between digital infrastructures and the services they host. Such integration is achievable by introducing more flexibility at both the network and computing layers. While at the network layer the required flexibility has been introduced by novel paradigms such as software defined networking and network function virtualization, the computing layer sees the emergence of *microservices* and *edge computing* paradigms as a means to meet requirements of the next generation applications.

To fully benefit from distributed computing infrastructures provided by edge computing, software architectures must be designed accordingly. A fundamental software architecture pillar for the so-called *edge-native* applications is represented by microservices. Microservices are a software architectural approach that consists in decomposing applications into small stateless independently developed and deployed functions called services able to communicate among each other.

Kubernetes [58] has emerged as a reference platform for the orchestration of microservices. It allows to automate, manage, and schedule the deployment of microservices in the form of one or more containers wrapped into entities called pods, which represent the smallest deployable units in Kubernetes. Pods are organized in *clusters* and can be moved over physical machines in different locations which are referred to as *nodes*.

A key component for the orchestration of microservices in Kubernetes is the scheduler which is responsible to elaborate and perform the placement of pods over the nodes. Being devised for traditional cloud environments relying on centralized high-capacity datacenters, standard Kubernetes scheduling strategies are aimed at optimizing computation resources such as CPU and memory [59]. However, considering edge scenarios, scheduling decisions may significantly impact the performance achievable by the applications orchestrated via Kubernetes.

Orchestration driven by application requirements still represents an open challenge in Kubernetes environment and some research efforts have been made in this direction. To achieve this goal architectural solutions enabling network-awareness in scheduling decisions need to be investigated. In [60] a scheduling strategy is proposed based on static characterization of network latency between cluster nodes. [61]

proposes a heuristic strategy for the deployment of placeholder pods for enhanced reliability and takes into account inter-node latency. [62,63] propose Kubernetes scheduler extensions to introduce network-awareness for efficient placement of pods. The above mentioned approaches focus on network awareness and rely on latency metrics collected at the network layer via independent applications responsible to monitor network performance parameters. However, latency experienced by end-users include a non-negligible contribution of the application-layer which for example may be impacted by computational resource saturation, hardware heterogeneity through cluster nodes and other external conditions. To perform accurate and realistic latency measurements is crucial for the effectiveness of the scheduling strategy, due to the variability of network and computation infrastructure conditions.

In this work, we propose an architectural solution to perform scheduling decision taking into account accurate latency data collected at the application-layer by the orchestrated service. We show the effectiveness of our architectural approach by implementing a simple discovery strategy able to dynamically allocate pods over the best physical nodes via ad-hoc deployment of replicas of the services. The proposed approach is able to reduce the latency experienced by users while at the same time maintains a low resource usage which represents a critical aspect in terms of costs [64].

## C5.2 System Model

The system model refers to a physical infrastructure consisting of a Kubernetes cluster where a set of $N = \{n_1, n_2, ...n_k\}$ nodes is managed by a Control Plane, as shown in the principle diagram in Figure J2.2.

The objective of the proposed method is to instantiate the service in the set of nodes closest to the end-user in terms of latency. This is reached by means of a *sentinel* replica of our service that enables the measurement of end-to-end application-layer latency for every node of the cluster. The enabling mechanism is a custom scheduler in the control plane that instantiates the service in multiple nodes, keeping track of which service is instantiated in which node. The symmetric functionality is the descheduler in the control plane which retrieves the latency information and deletes the less performing service instances. Both the user and the nodes run the *latency-meter* application which is instantiated as the client at the user side and as the server in the Kubernetes cluster. The server side of the *latency-meter* application is deployed through the custom scheduler in the Kubernetes cluster. The communication between

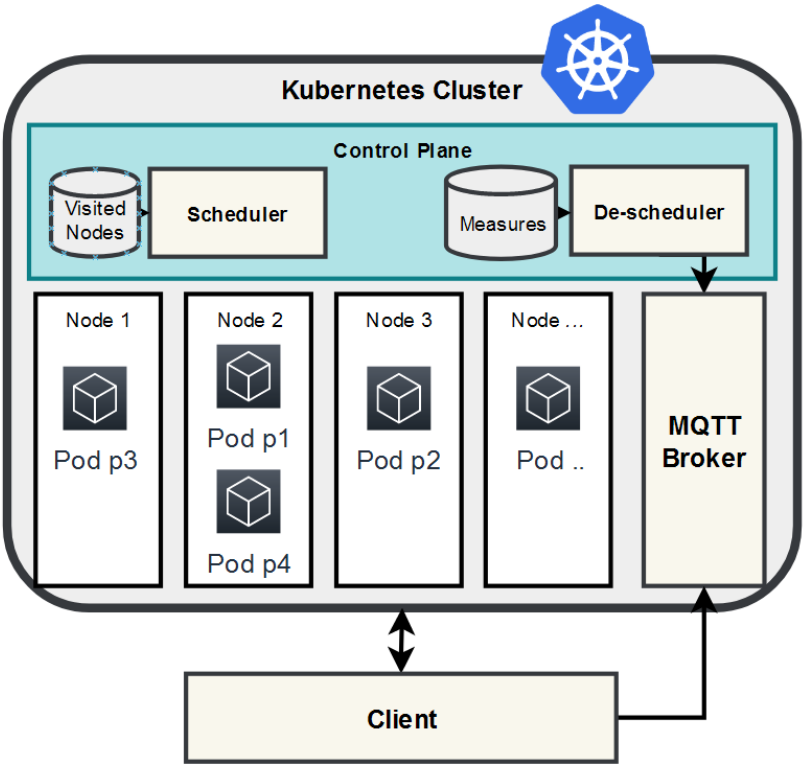


Figure C5.1: System Architecture.

the application and the de-scheduler is provided by the MQTT protocol via an MQTT broker.

The details and the functional description of the aforementioned components is reported in the following subsections.

### C5.2.1 Scheduler

Normally, the selection of which node will host a pod is often driven by the default Kubernetes scheduler. Its primary target is to optimize resource allocation in the cluster trying to fit the highest possible number of services inside the infrastructure. This selection, when not enforced, is opaque to both developers and clients of the target server application: from their point of view, the selected node is *random* and unpredictable. The expected latency is something in between the best and the worst case among all the cluster nodes. The use of a prediction mechanism is obviously possible but it would cause a significant overhead on the control plane as the number of applications and nodes increases. Furthermore, the prediction of the best choice in terms of network latency for the users requires a huge amounts of measurements data. In our model, instead, we adopt an *iterative* discovery strategy able to meet the optimal pod allocation in terms of the latency experienced by the clients of a target application. Our strategy generates a low overhead and does not rely on a predictor. It is worth noting that this approach also solves the cold-start problem when the available historical data are not sufficient for the system to take decisions.

This process is realized by providing custom latency aware scheduling and a de-scheduling services. Those two services cooperate together to iteratively move the service to the node which is nearest to the end-user. At each step, only the replica of the service with the best latency survives to the selection made from the de-scheduler. Then, the scheduler instantiates a new replica on a node which has not been tested yet. To do this, the scheduler keeps track of:

- $N$**:** the set of all the nodes in the cluster;
- $V$**:** the set of visited nodes in the cluster;
- $U$**:** the set of nodes to be probed which is $U = N - V$.

While running the latency aware scheduling mechanism it is possible to distinguish two main phases. The first one is a transient phase in which a sentinel replica of the Pod is deployed on each node to probe the application-layer latency. The second one starts when all nodes have been measured by the sentinel replica and the final decision happened. The latency goes down to the minimum allowed by current environmental conditions and remains stable for the end user in stationary conditions. The process is completely transparent to the end-use with no downtime during the whole process. The end-user will experiment only a variation in latency that converges to the minimum available in the cluster, i.e. an improvement in performances.

After a configured timeout, the $V$ set is emptied, so that the iteration can start again to detect and select the current best node allocation for the application. This enables the system to adapt to either a user that changes its position in relation to the cluster's nodes or a temporary change in network or computation available conditions that degrades the latency to a specific node.

The scheduler logic works as follows. Upon the request for scheduling a pod or a set of pods composing an application, the scheduler deploys an additional sentinel replica to test latency from another node. The procedure follows the steps shown in Algorithm 1.

The scheduling algorithm then waits for the de-schedule to kill one pod. The killed pod will be re-introduced to the scheduling queue from the Kubernetes controller and so the schedule can move forward scanning remaining nodes. When all the nodes have

**Algorithm 1** Scheduling logic

```
Input: Application $App$
Input: Set of nodes in the cluster $N$
Input: Set of visited nodes $V$
Input: Set of untested nodes $U$
Output: The selected node $n_i \in N$

$U \leftarrow N - V$
while $U \neq \{\}$ do
    $N_i \leftarrow random(U)$
    $V \leftarrow V \cup \{n_i\}$
    $scheduleProbe(n_i)$
    $U \leftarrow N - V$
end while
```

been visited, pod placement will be optimal in terms of latency. The sentinel replica used as probe by the scheduler is no longer needed and will not be scheduled again. Instead, if $U$ is not empty, the node $n_i$ is selected according to external requirements (e.g., load balancing, available resources etc.) or randomly.

### C5.2.2 Latency Measurement

The latency measurement application is composed by a client and a server application and has the aim to measure latency to the responding pod. This behaviour can be easily reproduced inside a real world application with little or no effort. In particular, the client has to measure, average and provide the current round-trip time ($RTT$) to reach the pod hosting the application server and the one hosting the sentinel replica, both accessible through the same IP address through a Kubernetes NodePort Service. This information is then sent to the de-scheduler (detailed in subsection C5.2.3) to decide which is the less-performing replica of the application in the cluster.

MQTT [65] protocol and a MQTT Broker have been adopted for this purpose to decouple the communication: a configured MQTT topic (e.g., latency-measurement) is used as the channel to exchange latency information. On one side, once the application client has measured and averaged the current $RTT$, it *publishes* such information, along with the ID of the measured node and a time reference, on the MQTT topic. On the other side, every component which requires the latency information (i.e., the de-scheduler) can *subscribe* to the MQTT topic to obtain and store such information.

The $RTT$ measurements are stored in a global data structure called *LatencyMeasurements* ($LM$) which is shown in Figure C5.2. This structure keeps the $RTT$ infor-

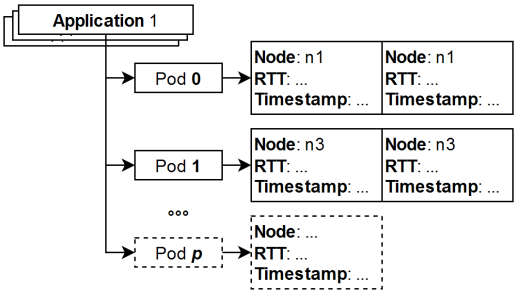


Figure C5.2: De-scheduler: data structure used to store latency information.

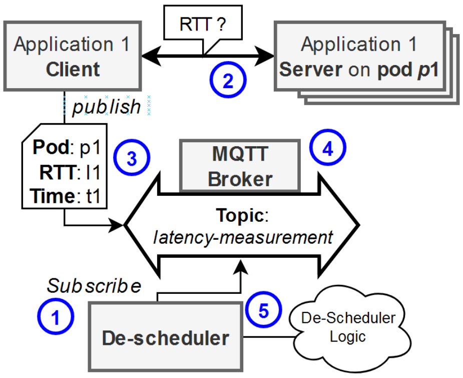


Figure C5.3: De-scheduling flow

mation on a per-application basis: the entry for an application contains the list of the pods of such application along with the list of received measurements data. Every measurement contains the node from where the pod run, the actual measurement and a timestamp counter used to relate the measurement with the time of measurement.

### C5.2.3 De-Scheduler

The de-scheduler component is responsible for retrieving the measurement from the MQTT broker and deleting the worst performing pods. Its operation are described in Figure C5.3:

1. Once activated, the de-scheduler contacts the MQTT Broker to *subscribe* to the MQTT topic that receives the RTT measurements

2. When an application client communicates with the server, it measures the RTT to communicate to a specific pod

3. After a timeout, the measured RTTs are averaged and serialized

4. The average RTT along the pod ID and timestamp are *published* to the MQTT topic via the MQTT Broker

5. Since the de-scheduler is subscribed to the MQTT topic, it receives the RTT data. It de-serializes the data and creates a new entry in the *LM* data structure, invoking the de-scheduling logic to decide if and which pod has to be de-scheduled.

The de-scheduling logic is triggered when there are enough measurements stored in the *LM* structure for a give application. In particular, we define the *De-scheduling Threshold* $T$ as the minimum number of measured pods required to trigger the de-scheduling logic. For example, $T = 2$ means that the de-scheduler is triggered when it receives the RTT measurement from at least 2 different pods. In general, given $R_n$ as the number of desired replicas for the service, then we have $T = R_n + 1$.

**Algorithm 2** De-scheduling logic (simplified)

```
Input: Application $App$
Input: De-Scheduling Threshold $T$
Input: Latency Measurements $LM$

while True do
    $M \leftarrow$ number of measured pods for $App$
    if $M \geq T$ then
        /* Retrieve the pod with maximum RTT */
        $P_k \leftarrow max_{RTT}(LM, App)$
        /* Deschedule $p_k$ and clear its entry in $LM$ */
        $Deschedule(p_k)$
        $LM[App][p_k] \leftarrow None$
    end if
end while
```

The de-scheduling logic works as follows (Algorithm 2). The de-scheduler monitors the number of measured pods $M$ in the *LM* data structure of the target application *App*. Once such number is greater or equal to the configured De-scheduling threshold $T$, the pod $p_k$ with the worst RTT is retrieved, de-scheduled and its measurements are deleted from $LM$. If two pods have the same RTT, the one with the older measurement timestamps is removed. In this way, the de-scheduler keeps alive the pods with newest measurements and deletes those having old ones.

The de-scheduling of a pod will cause a mismatch between current state and desired one inside the Kubernetes cluster. The Kubernetes controller tries to bring back the system to the desired state adding a new pod to the scheduling queue. This triggers the scheduler which starts a new scheduling procedure for the pod. As mentioned in Section C5.2.1, the latency aware scheduler stores the set of visited nodes, i.e. the nodes that have or had a pod running for target application. This set of nodes is excluded from the set of new schedulable nodes taken into account by the scheduling procedure. Eventually, the set of visited nodes will be equal to the available nodes, causing the scheduler and the de-scheduler mechanism to stop. This is an intended behaviour and leaves the application running on the $T-1$ best performing nodes, according to the desired state declared by the application server itself.

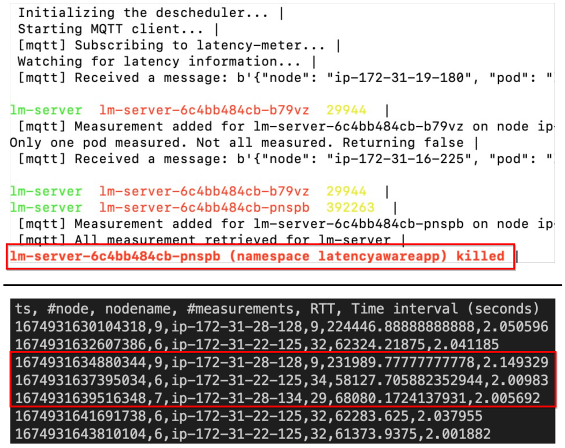
## C5.3 Implementation & Results

The presented approach has been implemented in a cluster of 10 virtual nodes. Nodes are Virtual Machines running on a micro-datacenter on VMware ESXi hypervisor controlled by a VMware vCenter platform. Each VM runs a MicroK8s Kubernetes distribution on the top of the latest Ubuntu 22.04.1 LTS operating system. We configured the cluster with high-availability capabilities, enabling the master node capabilities on each VM. This provides by default resiliency over node failure in addiction to the application-layer high availability provided by the K8s core. We deployed CoreDNS on Kubernetes to provide Domain Name resolution inside the cluster and ease the communications through our deployed applications.

Latency in this scenario depends by network and application latency. In a real deployment of the infrastructure, it is reasonable to assert the application-layer latency is sightly lower than the network one for nearly-real-time and low latency applications. This means that the network distance from the client to the server gives the most of the latency contribution. In our specific implementation, all VMs were co-located and had the same network latency to the client. To overcome that, we randomize the latency of each node around an arbitrary mean value per node with a given variance [66]. The mean values for each of the 10 nodes are respectively 1, 5, 10, 15, 20, 30, 40, 100, 200, 350 ms.

To implement the behaviour described in previous sections, we designed and developed a set of micro-services [67] which implement all the described logic.

After running the application into the Kubernetes cluster, the Latency Measurement Client (LMC) application needs to be configured to point to the cluster. When the client starts, it begin to collect latency metrics of the application latency. The server replies to the client requests providing information about the pod which is handling the task. Once messages are received by the client, the LMC application



```
 Initializing the descheduler... |
 Starting MQTT client... |
 [mqtt] Subscribing to latency-meter... |
 Watching for latency information... |
 [mqtt] Received a message: b'{"node": "ip-172-31-19-180", "pod": "

lm-server  lm-server-6c4bb484cb-b79vz  29944  |
 [mqtt] Measurement added for lm-server-6c4bb484cb-b79vz on node ip
Only one pod measured. Not all measured. Returning false |
 [mqtt] Received a message: b'{"node": "ip-172-31-16-225", "pod": "

lm-server  lm-server-6c4bb484cb-b79vz  29944  |
lm-server  lm-server-6c4bb484cb-pnspb  392263  |
 [mqtt] Measurement added for lm-server-6c4bb484cb-pnspb on node ip
 [mqtt] All measurement retrieved for lm-server |
lm-server-6c4bb484cb-pnspb (namespace latencyawareapp) killed |
```

```
ts, #node, nodename, #measurements, RTT, Time interval (seconds)
1674931630104318,9,ip-172-31-28-128,9,224446.88888888888,2.050596
1674931632607386,6,ip-172-31-22-125,32,62324.21875,2.041185
1674931634880344,9,ip-172-31-28-128,9,231989.77777777778,2.149329
1674931637395034,6,ip-172-31-22-125,34,58127.705882352944,2.00983
1674931639516348,7,ip-172-31-28-134,29,68080.1724137931,2.005692
1674931641691738,6,ip-172-31-22-125,32,62283.625,2.037955
1674931643810104,6,ip-172-31-22-125,32,61373.9375,2.001882
```

Figure C5.4: Excerpt of the output measurements taken from our Latency Aware de-scheduler (top side) and from our Latency Measurement Client app (bottom side)

is able to measure the overall RTT which is composed by two contributes: the network's latency and the application's latency. This value is sent to the MQTT Broker topic along with the ID of the measured pod. The de-scheduler gets the metrics from the MQTT Broker topic. When it has got enough metrics to do a comparison, it decides which is the pod that is performing worse and kills it applying the algorithm in Sec. 2. At this point, the Kubernetes controller discovers that the current status does not match with the desired state and asks for the creation of a new pod. Our schedule takes in charge the request and processes the creation of the new pod and the scheduling on one of the remaining nodes, following the algorithm 1. The LMC application, meanwhile, continues to measure the latency and is not aware on what is going on the cluster. The client perceives only a latency fluctuation due to the service moving to the end user.

Figure C5.4 shows the output of the de-scheduler application (top-side) and the one of the LMC (bottom-side). The red square on the de-scheduler output highlights the moment in which the de-scheduler kills the latency meter server (e.g. lm-server). The red square on the client's output shows the change in latency measured by the client.

The default scheduler is only able to schedule the required pod once. This means that the placement is done only at scheduling time and the server-side application deployed is bounded to a specified node's latency. In our case, we assume that the chosen node is the one with 200$ms$ of latency. This is the starting point of both *latency aware* and default scheduler curves in Figure C5.5. We use the default Kubernetes scheduler as a baseline, in the case where both the default scheduler and our latency

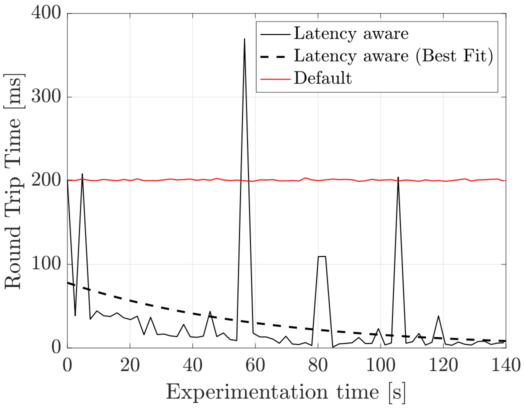


Figure C5.5: Latency measured over time from the client perspective

aware scheduler chose the same node at the beginning. The default scheduler does not take any further decision so the latency only fluctuate following Gaussian distribution around the average measured latency until the network is not congested. On the other hand, our latency aware scheduling mechanism is able to decrease end-to-end latency step by step while probing all the cluster nodes by means of the sentinel replica.

Figure C5.5 shows that our proposed latency-aware scheduler significantly reduces the latency while probing new nodes. For each new node evaluated, the perceived latency is bi-stable between the current minimum latency discovered and the latency the node to which the sentinel replica is deployed at the moment. The best fit curve is an interpolation that describes the attended mean latency value at each time using our approach. At second 120 in Figure C5.5 all nodes in the cluster have been probed by the sentinel. The de-scheduler takes its final decision saving the lowest latency's pod. From second 120 onward, the measured value is around 1ms, which is the lowest end-to-end application-layer latency allowed in our setup. This means that our mechanism converged to the best solution and no other action is required.

## C5.4 Conclusion and future works

In this work we proposed scheduling and de-scheduling algorithms to close the current gap of Kubernetes scheduler in the edge applications placement. We showed how to implement a Latency-Aware Kubernetes scheduler that is able to orchestrate microservices at the edge to find the optimal placement while seamlessly reducing the user's perceived latency.

Results show that our solution outperforms the default Kubernetes scheduler over time. Starting from the same node at 200ms, the default scheduler does not take any action on pod's placement. Instead, our latency-aware scheduling mechanism takes care to explore other nodes and places the pod to the most suitable node. The spikes on measurements are due to the placement of the probe replica and disappear as soon as the de-scheduler is able to take decision. The overall trend is a decreasing curve that reaches 1ms after measuring all nodes.

Future direction include the extension of this work to a scenario which include user mobility and network congestion.

## Acknowledgment

This work has been partially supported by the European Commission through the project H2020-MSCA-RISE-2019 OPTIMIST and by the Italian Government through projects SICURA and INCIPICT.

# Journal J1

# Security over the O-RAN Inter-Controller Interface: a Blockchain-based Anti-tampering Scheme for Traffic Policies

W. Tiberti, E. Di Fina, C. Centofanti, A. Marotta, D. Cassioli

Submitted to
*IEEE Journal on Selected Areas in Communications (JSAC)*

# Abstract

To dynamically adapt the network to the modern traffic scenarios, mobile operators and standardization bodies resorted to the definition of flexible mobile network architectures. Among numerous proposals, the Open-Radio Access Network (O-RAN) Alliance aims to create an open, interoperable, high-performance and low-cost architecture for the new generation of RANs. O-RAN features two RAN Intelligent Controllers (RICs) to manage the traffic and the established traffic policies. Previous works revealed that a malicious actor who has access to the inter-controller communication interface can spoof and manipulate traffic policies at will. In light of this, in this paper, we review and analyze the current definition/specification of the O-RAN inter-controller security and show the threats and vulnerabilities. To improve the O-RAN security, we propose a blockchain-based approach to guarantee that the mechanisms to establish the policies are not exploited by malicious users, thus maintaining policies *immutable*. This approach prevents any installation on the O-RAN Near-Realtime RIC of malicious or unwanted policies and provides useful information for verification and security auditing. Our approach has been verified successfully in two scenarios, including the official O-RAN software implementation. To the best of our knowledge, this is the first approach that provides immutability to an O-RAN communication interface.

## J1.1 Introduction

With the increasing complexity of mobile networks and services, inter-operability has become a crucial aspect. Introduction of novel technologies requiring network infrastructure upgrades or installation of additional devices has been till recent times constrained to the adoption of hardware and software offered by vendors already in-field. Such closure has been traditionally imposed by manufacturers to economically benefit from the so-called *vendor lock-in* condition. This implied for several years the design of Radio Access Networks (RANs) equipments as all-in-one units equipped with proprietary interfaces that allow communication only with other network devices of the same vendor and that implement the entire cell protocol stack.

With the advent of Software Defined Networking (SDN) and Network Function Virtualization (NFV), together with the need of unprecedented densified and flexible mobile networks, inter-operability and openness of RANs has become essential from the network operators viewpoint.

To achieve this goal, in 2018 the O-RAN Alliance was established, with the aim to create a new network architecture based on a set of standardized and open interfaces, i.e., an Open RAN (O-RAN). O-RAN is based on disaggregated and possibly virtualized RAN components connected via open interfaces and governed by radio intelligent controllers implementing data-driven closed-loop control [68]. The multi-vendor and programmable nature of O-RAN will significantly impact the way RAN will be designed, deployed, and operated.

While it is undeniable that such evolution towards disaggregation, openness, and programmability via data-driven approaches will make next-generation cellular networks more efficient and flexible, this poses novel challenges, especially from a cyber security perspective. In principle, the distributed and disaggregated O-RAN architecture effectively extends the attack surface for malicious users. This raises severe threats to network operators and the services supported by mobile networks. It is worth mentioning that, since security represents a major concern in O-RAN, a dedicated working group has been established by the O-RAN alliance with the aim of analyzing and define threat models and security measures towards zero-trust model [68].

Nevertheless, security in O-RAN still deserves investigation, as shown in the analysis of the O-RAN threat model and security solution strategy presented in [69], where vulnerabilities arising by missing authentication and authorization policies are shown by performing practical implementation of security strategies and test cases. In particular, the O-RAN's Service Management and Orchestration (SMO), the Near-Real

Time RAN Intelligent Controller (Near-RT RIC), and the $O1$ interface, are individuated as critical elements in this perspective.

In this paper we carry out a security analysis on the current O-RAN specification of show the threats and vulnerabilities we discovered on the inter-controller interface. In order to pose a mitigation those attacks, we propose a novel blockchain-based approach which maintains policies *immutable*, guaranteeing resilience against attacks that target traffic. This immutability property is important in scenarios where auditability is desired, such as in maintaining access logs for policy definition.

### J1.1.1 Contributions

The explicit contribution of this work can be summarized as follows:

- We review and analyze the O-RAN security in the state-of-the-art and in the current architecture specifications;
- We evaluate the impact of Man-in-the-Middle attacks on the inter-controller communication interface when an attacker has bypassed existing authentication measures and obtained access to such;
- We propose a blockchain-based approach to mitigate the active tampering of traffic policies by an attacker who use spoofing to forge malicious policies;
- We verify and evaluate the attacks and the performance of the proposed approach in three experiments featuring two different scenarios, including the current version [1] of the O-RAN implementation [70]. We show that the proposed approach fulfills the latency constraints of the O-RAN standard.

### J1.1.2 Paper Organization

The rest of the paper is organized as follows. Section J1.2 introduces the O-RAN Architecture, the controllers and the inter-controller interface; Section J1.3 reports on the state-of-the-art regarding O-RAN and, in particular, the O-RAN security issues; Section J1.4 analyzes the current O-RAN specification regarding security; Section J1.5 presents a Man-in-the-Middle attack scenario along with a blockchain-based approach useful to mitigate incoming attacks; Section J1.6 reports the experimental validation and performance evaluation of both attacks and mitigations. Finally, the paper concludes with the final remarks and future works contained in Section J2.8.

[1] At the time of writing, the current release of the official O-RAN implementation is ”Release F”

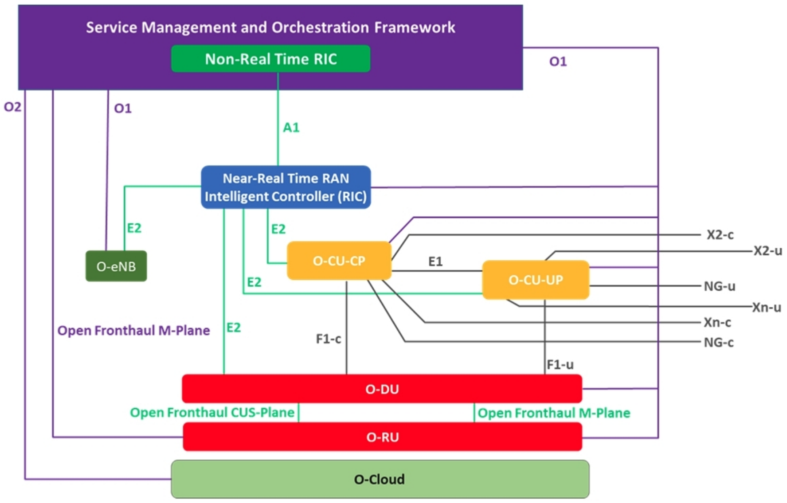


Figure J1.1: O-RAN Architecture [71].

## J1.2 The O-RAN Architecture

In Fig. J1.1 the logical architecture of O-RAN is shown. This architecture has two different controllers, Central Units, Distributed Units and Radio Units functions. Four interfaces ($A1$, $O1$, Open Fronthaul M-plane, $O2$) connect the Service Management and Orchestration (SMO) framework to the O-RAN network functions and O-Cloud [71]. The O-RAN architecture supports different controllers: the Non-Real Time RAN Intelligent Controller (RIC) and the Near-Real Time RIC. The timing of these controller is use case dependent: typical execution time for use cases involving the Non-RT controller is 1 second or more; for the Near-RT controller it is in the order of 10 milliseconds or more.

In the O-RAN architecture, the radio side includes Near-RT RIC, O-CU, O-DU, and O-RU Managed Functions, and the management side is comprised of the Service Management and Orchestration Framework (including the Non-RT RIC) [71].

The SMO is responsible for the RAN domain management, in particular for the *fault, configuration, accounting, performance and security (FCAPS) interface* to O-RAN Network Functions, Non-RT RIC for RAN optimization, O-Cloud Management, Orchestration and Workflow Management [71]. The SMO performs these services through four key interfaces to the O-RAN elements: $A1$ Interface between the Non-RT RIC in the SMO and the Near-RT RIC for RAN Optimization; $O1$ Interface between the SMO and the O-RAN Network Functions for FCAPS support; in the hybrid model, Open Fronthaul M-plane interface between SMO and O-RU for FCAPS support; $O2$ Interface between the SMO and the O-Cloud to provide platform resources and workload management [71]. The SMO provides the capability of managing the O-Clouds and supports for the orchestration of platform and application elements and workflow management through O2 interface. The O2 interface supports the management of the cloud infrastructure and the use of the cloud resources allocated to the RAN.

### J1.2.1 The O-RAN Intelligent Controllers

Located inside the SMO Framework, responsible for the RAN domain management, the **Non-RT RIC** consists of a subset of SMO's functionality, directly connected via $A1$ interface. Its primary goal is to support intelligent RAN optimization by providing policy-based guidance, Machine Learning (ML) Model Management and Enrichment Information (EI) to the Near-RT RIC function so that the RAN can optimize its operation. It consists of two sub-functions:

- **Non-RT RIC Framework**, a functionality internal to the SMO Framework that logically terminates the $A1$ interface and exposes the required services to Non-RT RIC Applications (rApps) through its $R1$ interface.

- **rApps**, i.e., modular applications that leverage the functionality exposed by the Non-RT RIC Framework to perform RAN optimization and other functions (for example, driving content - Policy or Enrichment Information - across the $A1$ interface) [72].

The Non-RT RIC Framework is responsible for exposing all required functionality to the rApps, either from the Non-RT RIC Framework, or the SMO Framework, via the $R1$ interface.

The **Near-RT RIC** hosts one or more applications, referred to as **xApps**, that use the $E2$ interface to collect near real-time information on, e.g., a user equipment (UE) basis or a Cell basis, and provide added-value services. The Near-RT RIC also hosts a Messaging Infrastructure that provides low-latency message delivery service between Near-RT RIC internal endpoints. The Near-RT RIC may receive declarative Policies and obtain Data Enrichment information over the $A1$ interface [73]. The control by the Near-RT RIC over the $E2$ nodes is steered through the policies and the enrichment data provided via $A1$ interface from the Non-RT RIC [71]. The Near-RT RIC can be connected and control multiple $E2$ nodes or a single $E2$ node.

### J1.2.2 A1 interface

The O-RAN architecture includes many interfaces that can cooperate to achieve higher level objectives. In this work we focus on $A1$ interface because it is critical to the system as it carries high order policy statements related to Policy Management, Enrichment Information and Machine Learning management services. $A1$ is an open logical interface between the Non-RT RIC functionality in SMO and the Near-RT RIC functionality, as shown in Fig. J1.1 e J1.2. The $A1$ interface termination enables the Non-RT RIC framework to send and receive messages to and from the Near-RT RIC. Both the $A1$ policies and the $A1$ Enrichment Information (EI), derived from either internal or external sources, are provided by the Non-RT to the Near-RT RIC via the $A1$ interface.

The **Policy Management Service** is responsible of transferring *policy management information* from Non-RT to Near-RT RIC and, viceversa, the *policy feedback* from Near-RT to Non-RT RIC. The $A1$ policies are declarative policies that contain statements on policy objectives and policy resources applicable to UEs and cells.

The **EI Service** deals with the discovery and request of $A1$ EI from Near-RT RIC to Non-RT RIC and delivery of $A1$ EI from Non-RT RIC to Near-RT RIC.

The **ML Model Management Service** at the $A1$ Interface is trained in the SMO layer and then used by the Non-RT RIC to improve the monitoring and guidance of the RAN based on the $O1$ observability or used by the Near-RT RIC to improve the optimization of the RAN [74, 75].

The purpose of $A1$ policies is to enable the Non-RT RIC function in the SMO to guide the Near-RT RIC operation, and hence the RAN, towards a better fulfilment of the *RAN intent* according to the new "intent-based network management framework" introduced specifically in the O-RAN [76]. By utilizing the observability over $O1$ and the $A1$ policy feedback, it may happen that the Non-RT RIC concludes that the RAN Intent is not achieved. The Non-RT RIC can then decide to use $A1$ policies that enable the Near-RT RIC to e.g. optimize the radio resource management (RRM) for a single UEs or a group of UEs [75].

There are different types of $A1$ policies referred to as *policy types*, identified by the `PolicyTypeID`. An $A1$ policy is identified by a policy identifier `PolicyID` assigned by Non-RT RIC. $A1$ policies consist of a *scope identifier* and one or more *policy statements*. The scope identifier represents what the policy statements are to be applied on (e.g. UEs, QoS flows, or cells). The policy statements represent the goals to the Near-RT RIC and covers policy objectives and policy resources [75].

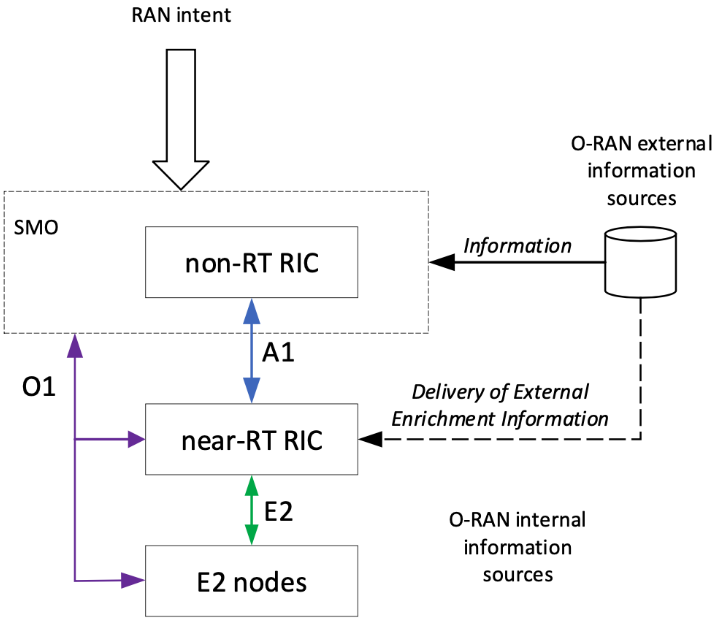

RAN intent
O-RAN external information sources
SMO
non-RT RIC
Information
A1
O1
Delivery of External Enrichment Information
near-RT RIC
E2
O-RAN internal information sources
E2 nodes

Figure J1.2: $A1$ interface as illustrated in [75].

$A1$ policies are not critical to traffic but they are non-persistent, i.e., do not survive a restart of the Near-RT RIC. The $A1$ interface uses the TCP at transport layer to provide the communication service, TLS to provide secure HTTP connections, HTTP as the application-level protocol and JSON format at the data interchange layer.

## J1.3 Related Works

Many authors are studying this innovative standard and all works are recent. In [68] a detailed tutorial on O-RAN and challenges of O-RAN networks are presented and addressed to researchers in the sector. Artificial Intelligence and Machine Learning are the central themes of the studies carried out until now, as we find in [77], [78] and [79]. How O-RAN can improve 5G communications is a central topic in [80] and in [81].

Being a recent topic and given the number of different aspects, features and technology involved, providing a comprehensive and deep analysis of O-RAN security is not possible. Two are the approaches adopted: some works, as [82] and [83]m provide an high level well-rounded overview and taxonomy of the security issues, but most of the research works focus on the security of specific parts of the O-RAN. In the next paragraphs, we give an overview on how the research in security is moving on the different parts of the O-RAN architecture.

The question whether an open architecture is more or less secure has no trivial answer: in [84] authors try to answer for what concerns the virtualization, containerization and AI/ML usage in both 6G and O-RAN. Among the results, they highlight a possible increase of the attack surface of the whole RAN. In fact, the AI/ML feature introduces a whole class of attacks in the O-RAN, as described in [85], such as the *Adversarial Attacks* which is well-known in the context of high-end security AI/ML applications but relatively new in the O-RAN context.

Despite the great effort of the O-RAN Alliance in adopting state-of-the-art security solutions and enforce best practices, the research community suggests that O-RAN may suffer from the lack of an enforced and well-defined authentication mechanism as stated in [69]. O-RAN may support some of the 5G authentication mechanisms used to provide a stronger authentication. For example, in [86] the authors present an alternative authentication and key agreement protocol to resist to active attacks caused by malicious serving networks. In a previous work [87], we analyzed the possible consequences of Man-in-the-Middle attacks on the inter-controller interface. The results show that, once the attacker has access to the interface, such attacks are trivial and may allow the attacker to tamper traffic policies, abuse the administration services, perform denial-of-service attacks and potentially escalade the attack to other O-RAN components and interfaces. Other works focus instead on the detection of intrusions: in [88] authors propose an algorithm for 4G and 5G to provide an effective anomaly detection feature which has been also tested on the O-RAN architecture.

Other parts of O-RAN are the subject of security analyses: for example, in [89] authors focus on the O-RAN Front Haul and propose an approach for securing it by providing confidentiality, authentication and integrity.

Thanks to their ability immutability to data sequences, technologies based on blockchains are ubiquitous and RANs make no exception. It is worth noting, however, that the objective for which blockchains are used may be very different across different applications: an example is given in [90], where the authors adopt a blockchain approach to provide a way to securely share the RAN resources across operators. In contrast to this, in this paper we instead use blockchains exclusively for their abilities of granting immutability to data sequences, a feature we use to prevent the tampering of traffic policies as described in Section J1.5.

## J1.4 O-RAN Security Overview

The goal of the O-RAN Alliance is to achieve a secure, open, and interoperable RAN. Building upon security advancements from the 3GPP and IETF standards development organizations, the O-RAN Alliance is specifying an O-RAN security architecture that enables 5G CSP (Communication Service Providers) to deploy and operate O-RAN with the same level of confidence as a 3GPP-specified RAN.

In order to highlight the importance of security, the O-RAN Alliance has created a separated working group focusing on security, the so-called Security Focus Group (SFG). The SFG is responsible for adopting the best state-of-the-art security methodologies and solutions to ensure resilience against attacks. The SFG activities are documented as part of the O-RAN specification, which contains information on security requirements, protocols, specifications and recommendations.

### J1.4.1 General Security Approach

As a general guideline, the O-RAN SFG has chosen to adopt well-known security approaches in the definition of the O-RAN architecture. O-RAN SFG adopts the *Security-by-Design* approach to guide the definition of every security-related aspect from the design phase. The Zero-Trust approach [91] and the STRIDE model [92] are adopted for conducting threat analyses and managing the security during the lifecycle of the O-RAN components.

### J1.4.2 O-RAN Threat modeling and Vulnerability analysis

The starting point to understand the O-RAN security perspective is the definition of the *Threat Model* adopted and the *Risk Analysis*, both reported in [93]. The document follows the ISO 27005 standard [94] in defining the stakeholders, assets, threats, vulnerabilities and possible countermeasures.

O-RAN's attack surface can be divided into six main groups:

- SMO, Non-RT RIC (including rApps), and Near-RT RIC (including xApps)
- Open interfaces: $A1$, $E2$, $O1$, $O2$, and Open FH
- Modified architecture: Lower Layer Split with Open FH
- Trust Chain: decoupling of hardware and software along with use of third-party xApps and rApps.

- Containerization and Virtualization: Container and cloud security risks.
- Open-source software: Exposure to zero-day vulnerabilities and public exploits.

Regarding the $A1$ interface and the $A1$ policies, both are defined as *critical* assets (i.e., `ASSET D-07`) requiring confidentiality, authenticity and integrity. The main vulnerabilities that affect the $A1$ interface, as identified by the SFG, are:

- The presence of a malicious Non-RT RIC or malicious Near-RT RIC (`T-A1-01`)
- Sniffing on the $A1$ interface by a malicious actor (`T-A1-02`)
- Tampering and Message Injection on the $A1$ interface (`T-A1-03`)
- The creation of malicious $A1$ policies (`T-NONRIC-03`)

In the scope of this paper, the last two vulnerabilities are crucial: a malicious actor who has access on the $A1$ interface could spoof and inject malicious policies to the Near-RT RIC without being noticed and possibly causing denial-of-services or more subtle attacks (e.g., selectively prioritize some traffic).

## J1.4.3 Protocol Security

In order to secure the communications among the O-RAN components, the SFG defined a set of security measures to be adopted in the communication protocols for a compliant O-RAN implementation [95]. The protocols under analysis and their security specifications are detailed below.

### J1.4.3.1 Cryptography suites

A common element in the following protocol specifications is the set of cipher suites supported by the O-RAN. In this context, a cipher suite is the selection of 1) a symmetric-key encryption algorithm and a mode of operation; 2) a cryptographic hash function; 3) a digital signature algorithm; 4) a key-exchange algorithm and 5) a message authentication code. Notable cipher suites that are *not* suggested include RSA for encryption and SHA1 for hashing. Despite to this careful selection of cipher suites, the use of some of the accepted algorithms is not universally approved, as we report in Section J1.4.5.

#### J1.4.3.2 SSH

The Secure Shell (SSH) protocol is used to grant confidentiality, authentication and integrity in both O-RAN and 3GPP interfaces. The SFG, however, puts restrictions on the SSH configuration to avoid security holes. For example, O-RAN supports only SSH version 2 or greater and a restricted set of cipher suites. Also, the SFG suggests to "*stay current*" and provide an upgrade path in case of new (and probably more secure) SSH versions are released.

#### J1.4.3.3 TLS and DTLS

The use of the Transport Layer Security (TLS) is critical for securing the O-RAN communications. The SFG specifies that currently only TLS version 1.2 and 1.3 should be supported and only version 1.3 in the future. An additional element is the definition of three different classes of cipher suites (*modern*, *intermediate* and *old*) to address the different requirements and features specific to the implementation. As for TLS, also a recent version of DTLS is required (greater or equal to version 1.2). Version 1.0 is explicitly not supported.

#### J1.4.3.4 IPSec

For Layer 3 security, the SFG suggests the use of IPSec in *Tunnel mode*, ESP Authentication and support for NAT Traversal, IKEv2 and X.509v3 certificates. The specification reports also additional details that are outside the scope of this paper.

### J1.4.4 Security Recommendations

The SFG has defined a set of general recommendations for provide a secure O-RAN implementation, lifecycle and testing procedures. Some of the general recommendations are:

- Definition of a Public Key Infrastructure (PKI)
- Access Management
- Logging, Monitoring and periodic Auditing
- Physical Protection
- Incident Response and Patch Management

For the $A1$ interface, the core recommendation (`REQ-SEC-A1-1` and `SEC-CTL-A1`) is to use TLS to support confidentiality, origin authentication, integrity, replay protection.

### J1.4.5 Existing security issues

Despite such effort, the current O-RAN specification on the security protocols [95] is vague: while the suggested protocol stack features safe hash functions, TLSv1.2 and v1.3 and IPSec, it also allows non-secure alternatives which may brake O-RAN security altogether, e.g., short keys, No TLS, SHA1 [96] as hashing function, support for null-confidentiality/no replay protection in IPSec etc. Moreover, the current specification forces to use specific cipher suites which are not universally approved for their security (e.g., NIST standard P256 elliptic curve [97]).

The Application Layer, also, opens the way to an addition attack surface, represented by the Web and HTTP security. The O-RAN Alliance and the SFG do not enforce strong security measures in the defined HTTP REST API endpoints but only general measures (e.g., `T-ProtocolStack` threats defined in [98]). This may expose the implementation to severe security flaws such as *A1 Policy overwriting*, *A1 Policy Deletion* or similar attacks as we demonstrate in Section J1.6.3.

The overall security scenario is also worsened by two facts. First, some of the most critical security specifications documents are not yet completed or contain misleading information at the time of writing. Examples are the empty sections in [99], the references in [100] to yet-to-exist protocol secure usage as for OAuthv2 or JWT in [95] or the lack of analysis of the attack surface brought by the Machine-Learning techniques adopted in the O-RAN architecture. Second, many critical security aspects are left to the implementation, which may decide to skip over some security measures while still maintaining perfect compliance with the O-RAN specification. Moreover, the official implementation of O-RAN [70] lacks a default secure environment (i.e., TLS and the PKI needs to be configured entirely by the implementor) which does not help operators in ensuring that all the theoretical security aspects are addressed correctly.

The resulting scenario, in our opinion, makes the implementation of a fully-secure O-RAN a non-trivial task.

## J1.5 Vulnerability Analysis and Mitigation Strategy

### J1.5.1 Threat Model

For sake of demonstrating the security issues in the $A1$ interface and the possible mitigations, we make the following assumptions:

1. There is a motivated and well-informed attacker (i.e., with deep knowledge on the O-RAN architecture, specifications and implementation) which has gained access on the sub-network containing the two RICs and the $A1$ interface. Such access may come from a rogue device, either physical or virtual (i.e., Virtual Machines, containers) and it is not detectable while the attacker is passive or while he/she is sending well-formed messages.

2. The attacker aims to know about the active $A1$ policies and to impose his/her own (malicious) policies, deleting the current policies or manipulating them to cause denial-of-services.

3. The attacker starts its activity *after* the O-RAN has been deployed. He/she is unable to the access $A1$ interface at or before deployment time and he/she is unable to sniff preliminary and configuration data from the RICs interfaces.

4. The cryptographic algorithms adopted are safe and trusted and the attacker has no enough time/computational power to break them via bruteforcing (e.g., for hash functions, he/she cannot find hash collisions deterministically)

### J1.5.2 Vulnerability Assessment of A1 Interface

The two controllers play a key role in the O-RAN architecture, thus it is very important to analyze the impact of malicious attacks over them. In particular, the $A1$ interface enables the communications between the Non-RT RIC and the Near-RT RIC, and, as identified in [93], represents a critical target of attacks. Actually, a malicious user could **intercept** the $A1$ policies and related schemes and alter them. This kind of attacks may produce a malfunctioning of the O-RAN and degrade its performance.

Let's analyze the regular operation setup for the $A1$ interface prior to launching the attack. It is shown in Fig. J1.3.

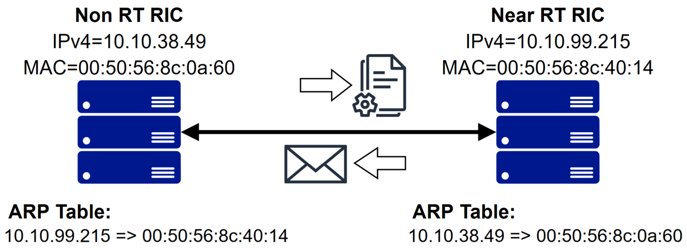


Figure J1.3: Setup of the O-RAN $A1$ interface in regular operation.

We implemented this scenario in the official implementation provided by the O-RAN Alliance and inspired to the real implementation of the proposed architecture and protocols. In the regular operation scenario, both the controllers have got the IP and MAC addresses and can communicate, having stored the correct entries in their ARP tables.

While assessing the official O-RAN implementation, we discovered two possible attack classes:

- Layer-2 Man-in-the-Middle attacks
- Policy Tampering Attacks.

In the following, we provide a detailed description of the two attacks in the O-RAN $A1$ interface, whereas the experiments are described in Sec. J1.6.

#### J1.5.2.1 ARP Cache Poisoning Attack

The two controllers can be tricked into sending messages to the attacker's machine. This kind of attack is known as *ARP Cache Poisoning*, which is a well-known Layer-2 Man-in-the-Middle attack. The attacker then proceeds to forward (or tamper) the information to its recipients.

The MitM attack is performed by using *Ettercap* [101] installed on an attacker-controlled virtual machine, which is virtually connected to both the two O-RAN controllers to simulate a post-intrusion scenario, where the attacker has a foothold in the same sub-network of the O-RAN controllers. Ettercap allows to perform an ARP Cache Poisoning attack by means of *ARP Gratuitous Reply*. This kind of messages are not solicited by any ARP Request and generally they are not malicious.

Their use by hosts with a double network interface is not malicious. Obviously, these interfaces have different MAC addresses but associated to the same IP address.

Hosts with multiple network interfaces, however, use only one of them to communicate: the other is activated only if the first fails. To avoid that the communications in progress at the time of the failure are interrupted, the second interface is activated and forwards *unicast ARP Reply* which are not solicited by an ARP Request. These ARP Replies have the same IP address of the host, but the MAC address is the one of the second network interface. In this case, the devices that are communicating with the host update their ARP Table and may continue to forward messages if the second interface activation process is completed before the communication is interrupted due to timeout.

In the following we will show how this mechanism can be exploited by a malicious user to perform a MiM attack, because, when not ignored, those replies can result in an indirect attacker-controller manipulation of the content of the victim's ARP Table, which get filled with wrong (MAC, IP) entries containing the victim IP but the **attacker MAC address**.

This manipulation results in the victim unwillingly sending messages to the attacker in place of its intended recipient. The attacker finalizes its MitM position and malicious communication channel by using the same attack also toward the victim's recipient. Ettercap allows to perform all the required steps with no significant effort.

When the intended communication channel uses TLS, this attack fails, since the recipient authenticates itself sending cryptographic signed certificates (e.g., X.509 certificates). If TLS mutual authentication is enabled, both peers send their certificates each other so that both can verify the identity without trusting only the sender MAC address. However, attackers may bypass this scenario by simply applying the following two techniques that leverage on TLS configuration issues:

- If both TLS and non-TLS channels are available, the attacker can use MAC Spoofing to impersonate the recipient during the TLS handshake with the sender while establishing a non-TLS connection with the intended recipient. This is known as the **SSL Strip attack**.

- The attacker can act as a proxy server for the sender. To do so, the attacker has to inject its certificate in the certificate store of the sender. Once this happens, the attacker automatically receives all traffic from the sender. At this point, the attacker can manipulate, forward and/or drop traffic at will.

#### J1.5.2.2 Policy Tampering Attacks

We discovered a vulnerability in the $A1$ policies management flow that allows an attacker to arbitrary overwrite existing $A1$ policies by forging $A1$ policies having an existing `PolicyId`.

We performed an attack by leveraging the *Burp Suite* software. We altered the fields of the `PolicyID` and `PolicyTypeID` in a *policy set request* issued towards the Non-RT RIC to introduce a policy with an existing `PolicyID` (e.g., *"sameID"*) but a different `PolicyTypeID`.

Since the Non-RT RIC hosts the web application which is responsible to elaborate HTTP requests and to interact with the Near-RT RIC if needed, the attack results in the following situation:

- the Non-RT RIC contains two policies with the same `PolicyID`
- only one of them (i.e., the attacker's policy) is stored at the Near-RT RIC.

In other words, with this attack a specific policy can be **intentionally overwritten** by introducing a new malicious policy with the same `PolicyID`.

This situation creates an inconsistency between the Near-RT RIC and the Non-RT RIC which may generate errors in the visualization of policies at the Non-RT RIC and may inhibit the web-user to list all available policies.

### J1.5.3 Blockchain anti-tampering implementation

As described in Section J1.4, the O-RAN Alliance strongly suggests (but not forces) to adopt TLS in the protocol stack of the communication interface. The correct use of recent versions (i.e., version 1.2 and 1.3) of TLS would grant confidentiality, authentication and provide a well-tested environment.

However, there may be issues: a correct configuration of TLS and the setup/maintenance of a proper Public-Key Infrastructure (PKI), may represent an issue in some implementations due to the additional requirements and the latency introduced in the communications. Moreover, the TLS implementation may have vulnerabilities: if on one side, using open-source components also for TLS (e.g., the OpenSSL library) grants a fast discovery and patching of vulnerabilities, on the other side attackers also have a fast access to new vulnerabilities, which may still be exploitable on the O-RAN implementation they want to target, due to delays on the actual auditing and patching. This is not an edge case, as demonstrated by recent OpenSSL vulnerabilities.

In security-critical O-RAN implementations, a redundancy mechanism to protect communication interfaces would grant a second layer of defense in case the deployed security mechanisms fail for some reason. In this context, we propose the use a blockchain-based approach to ensure that an attacker, even having full access to a target communication interface, would fail to forge and abuse the services provided by O-RAN components. We decided to apply our approach in the inter-controller communication interface, (the $A1$ interface) which is used to exchange the messaging related to the management of traffic policies. In order to introduce the usage of a blockchain data structure, we decided to enhance the basic $A1$ Policy structure as described in [75] by adding two top-level entries in the JSON object containing the policy (Figure J1.4):

- A `nonce`, i.e., a $n$-byte long random value;
- The `prevHash`, i.e., the digest of an hash function computed on the last well-formed message received.

The nonce is adopted to increase the entropy of the JSON object, which would be otherwise vulnerable to quasi-plaintext attacks due foreseeable structure of an $A1$ policy.

The `prevHash` is used instead to provide the blockchain structure: this value is the result of an hash function (in our case, SHA256 [95]) computed on the last message and stored in both sender and receiver. Upon the next message, the sender has to provide such value so that the receiver can verify that it matches the one stored. If so, the message is accepted and the stored `prevHash` is updated.

Both fields are encoded using the *Base64* encoding to avoid the presence of syntax-breaking characters.

An attacker wishing to break-in and forging a valid `prevHash` value for a malicious policy, has to know recursively every `prevHash` of all the messages exchanged until that moment, and in particular, the so-called *root* hash, i.e., the `prevHash` value used in the very first message ever exchanged. According to the proposed threat model (Section J1.5.1), we assume that, even if an attacker gains access to the $A1$ interface, he/she has no way to know initial configuration data including the *root* hash. Securing communication at the deployment time is out of the scope of this work and is a well-known problem in the literature.

Not all the message types need the additional defense offered by our approach. Following [102], the $A1$ policy-related message types are shown in Figure J1.5. Among

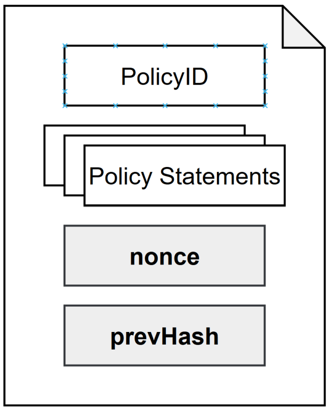
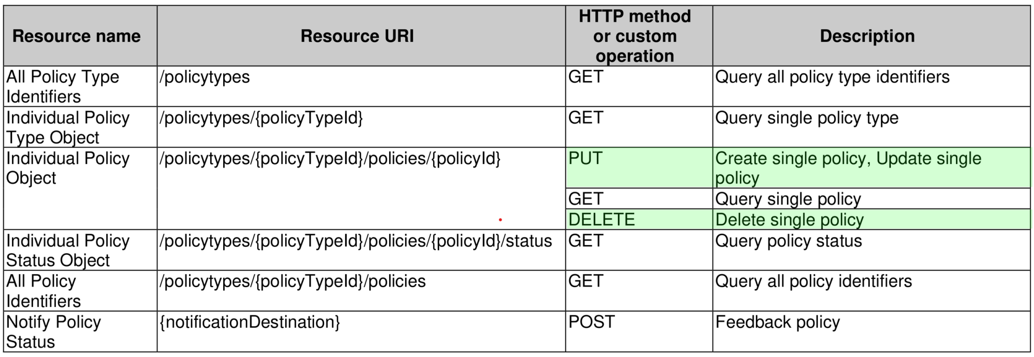


Figure J1.4: Proposed variation in Structure of an $A1$ Policy . The message types of interest are highlighted in green

these, we decided to focus our attention on the message types that do *alter* the current policies on the Near-RT RIC and not those which are used to query and retrieve information. The resulting set of message types of interest is made by the *Policy Creation* message, the *Policy Update* message and the *Policy Deletion* message. The first two are both characterized by the use of the PUT HTTP method and contain an $A1$ policy as payload, so our approach can be applied seamlessly. The latter uses instead the DELETE method and may have no payload, according to [102]. In this case, we provide a JSON payload which contains as top-level fields the PolicyID, a nonce and the prevHash.



| Resource name | Resource URI | HTTP method or custom operation | Description |
|---|---|---|---|
| All Policy Type Identifiers | /policytypes | GET | Query all policy type identifiers |
| Individual Policy Type Object | /policytypes/{policyTypeId} | GET | Query single policy type |
| Individual Policy Object | /policytypes/{policyTypeId}/policies/{policyId} | PUT | Create single policy, Update single policy |
| | | GET | Query single policy |
| | | DELETE | Delete single policy |
| Individual Policy Status Object | /policytypes/{policyTypeId}/policies/{policyId}/status | GET | Query policy status |
| All Policy Identifiers | /policytypes/{policyTypeId}/policies | GET | Query all policy identifiers |
| Notify Policy Status | {notificationDestination} | POST | Feedback policy |

Figure J1.5: $A1$ Policy-related message types.

The last aspect we want to address is the storage of the prevHash values. To let the proposed approach work, only the value of the *last* hash is required, but for verification purposes, it may be useful to store not only the last prevHash but a window of, let's say $M$ previous $A1$ message payloads along with the prevHash that was expected for them. This would provide a way to do a preliminary blockchain verification on the last $M$ messages that would reduce the need to fully verify the consistency of the blockchain at every incoming message.

To summarize, this approach avoids $A1$ policy tampering from attackers even when there are no additional security measures. It grants *immutability* to the whole sequence of messages, giving the possibility to O-RAN operators to verify the correctness of the blockchain *a-posteriori*, e.g., for auditing purposes.

## J1.6 Experimental Results

### J1.6.1 Experimentation Setups

In order to validate the effectiveness of the proposed approach in protecting the $A1$ interface against tampering attacks, we set up two experimentation setups:

1. A preliminary setup based on a simulated $A1$ interface with HTTP REST APIs implementing the minimal set of endpoints and services required by the O-RAN specification, according the $A1$ Application Protocol [102];

2. A setup involving the official O-RAN reference implementation.

The first setup consist of two parts: a Flask-based Python web Server that is capable to recognize and parse the policy-related messages whose target REST endpoints and formats are defined according to [103]; an automated web client that performs multiple HTTP requests to the REST endpoints, each containing an $A1$ Policy in JSON format.

The second setup is instead an O-RAN implementation, based on the Release "F" of the O-RAN software [70]. Such implementation consists in a Near-RT RIC, an SMO/Non-RT RIC and the major interfaces involved, in particular, the $A1$ interface. In this setup, the RICs are implemented as containers in a Kubernetes cluster deployed in a single physical server. In compliance with our threat model (Section J1.5.1), we included an attacker VM in the cluster network so that it can reach both the O-RAN controllers. We configured the attacker's VM with the static IP address `10.10.99.215`.

With the following experiments, our objective is to 1) demonstrate that the lack of a strong authentication over the $A1$ interface may lead to various types of attacks; 2) Once an attacker has access to the $A1$ interface, tampering $A1$ policies is trivial and may cause additional damage; 3) the proposed blockchain-based approach helps reduce (or even stop) an attacker trying to introduce malicious $A1$ policies with a limited latency cost in most cases.

### J1.6.2 Experiment 1 - Man-in-the-middle Attack

In this experiment we aimed to determine how effective are Man-in-the-middle attacks (MitM) on the $A1$ interface when the attacker already gained access to it (as defined in the threat model in Section J1.5.1). To perform such an attack, we deployed the experimentation setup based on the official implementation provided by the O-RAN Alliance. Among the possible MitM attacks, we decided to exploit the Cache Poisoning attacks (CPA) on Address Resolution Protocol (ARP).

We managed to successfully perform an ARP CPA attack to the $A1$ interface taking the control of the communication and being virtually able to tamper messages exchanged by the Non-RT RIC and the Near-RT RIC or just forward original messages, being completely transparent to the existing infrastructure.

It is worth noting that a properly setup of the communication between the two controllers on the $A1$ interface may be secured through TLS. If the mutual authentication mechanism is enabled, both sender and receiver have to check received data against certificates (e.g.: X.509). Doing so, MAC spoofing is not enough for a successful attack, since the attacker has to provide a valid (and properly signed) certificate. It should be noted that some TLS setup may be attacked through specific techniques, e.g., acting like a proxy in the middle of communication. Those attacks are known in the literature as *SSL Strip* attacks.

Our MITM attack is done exploiting Ettercap [101], a software designed to run MITM attacks. We set up Ettercap to attack both controllers and take control of the communication. The attack performs the following actions (Figure J1.6):

1. Targeting the Non-RT RIC, ARP Gratuitous Replies are sent by the attacker with a spoofed source address to match an address of a trusted node of the network. The content of the ARP Gratuitous Replay contains the IP address of the Near-RT RIC but the MAC address of the attacker;

2. the Non-RT RIC discovers that received addresses differ from the Near-RT entry in its ARP table. So, the ARP entry is updated and the new addresses are stored into the ARP table;

3. The attacker changes the target, sending ARP Gratuitous Replies to the Near-RT RIC. The content of the message has the IP address of the Non-RT RIC but the MAC address of the attacker;

4. In response of those messages, the Near-RT RIC updates the Non-RT ARP entry in its ARP table to match the new address;

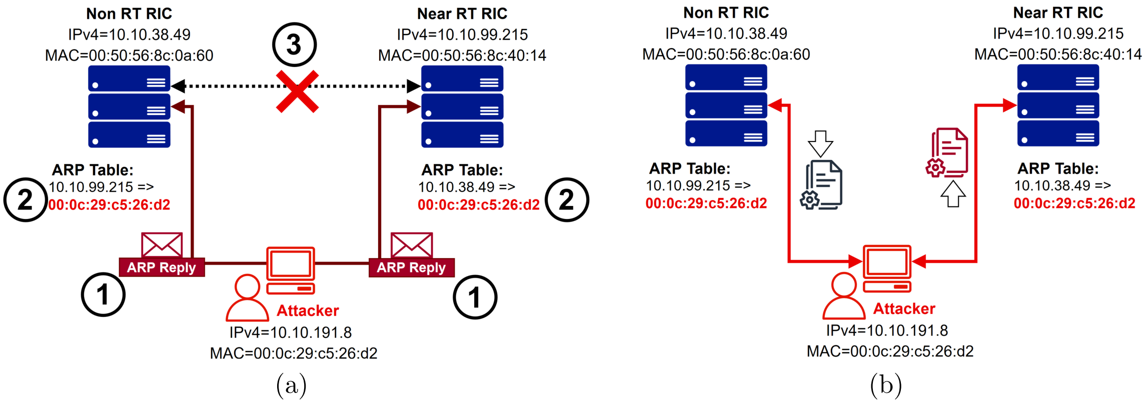
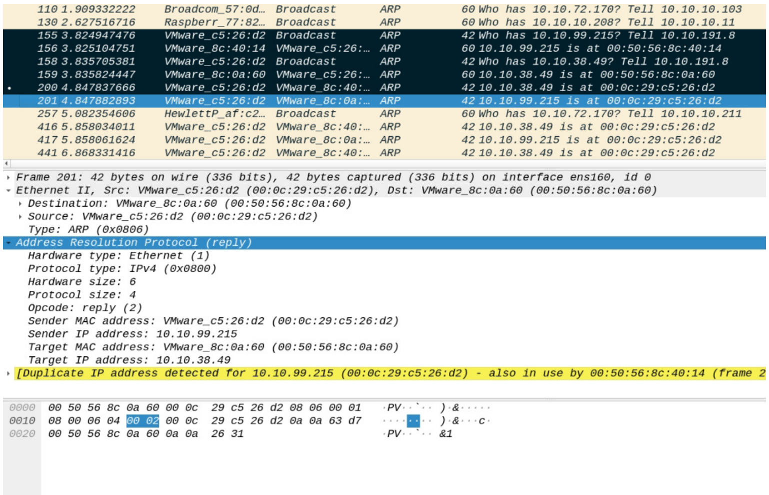


Figure J1.6: ARP Cache Poisoning attack: a) Manipulation of ARP Tables via ARP Gratuitous Replies; b) (after the attack) Malicious link established

5. At this point, the communication is compromised: the Non-RT RIC will send to the attacker all the traffic directed to the Near-RT RIC and the Near-RT RIC will send to the attacker all the traffic directed to the Non-RT RIC. The attacker can just forward traffic (remaining transparent) or it can decide to tamper the content of the messages, drop them or create new ones in both directions.

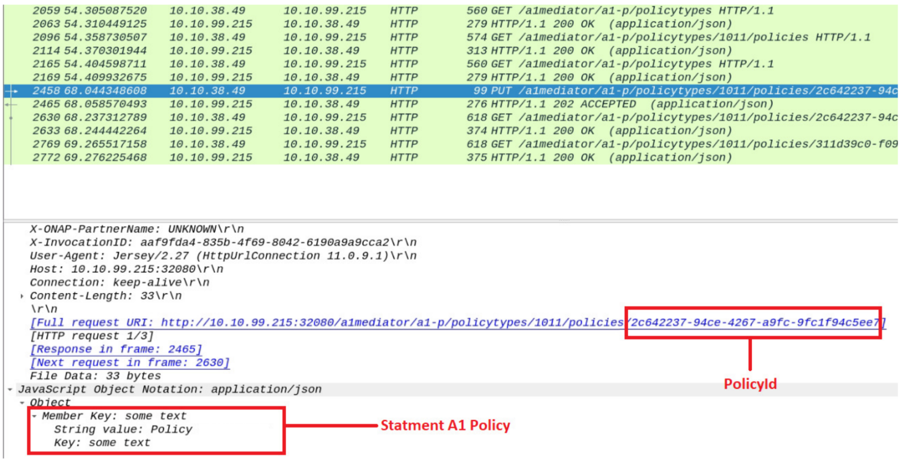


```
  110 1.909332222   Broadcom_57:0d… Broadcast     ARP        60 Who has 10.10.72.170? Tell 10.10.10.103
  130 2.627516716   Raspberr_77:82… Broadcast     ARP        60 Who has 10.10.10.208? Tell 10.10.10.11
  155 3.824947476   VMware_c5:26:d2 Broadcast     ARP        42 Who has 10.10.99.215? Tell 10.10.191.8
  156 3.825104751   VMware_8c:40:14 VMware_c5:26:… ARP       60 10.10.99.215 is at 00:50:56:8c:40:14
  158 3.835705381   VMware_c5:26:d2 Broadcast     ARP        42 Who has 10.10.38.49? Tell 10.10.191.8
  159 3.835824447   VMware_8c:0a:60 VMware_c5:26:… ARP       60 10.10.38.49 is at 00:50:56:8c:0a:60
• 200 4.847837666   VMware_c5:26:d2 VMware_8c:40:… ARP       42 10.10.38.49 is at 00:0c:29:c5:26:d2
  201 4.847882893   VMware_c5:26:d2 VMware_8c:0a:… ARP       42 10.10.99.215 is at 00:0c:29:c5:26:d2
  257 5.082354606   HewlettP_af:c2… Broadcast     ARP        60 Who has 10.10.72.170? Tell 10.10.10.211
  416 5.858034011   VMware_c5:26:d2 VMware_8c:40:… ARP       42 10.10.38.49 is at 00:0c:29:c5:26:d2
  417 5.858061624   VMware_c5:26:d2 VMware_8c:0a:… ARP       42 10.10.99.215 is at 00:0c:29:c5:26:d2
  441 6.868331416   VMware_c5:26:d2 VMware_8c:40:… ARP       42 10.10.38.49 is at 00:0c:29:c5:26:d2

› Frame 201: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface ens160, id 0
˅ Ethernet II, Src: VMware_c5:26:d2 (00:0c:29:c5:26:d2), Dst: VMware_8c:0a:60 (00:50:56:8c:0a:60)
  › Destination: VMware_8c:0a:60 (00:50:56:8c:0a:60)
  › Source: VMware_c5:26:d2 (00:0c:29:c5:26:d2)
    Type: ARP (0x0806)
˅ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: VMware_c5:26:d2 (00:0c:29:c5:26:d2)
    Sender IP address: 10.10.99.215
    Target MAC address: VMware_8c:0a:60 (00:50:56:8c:0a:60)
    Target IP address: 10.10.38.49
› [Duplicate IP address detected for 10.10.99.215 (00:0c:29:c5:26:d2) - also in use by 00:50:56:8c:40:14 (frame 2

0000  00 50 56 8c 0a 60 00 0c  29 c5 26 d2 08 06 00 01   ·PV··`·· )·&····
0010  08 00 06 04 00 02 00 0c  29 c5 26 d2 0a 0a 63 d7   ········ )·&··c·
0020  00 50 56 8c 0a 60 0a 0a  26 31                     ·PV··`·· &1
```

Figure J1.7: Capture of the ARP messages generated by Ettercap.

Now, using Wireshark software we verify the exchange of $A1$ policies between the Non-RT RIC and the Near-RT RIC. We observe the traffic passing through the attacking machine's interface (identified by the label "ens160"). We filter the packets related to the HTTP protocol sent or received by the Near-RT RIC, and see that a policy belonging to policy type "1011" is created on the Non-RT RIC, using the

“Control Panel” web-application, present on the Non-RT RIC itself. This policy contains only a statement with the string “Policy”. Fig. J1.8 shows the content of



```
2059 54.305087520   10.10.38.49    10.10.99.215   HTTP     560 GET /a1mediator/a1-p/policytypes HTTP/1.1
2063 54.310449125   10.10.99.215   10.10.38.49    HTTP     279 HTTP/1.1 200 OK  (application/json)
2096 54.358730507   10.10.38.49    10.10.99.215   HTTP     574 GET /a1mediator/a1-p/policytypes/1011/policies HTTP/1.1
2114 54.370301944   10.10.99.215   10.10.38.49    HTTP     313 HTTP/1.1 200 OK  (application/json)
2165 54.404598711   10.10.38.49    10.10.99.215   HTTP     560 GET /a1mediator/a1-p/policytypes HTTP/1.1
2169 54.409932675   10.10.99.215   10.10.38.49    HTTP     279 HTTP/1.1 200 OK  (application/json)
2458 68.044348608   10.10.38.49    10.10.99.215   HTTP      99 PUT /a1mediator/a1-p/policytypes/1011/policies/2c642237-94c
2465 68.058570493   10.10.99.215   10.10.38.49    HTTP     276 HTTP/1.1 202 ACCEPTED  (application/json)
2630 68.237312789   10.10.38.49    10.10.99.215   HTTP     618 GET /a1mediator/a1-p/policytypes/1011/policies/2c642237-94c
2633 68.244442264   10.10.99.215   10.10.38.49    HTTP     374 HTTP/1.1 200 OK  (application/json)
2769 69.265517158   10.10.38.49    10.10.99.215   HTTP     618 GET /a1mediator/a1-p/policytypes/1011/policies/311d39c0-f09
2772 69.276225468   10.10.99.215   10.10.38.49    HTTP     375 HTTP/1.1 200 OK  (application/json)

    X-ONAP-PartnerName: UNKNOWN\r\n
    X-InvocationID: aaf9fda4-835b-4f69-8042-6190a9a9cca2\r\n
    User-Agent: Jersey/2.27 (HttpUrlConnection 11.0.9.1)\r\n
    Host: 10.10.99.215:32080\r\n
    Connection: keep-alive\r\n
  › Content-Length: 33\r\n
    \r\n
    [Full request URI: http://10.10.99.215:32080/a1mediator/a1-p/policytypes/1011/policies/2c642237-94ce-4267-a9fc-9fc1f94c5ee7]
    [HTTP request 1/3]
    [Response in frame: 2465]
    [Next request in frame: 2630]
    File Data: 33 bytes
▾ JavaScript Object Notation: application/json
  ▾ Object
    ▾ Member Key: some text
        String value: Policy
        Key: some text
```

Figure J1.8: Capture of messages for the creation of a *A*1 policy during the confusion attack.

the creation of the *A*1 policy and its identifier passed in the HTTP PUT request equal to “`2c642237-94ce-4267-a9fc-9fc1f94c5ee7`.” The blue-highlighted packet reports the **HTTP PUT request** with the first *IP address* is the Non-RT RIC and the second one is the Near-RT RIC. The red boxes show the Policy ID and the statement.

### J1.6.3 Experiment 2 - Policy Tampering Attacks

In this experiment we aim at discovering possible vulnerabilities in the *A*1 policies management flow and evaluate the effectiveness of the injection of malicious *A*1 policies. In particular, we discovered a vulnerability that allows an attacker to arbitrary overwrite existing *A*1 policies by forging *A*1 policies having an existing `PolicyID`. By exploiting the *Burp Suite* we perform an attack consisting in:

- altering the fields of the `PolicyID` and `PolicyTypeID` in a policy set request towards the Non-RT RIC
- introducing a policy with an existing `PolicyID` (e.g., *”sameID”*) but a different `PolicyTypeID`.

The outcome of the attack is that while the Non-RT RIC contains two policies with the same `PolicyID`, only one (i.e., the attacker’s policy) is stored at the Near-RT RIC. Thus, this attack can be utilized to intentionally overwrite a specific policy by introducing a new malicious policy with the same `PolicyID`. Nonetheless, the attack creates a misalignment between the Near-RT RIC and the Non-RT RIC. This results in the generation of errors in the visualization of policies in the control panel web-application at the Non-RT RIC and may inhibit the web-user to list all available policies.

In addition to the above described experiment, we wanted to analyze the effect of introducing $A1$ policies with `PolicyID` and `PolicyTypeID` values non well-formed. In particular, we tested the effect of the introduction in this fields of special characters (i.e., ;,?, and %) since these values assume special meaning when utilized into a URI.

By utilizing the *Burp Suite* we send $A1$ altered policies via HTTP PUT requests towards the control panel application. Our test show that for the `PolicyTypeID` field only numeric values are accepted.

We test the `PolicyID` field by sending an $A1$ policy specifying `PolicyTypeID` 1000 and the string ”`policy;`*SpecialCharacter*” as `PolicyID` value. In this case, we get as response to the HTTP PUT request a HTTP response with code ”201 Created”.

We observe the following consequences as a result of the attack:

- when trying to retrieve policies with `PolicyTypeID`=1000 we get a 404 message. This happens because the application is not able to handle the “;” special character correctly

- when trying to visualize policy information via control panel we get an error message and no data is shown. This consequence is particularly relevant because it inhibits the user to access to any policies having `PolicyTypeID`=1000

- at the Near-RT RIC side an unstable behavior is observed since it shows the list of installed policies containing randomly: (i) only non-malicious policies; (ii) only the malicious policy; (iii) both malicious and non-malicious policies;

### J1.6.4 Experiment 3 - Blockchain-based Approach

In this experiment the attacker aims to forge and push malicious (but well-formed) traffic policies on the $A1$ interface while spoofing its identity as the Non-RT RIC. Given the adopted threat model (see Section J1.5.1), the destination (i.e., the Near-RT RIC) has no way to distinguish whether the policy source is the Non-RT RIC or not. So, the policy is likely to be added (or replaced if a similar policy is already present) in the Near-RT RIC without any issue, meaning that the attacker has successfully achieved its objective (Figure J1.9).

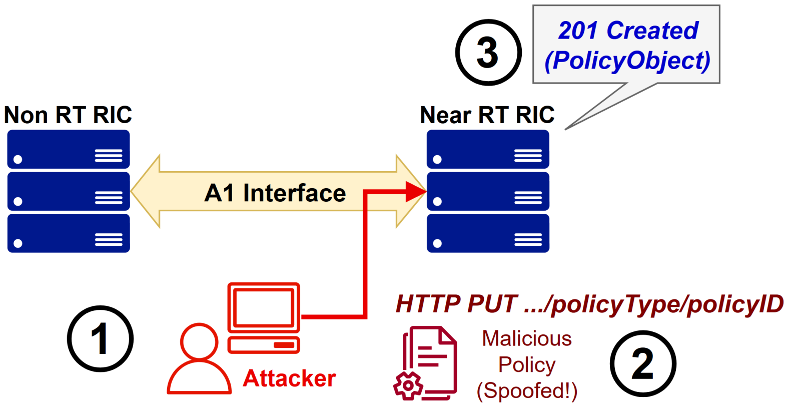


Figure J1.9: Policy Injection Attack.

In order to avoid such a scenario, we experimented the proposed approach (as described in Section J1.5) in both the experimentation setups.

The result is shown in Figure J1.10. First, the attacker forges a malicious policy (1). According the adopted threat model, he/she knows that now a `prevHash` field is required for an well-formed policy but, lacking the knowledge of the previous messages hashes (in particular, the *root* hash decided at the configuration time) and without any additional hint/leaked information, the best the attacker can do is to make the best possible guess for `prevHash` field and send the *A*1 policy through an HTTP PUT request to the Near-RT at the policy URI (2). In the meantime, the Near-RT has stored the `prevHash` value computed on the last valid policy-related message (3). Upon arrival of a new policy message (e.g., a Policy Creation message), the policy is parsed and the `prevHash` is retrieved. Even if the attacker has captured the previous message, the probability he/she guessed the hash causing a collision is negligible, hence the comparison of the stored hash and the hash provided is likely to fail. At this point, the Near-RT **refuses** the policy and provides an HTTP Response with an error (4).

In the first experimentation setup, we instructed the Flask Server to check and parse the `prevHash` field for every policy creation, update and delete message. A packet capture of the policy creation message is shown in Figure J1.11 while the JSON payload (with the additional fields introduced by the proposed approach) is show in Figure J1.12.

Whether one of such message is successfully parsed and the `prevHash` checked against the stored hash, the latter is updated with the hash computed on the new (accepted) message (Figure J1.13).

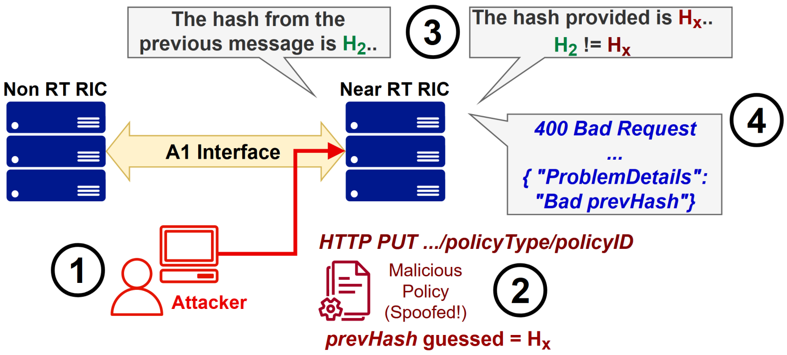


Figure J1.10: Proposed Approach in action.

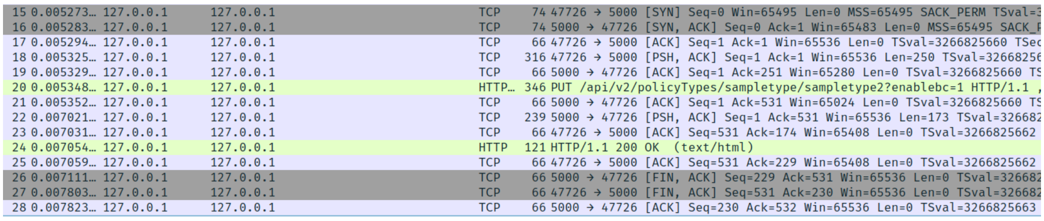

```
15 0.005273… 127.0.0.1       127.0.0.1       TCP      74 47726 → 5000 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM TSval=3
16 0.005283… 127.0.0.1       127.0.0.1       TCP      74 5000 → 47726 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_P
17 0.005294… 127.0.0.1       127.0.0.1       TCP      66 47726 → 5000 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=3266825660 TSec
18 0.005325… 127.0.0.1       127.0.0.1       TCP     316 47726 → 5000 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=250 TSval=32668256
19 0.005329… 127.0.0.1       127.0.0.1       TCP      66 5000 → 47726 [ACK] Seq=1 Ack=251 Win=65280 Len=0 TSval=3266825660 TS
20 0.005348… 127.0.0.1       127.0.0.1       HTTP…   346 PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=1 HTTP/1.1 ,
21 0.005352… 127.0.0.1       127.0.0.1       TCP      66 5000 → 47726 [ACK] Seq=1 Ack=531 Win=65024 Len=0 TSval=3266825660 TS
22 0.007021… 127.0.0.1       127.0.0.1       TCP     239 5000 → 47726 [PSH, ACK] Seq=1 Ack=531 Win=65536 Len=173 TSval=326682
23 0.007031… 127.0.0.1       127.0.0.1       TCP      66 47726 → 5000 [ACK] Seq=531 Ack=174 Win=65408 Len=0 TSval=3266825662
24 0.007054… 127.0.0.1       127.0.0.1       HTTP    121 HTTP/1.1 200 OK  (text/html)
25 0.007059… 127.0.0.1       127.0.0.1       TCP      66 47726 → 5000 [ACK] Seq=531 Ack=229 Win=65408 Len=0 TSval=3266825662
26 0.007111… 127.0.0.1       127.0.0.1       TCP      66 5000 → 47726 [FIN, ACK] Seq=229 Ack=531 Win=65536 Len=0 TSval=326682
27 0.007803… 127.0.0.1       127.0.0.1       TCP      66 47726 → 5000 [FIN, ACK] Seq=531 Ack=230 Win=65536 Len=0 TSval=326682
28 0.007823… 127.0.0.1       127.0.0.1       TCP      66 5000 → 47726 [ACK] Seq=230 Ack=532 Win=65536 Len=0 TSval=3266825663
```

Figure J1.11: Experiment 3: Wireshark's packet capture of a policy creation/update message (HTTP PUT)

Figure J1.14 shows the server reaction to a failure in the `prevHash` check.

We took the time required by the server to perform such operations $N$ times (e.g., $N = 100$) while varying the size of the policy contained as payload in the messages. To maintain the policies well-formed, such size variation has been implemented by incrementing the number of *Policy Statements* contained (i.e., $T$), from a single statement ($T = 1$) up to a selectable limit (e.g., in the performed experiments $T <= 100$). [2]

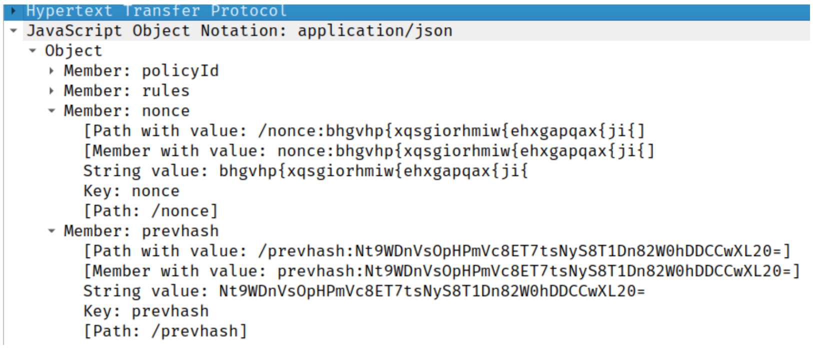
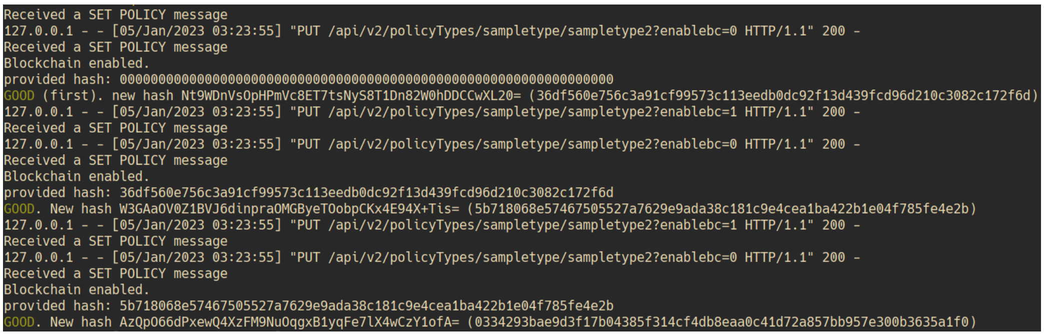

```
▸ Hypertext Transfer Protocol
▾ JavaScript Object Notation: application/json
  ▾ Object
    ▸ Member: policyId
    ▸ Member: rules
    ▾ Member: nonce
        [Path with value: /nonce:bhgvhp{xqsgiorhmiw{ehxgapqax{ji{]
        [Member with value: nonce:bhgvhp{xqsgiorhmiw{ehxgapqax{ji{]
        String value: bhgvhp{xqsgiorhmiw{ehxgapqax{ji{
        Key: nonce
        [Path: /nonce]
    ▾ Member: prevhash
        [Path with value: /prevhash:Nt9WDnVsOpHPmVc8ET7tsNyS8T1Dn82W0hDDCCwXL20=]
        [Member with value: prevhash:Nt9WDnVsOpHPmVc8ET7tsNyS8T1Dn82W0hDDCCwXL20=]
        String value: Nt9WDnVsOpHPmVc8ET7tsNyS8T1Dn82W0hDDCCwXL20=
        Key: prevhash
        [Path: /prevhash]
```

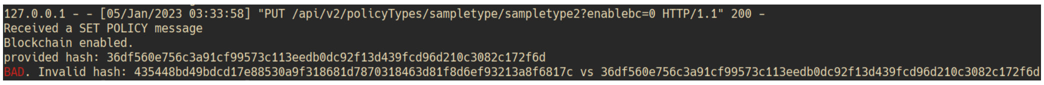
Figure J1.12: Experiment 3: application of the proposed approach on structure of an *A*1 Policy (Wireshark)

[2] the resulting policy size (in bytes) ranged from ∼300 bytes ($T = 1$) to ∼20 Kilobytes ($T = 100$)



```
Received a SET POLICY message
127.0.0.1 - - [05/Jan/2023 03:23:55] "PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=0 HTTP/1.1" 200 -
Received a SET POLICY message
Blockchain enabled.
provided hash: 0000000000000000000000000000000000000000000000000000000000000000
GOOD (first). new hash Nt9WDnVsOpHPmVc8ET7tsNyS8T1Dn82W0hDDCCwXL20= (36df560e756c3a91cf99573c113eedb0dc92f13d439fcd96d210c3082c172f6d)
127.0.0.1 - - [05/Jan/2023 03:23:55] "PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=1 HTTP/1.1" 200 -
Received a SET POLICY message
127.0.0.1 - - [05/Jan/2023 03:23:55] "PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=0 HTTP/1.1" 200 -
Received a SET POLICY message
Blockchain enabled.
provided hash: 36df560e756c3a91cf99573c113eedb0dc92f13d439fcd96d210c3082c172f6d
GOOD. New hash W3GAaOV0Z1BVJ6dinpraOMGByeTOobpCKx4E94X+Tis= (5b718068e57467505527a7629e9ada38c181c9e4cea1ba422b1e04f785fe4e2b)
127.0.0.1 - - [05/Jan/2023 03:23:55] "PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=1 HTTP/1.1" 200 -
Received a SET POLICY message
127.0.0.1 - - [05/Jan/2023 03:23:55] "PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=0 HTTP/1.1" 200 -
Received a SET POLICY message
Blockchain enabled.
provided hash: 5b718068e57467505527a7629e9ada38c181c9e4cea1ba422b1e04f785fe4e2b
GOOD. New hash AzQpO66dPxewQ4XzFM9NuOqgxB1yqFe7lX4wCzY1ofA= (0334293bae9d3f17b04385f314cf4db8eaa0c41d72a857bb957e300b3635a1f0)
```

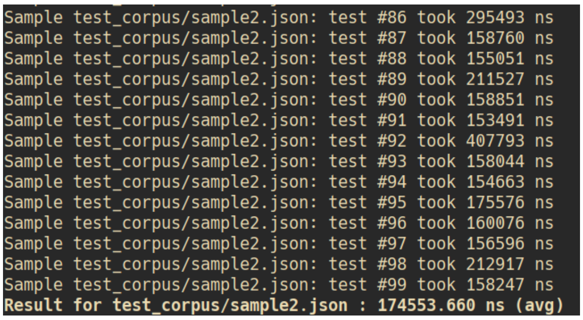
Figure J1.13: Experiment 3 (server): parsing and checking of incoming $A1$ policies



```
127.0.0.1 - - [05/Jan/2023 03:33:58] "PUT /api/v2/policyTypes/sampletype/sampletype2?enablebc=0 HTTP/1.1" 200 -
Received a SET POLICY message
Blockchain enabled.
provided hash: 36df560e756c3a91cf99573c113eedb0dc92f13d439fcd96d210c3082c172f6d
BAD. Invalid hash: 435448bd49bdcd17e88530a9f318681d7870318463d81f8d6ef93213a8f6817c vs 36df560e756c3a91cf99573c113eedb0dc92f13d439fcd96d210c3082c172f6d
```

Figure J1.14: Experiment 3 (server): example of a `prevHash` check failure

Figure J1.15 shows the latency computed by the server and sent as response to the client, which parse and average the values after all the tests for a given value of $N$ are completed.



```
Sample test_corpus/sample2.json: test #86 took 295493 ns
Sample test_corpus/sample2.json: test #87 took 158760 ns
Sample test_corpus/sample2.json: test #88 took 155051 ns
Sample test_corpus/sample2.json: test #89 took 211527 ns
Sample test_corpus/sample2.json: test #90 took 158851 ns
Sample test_corpus/sample2.json: test #91 took 153491 ns
Sample test_corpus/sample2.json: test #92 took 407793 ns
Sample test_corpus/sample2.json: test #93 took 158044 ns
Sample test_corpus/sample2.json: test #94 took 154663 ns
Sample test_corpus/sample2.json: test #95 took 175576 ns
Sample test_corpus/sample2.json: test #96 took 160076 ns
Sample test_corpus/sample2.json: test #97 took 156596 ns
Sample test_corpus/sample2.json: test #98 took 212917 ns
Sample test_corpus/sample2.json: test #99 took 158247 ns
Result for test_corpus/sample2.json : 174553.660 ns (avg)
```

Figure J1.15: Experiment 3 (client): latency retrieval for $T = 2$

The introduction of the proposed anti-tampering blockchain-based mechanism may introduce additional delay in the transmission and elaboration of $A1$ policies. Such delay might impact design decisions in the deployment and placement of O-RAN components. In order to evaluate this delay, we compare the time required to apply $A1$ policies with and without the proposed blockchain (BC) based approach.

We perform such analysis by varying the number of policy statements in the policy. The resulting average time obtained for every policy size considered is shown in Figure J1.16. Results show that in both cases the latency increases linearly with the number of policy statements. For the BC case, the delay increases more rapidly due to the burden related to BC operations. However, higher penalty is payed when the number of policy statements is particularly high, which represents a condition less likely to occur. Nonetheless, it is worth noticing that the additional delay remains in the order of tens of microseconds. This represents a negligible contribution compared to the delay observable in a real deployment with Near-RT and Non-RT distributed over different network nodes where such delay is expected to be in the order of milliseconds.

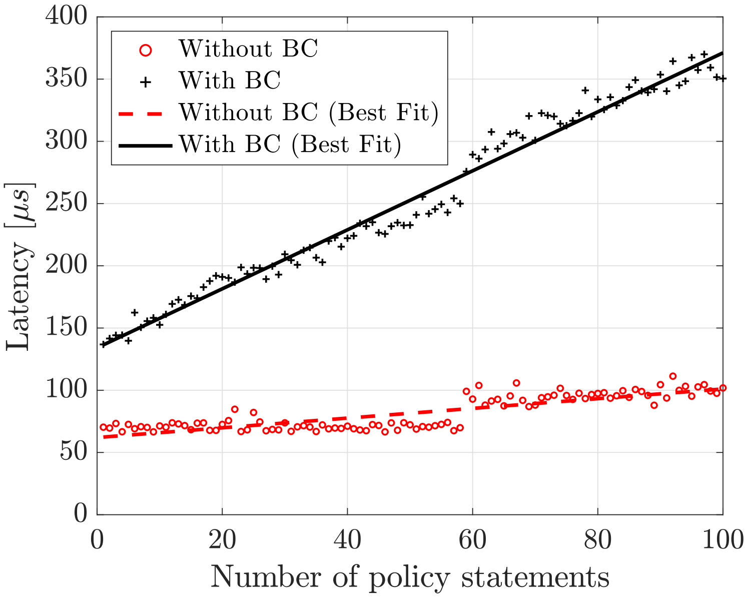


Figure J1.16: Latency comparison.

## J1.7 Conclusion

O-RAN is a recent project which is still under development on many aspects. Despite to this, it already offers a innovative reference architecture which uses state-of-the-art open components. The adoption of open-source components allows O-RAN to have a quick development and evolution. This aspect, however, may also represent an issue, in particular for security: implementors have to provide a fast response to the constantly evolving attack surface and to new vulnerabilities.

In this work, we provided an overview of the state-of-the-art of O-RAN security in the literature and an analysis of the current version of the architectural specification of O-RAN, in particular on security. The specification offers a valid set of security protocols and techniques, although we noticed that it is still not complete, it contains inconsistencies and leaves many non-trivial details to arbitrary implementation choices. In particular, in this work we focus on the inter-controller communication interface, the $A1$ interface, used to manage the traffic policies. We demonstrate that, under some assumptions, an attacker can execute a Man-in-the-Middle attack and take control of the communication interface, tamper communication or forge malicious policies without being detected.

In light of this, we proposed a blockchain-based approach for enhancing the $A1$ communications by adding an additional security layer to existing security measures (e.g., TLS) which grants immutability of policies, anti-tampering features and useful auditing information.

We have validated the Man-in-the-middle attacks and the blockchain-based approach with three experiments across two different experimentation scenarios, one of which is the current version of the official O-RAN implementation. The results show that, while the attacks are feasible, the use of the proposed approach poses an important restriction to the attacker capabilities in manipulating traffic policies with a negligible cost in terms of the latency due to the additional processing.

We are currently working on evaluating the security of the other subsystems and communication interfaces of O-RAN (e.g., $O1$), while performing more complex experiments using the official O-RAN implementation such as evaluating attacks on fully secured implementations and by exploiting vulnerabilities coming from the virtualization and containerization solutions on which O-RAN is based.

# Acknowledgments

This work was partially supported Italian Government under MiSE "Programma di supporto tecnologie emergenti - Asse I (Casa delle Tecnologie Emergenti) Progetto SICURA" - CUP C19C20000520004. Part of this work has been presented to the 2022 5th IEEE Future Networks World Forum. The authors wish to thank Mr. Antonio Di Luca for his contributions in setting up the O-RAN official simulator tool.

# Journal J2

# On the Exploitation of 5G Multi-Access Edge Computing for Spatial Audio in Cultural Heritage Applications

C. Rinaldi, F. Franchi, A. Marotta, F. Graziosi, C. Centofanti

Submitted to
*IEEE Access*

# Abstract

This work presents a service for the improvement of cultural heritage experiences, which exploits the advantages coming from the 5G paradigm. Indeed, in a scenario where many users need to be served by a real-time solution which is in turn required to work on different devices, the potentialities of 5G technology show their suitability. In particular, moving the computation to the edge of the network ensures the availability of resources needed for binaural spatial audio rendering in an independent fashion with reference to the client device and at the same time it guarantees real-time availability of this data since the core network, with its impairments, is not involved. This work demonstrates how 5G could be a critical enabler for delivering low latency services at guaranteed levels, data-centric services, differentiated customer experiences, improved security and reduced costs to the users.

## J2.1 Introduction

The 5$^{\text{th}}$ generation of mobile network promises great changes that may lead to a different way of understanding mobile communications, moving from the need of connecting people to the need of connecting their worlds. The business ecosystem around this paradigm is involving many different vertical industries on various fields.

Three categories of services can be identified for 5G services, [104]:

- **enhanced Mobile Broadband** (eMBB) that aims to support high bandwidth demanding services;
- **ultra Reliable and Low Latency Communications** (uRLLC) has been introduced in order to cope with safety and mission critical services by guaranteeing high reliability and low latency communications;
- **massive Machine Type Communications** (mMTC) as enabler for Internet of Things (IoT) services which require high density of connected devices.

Vertical services exploited by 5G trials should include media and entertainment, public safety, e-health, automotive, transport and logistic, Cultural Heritage (CH), [105]. The latter, in particular, has been significantly affected by the progress of digital information especially conceiving its dissemination [106], offering new technological possibilities for developing, e.g. the market of tourist services [107], and CH organizations have to address new user needs by creating innovative applications [108], such as Augmented and Virtual Reality (AR/VR) based. AR technology gives a different perception of reality, as it enriches reality with a computer-generated layer containing visual, audio, and tactile information, while using a "virtual" representation of a classic museum allows access to aspects of the artifact that may otherwise be hidden [109].

During the past years, the main aim of AR/VR applications for CH changed from a mere virtual recreation of object to display, to create an entire virtual environment able to disseminate and teach culture. The idea is the opposite of a "dead museum": users must not be exposed to an accumulation of 3D heritage objects, but feel and understand another culture through those items. An important aspect is the relation between AR/VR and education. This new way to present Culture enhances the learning process, encouraging students and researchers through stimulating methods of presentation of archival materials and historical events. Users can therefore travel through space and time without moving from their home [110]. Numerous AR/VR applications exists for CH or tourists enjoyment of places with a rich past, allowing a realistic navigation of environments that no longer exist or that may be inaccessible, [111], [112], [113], [114], [115].

The most of the previously cited experiences do not take into account the advantages that may arise by a proper exploitation of sounds together with visual effects, except for audio content presentation purposes. On the contrary, an acoustic guide, properly placed in the virtual space, may drive the user toward a certain direction or the acoustic landscape of a specific historical period could be reproduced to improve the virtual experience. To the authors' knowledge only a few experimentation have been carried out in this context.

The presented service implements spatial audio rendering through 5G in order to exploit possible advantages such as the reduction of the computational requirements for the local devices and the increase of the amount of users that may require the same service on smaller and lighter AR/VR devices.

### J2.1.1 Motivation and contributions

5G technology foresees impressive numbers as guaranteed latency of some milliseconds and throughput higher than $1Gb/s$, [116], which may perfectly fit the requirements coming from the exploitation of AR/VR solutions for CH.

To be more specific, the need of a binaural sound rendering solution to act in real time as the user moves the head or changes the position in time as well as the addition of video streaming to the audio, would require ultra-low latency and high throughput for a proper experience.

Moreover, being aware that the computational requirements of real time adaptation of both audio and video streaming are high, the scenario under analysis may also benefit from a new born paradigm, in the context of 5G networks, known as Multi Access Edge Computing (MEC). A MEC approach in such context has also the advantage of allowing to offload the multimedia elaboration to the edge of the network, making the user devices less complex and power hungry.

MEC enables services and applications to be hosted 'on top' of the mobile network, i.e. above the network layer. These services and applications can benefit from being in close proximity to the users and from receiving local connectivity enabling new business opportunities. In malls, [117], university areas, [118], or museums, [119] that are filled with high-value users, 5G MEC can provide value added services, such as local cache service, location service, and targeted advertising. At business campuses, factories, and seaports, 5G MEC can provide enterprise-level services, such as virtual private networks, service hosting, and dedicated applications. Due to the possibility to drastically reduce experienced latency and offloading computation to the edge, MEC has emerged as an enabler for a wide range novel services including Industial IoT [120], low-latency mission critical applications [121], vehicular communications [122], and multimedia services [123].

Referring to the proposed scenario, by admitting edge computing, the limit of $40ms$ between head movement and spatialized sound, foreseen in [124] as the limit above which a processing/transmission delay is perceivable, can be also exploited for improving signal processing algorithms given that the communication is performed on a network responding to the uRLLC paradigm.

Advantages coming from exploitation of 5G for AR/VR for CH thus need to be deepened because 5G CH AR/VR could represent a possible killer application for this communication technology.

This paper focuses on the exploitation of audio spatialization service, supported by the 5G network architecture, for improving AR/VR experiences for CH. Here follows a list of the main topics involved in the scenario under analysis that will be discussed in the remaining of the paper:

- exploitation of *Resonanace Audio* for a CH AR/VR application scenario;
- definition of a 5G based MEC architecture for AR/VR support;
- analysis of advantages achievable through the proposed solution.

Moreover, some experiences related to exploitation of 5G for AR/VR applications were already conducted by the authors within the 5G trial carried out in L'Aquila, [125].

It is worth noting that, in this phase of the research, we assume a perfect and reliable behavior of the eventual head orientation localization system, i.e. we do not assume head tracking and thus the issues that may arise such as adaptation of the binaural sound to these movements. We are anyway confident of the proper behavior of the orientation/localization system on the specific device we are using for AR. Moreover, we do not face the problem of active noise cancellation for excluding sounds from the real word since our aim is to exploit lo-fi devices.

The remaining of the paper is organized as follows: in Section J2.2 the SoA on the various topics involved is presented; Section J2.3 sketches the scenario under analysis; details on the advantages of MEC for the previously described scenario are given in Section J2.4; Section J2.5 discusses about the referred spatialization tools while the application is described in Section J2.6. Performance analysis and results are reported in Section J2.7 while conclusions are drawn in Section J2.8.

## J2.2 Previous works

Real time binaural rendering solutions for AR/VR enjoyment of CH by exploiting 5G solutions is a topic involving various research aspects. In this section a brief state of the art of the most relevant topics involved in the project is presented.

### J2.2.1 Spatial audio for AR/VR CH applications

As previously stated, the importance of sounds in the context of AR/VR for CH is not yet understood and the research literature is quite poor.

One of the few exceptions is given by [126], in which authors present a signal processing method for fast real–time binaural synthesis, whose main target application is the fruition of cultural heritage and in [127] where a smart headphones set is presented in order to remotely take the listener's head orientation and properly generate an audio output to attract tourists' attention toward specific points of interest in the 3D space. An interesting analysis of hardware and software requirements for this purpose is presented in [128] without references to real applications.

In [129], authors propose an interaction system for attracting the visitor toward specific cultural attractions through 3D audio. The work is focused toward the design and development of a system for gathering head orientation in real time and some interesting tests on the exploitation of spatial sounds were also presented

*"The Ghost Orchestra"* is an interesting project involving exploitation of binaural spatialization and visors for VR (i.e. Oculus) for cultural heritage as described in [130].

### J2.2.2 Spatial audio for other significant scenarios

The interest on audio spatialization by the research community has its origins on a paper written in the 60s by Schroeder, [131] introducing the idea of artificial reverberation based on digital signal processing, [132]. Indeed reproducing the behavior of acoustic waves spreading indoor requires ideally to spatialize each reverberation, i.e. to reproduce at the ears the sensation of a 3D space. Since then, the applied research and the market have gone toward different applications spanning from games, [133], to electroacustic music, [134] and soundscape design, [135] up to sonification, [136].

All these scenarios have in common the initial exploitation of the stereophonic approach, that provides a multi-channel reproduction system going from the traditional two channel stereo to the modern configurations with five, seven or more loudspeakers. This can be considered a channel based approach that freezes the position of the sound to the signals relations between loudspeakers, [137] and it is the one that is currently still used for cinema, home theater and pure audio content. The need of properly preserve spatial cues of an auditory scene has brought to the separation in coding of source signal and source location and to all the variety of spatialization techniques partially listed in the paragraph below, that have in turn significantly increased the potentials of video games, audio games, music expressions etc. Some remarkable results of applying spatial audio in these contexts are Mojang AB Minecraft for games or Karlheinz Stockhausen Cosmic Pulses for electroacustic music.

### J2.2.3 Binaural sound rendering solutions

The binaural sound reproduction through headphones requires full control of sound synthesis and binaural cues to be guaranteed, at the expense of the need by the users of wearing devices that may be considered intrusive, especially when noise cancellation and ear occlusion are required by the application. In order to achieve auralisation through headphones, HRTF (head related transfer function) filters are commonly used for left and right-ear because with headphones, the effect of the head and the pinna (with earplugs) is bypassed. They are then convolved with an anechoic sound signal for audio rendering. The most of the literature is focused toward the exploitation of non personalized HRTF that are recorded using a dummy head (e.g. KEMAR manikin, [138]) on a discrete spatial grid in both azimuth and elevation, [139]. Given the apparent impossibility of overcoming problems of unnatural coloration of the frequency spectrum and localization degradation, recent literature is instead focusing toward the exploitation of personalized HRTF, which still requires a long evaluation procedure (see e.g. [140]). It is worth noting that the availability of HRTFs coming from the last decade of research has brought to a standardization process known as as the Spatially Oriented Format for Acoustics (SOFA), [141], a personalized version is claimed to be available here [142]. In both cases, given the discrete points along which HRTFs are measured/computed interpolation techniques have to be employed, [143].

Most of the available tools for binaural audio reproduction are based on procedures to move from surrounding systems solutions to binaural sounds. Each of them need to use static, dynamic and environmental cues, [144]. Static cues are given by HRTF or HRIR involving all the issues related to physical characteristics of each individual;

dynamic cues are related to the motion of the listener and environmental cues are given by the room transfer function (RTF) or room impulse response (RIR).

For instance with the purpose of down mixing 5.1 to binaural, a solution exploiting virtual loudspeakers and HRIR, taking also into account the room response and head tracking data is proposed in [145].

The basic procedure for *Ambisonic* binaural rendering defines the virtual loudspeakers layout for which the corresponding output is computed as a linear combination of the B-format channels and finally HRTFs are introduced for each virtual loudspeaker and the obtained left channels are summed together, as it happens for right channels, and they respectively feed the left and right channels of the headphones, [146]. An optimized solution has been recently proposed in [147], where binaural decoding of Ambisonic soundfields is achieved basing on pre-computed, spherical harmonic-encoded binaural filters. Authors in [148] presented a *vector base amplitude panning (VBAP)* implementation for 3D head-tracked binaural rendering, where the binaural implementation of VBAP is achieved in the same way as virtual Ambisonics.

With the purpose of reducing computational requirements, various solutions have also been investigated simplifying the HRTFs and RIRs, see for instance [149, 150] and references therein, most of them at the expenses of perceived quality.

Finally, referring to [147] and [148], in the light of the main topic of this work, that authors propose opposite solutions about the device in charge of making spatial sound computations. The first reference indeed suggests to implement the VBAP rendering on an embedded Linux device that is placed locally. Advantages and drawbacks of the solution are discussed in terms of proper reproduction of the virtual source and response of the system to head-tracking data. It is interesting to observe that authors suggest the exploitation of second and third order Ambisonic to increase accuracy, stating that this would require computational resources not yet available on the CPU of an embedded system. On the other hand, authors in [148] investigate the opportunity of exploiting the distributor-side. This choice was mainly justified by economic reasons and drawbacks considered only in terms of absence of information about the listener environment. Problems arising in both cases could be overcome by exploiting the architecture proposed in this paper.

### J2.2.4 5G enabled AR/VR

According to the recent Molex State of 5G survey, [151], AR and VR applications top the list of primary use cases for 5G technology in consumer applications.

AR and VR experiences introduce many technical challenges related to the need of combining and synchronizing the real or virtual world with the user's motions. This requires high computational resources for rendering that can benefit from moving partially or totally the computational tasks to the edge. As a consequence, introducing 5G in this scenario would allow to satisfy users quality of experience (QoE) guaranteeing very low latency thus a realistic experience. The exploitation of the concept of private network is the solution proposed by Ericsson [152], basing on which VR services are delivered either through enterprise dedicated private networks or through a logical network slice created on the top of the existing physical public network.

Despite the large amount of informative articles available on the web about the advantages 5G could bring to AR/VR applications (e.g. [153], [154], [155]), scientific papers or available products exploiting these two paradigms together are quite limited.

An interesting discussion about advantages of 5G and MEC in the context of Mobile Augmented Reality (MAR) is discussed in [156] where the authors also discuss a possible application on tourism, stating the current status of infancy of these applications so far. In [157] a demonstration of the important role of 5G networking for VR game is shown by moving game servers without service interruption. An interesting project is described in [158] where public trials demonstrating the advantages of 5G for smart tourism are reported. Authors in [159] and [160] discuss in detail the potentiality of 5G and Beyond 5G (B5G) cellular networks for realizing mobile web augmented reality, presenting encouraging results.

## J2.3 Scenario

We assume as general scenario a museum where more users exploit AR devices for moving around and getting dedicated information as a function of what they are observing and their preferences that have been eventually previously expressed. The referred scenario for each user is sketched in Figure J2.1. The behavior of the service consists in a bidirectional communication between a client, that gets environmental information using different sensors (e.g., cameras, gyro, beacons, etc.), and a MEC application able to process the information in order to produce a proper spatialized stream. As a result the client application can reproduce a personalized sound built in real-time basing on users' position into the museum space.

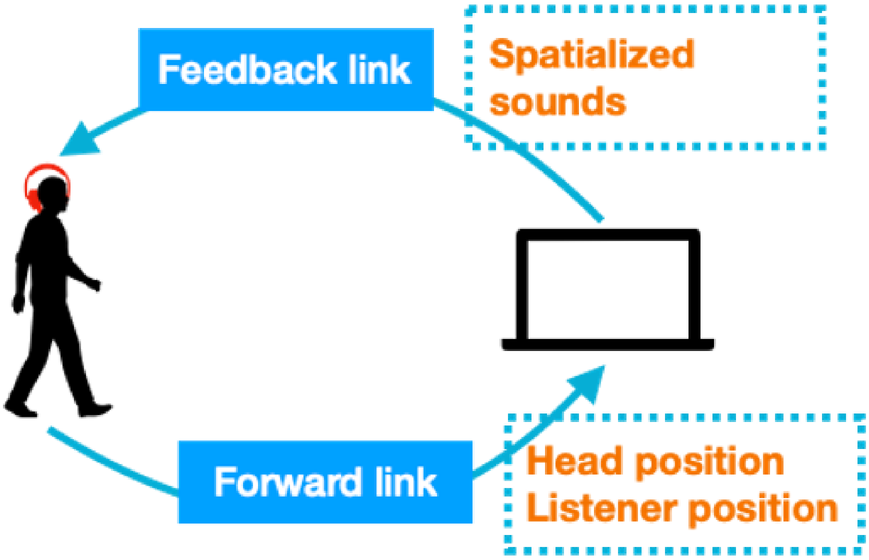


Figure J2.1: Spatial audio scenario

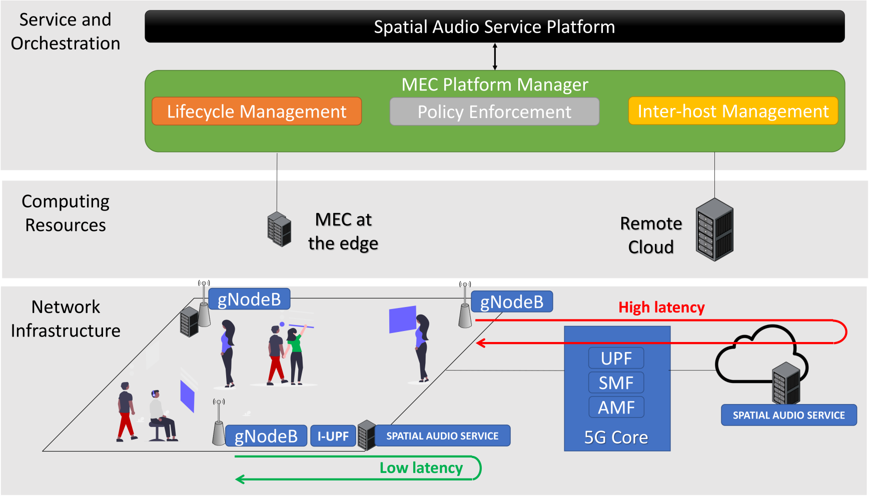


Figure J2.2: Detailed service architecture from network layers point of view

## J2.4 5G supporting spatial audio at edge

Audio applications dealing with binaural sounds require a very fast computation and playback of sounds to fulfill users' overall expectations. This can be reached both by computing audio at the user's device and by delegating computation to a remote node. The former option can offer the most reliable solution but it requires energy to perform computation. More energy means more weight to be carried by the end user (e.g., batteries) or reduced battery life. The latter option allows to release power constraints but introduces information transmission delay to move data through the network. Time needed to reproduce a valid sound sample is composed by the time to compute the sound plus two times the network delay to move information forward and backward from the end user device to the computational node:

$$T_{\text{sound}} = T_{\text{processing}} + 2T_{\text{network}}$$

where

$$T_{\text{network}} \approx T_{\text{forward}} \approx T_{\text{backward}}$$

With respect to previous mobile communications systems, 5G offers an unprecedented level of flexibility to fulfill service-specific requirements in terms of throughput, latency, and reliability. Such flexibility is achieved through novel techniques for the radio transmission and by novel architectural approaches. Two enabling technologies from the architectural viewpoint are: Software Defined Network (SDN) that allow to separate control and user plane information and to flexibly adapt the behavior of the network via software; Network Function Virtualization (NFV) that allows to implement and *orchestrate* traditional network functionalities over virtualized infrastructures.

Due to the virtualized nature of the 5G architecture, the core network is realized as a set of services among which the main ones are: (i) Access and Mobility Management Function (AMF) that is responsible for handling connection and mobility management tasks; (ii) Session Management Function (SMF) that is responsible for managing connections and session contexts; (iii) User Plane Function (UPF) that is the anchor element between the mobile and data networks.

Figure J2.2 shows the considered 5G architecture for the spatial audio service. Normally, the traffic generated by the users has to traverse the core network in order to leave the mobile network and reach the data network. However this may result in too high delay not compatible with the service under consideration (red line in Figure J2.2). In order to overcome this issue and fully enable MEC capability an instance of the UPF is deployed at the edge of the network. Since it is possible to separate control and user planes, the control plane is still redirect to the 5G Core to perform signaling and control while the intelligent UPF (I-UPF) at the edge is used as an anchor towards the data-network for the user plane. This way, the user plane traffic can leave immediately the mobile network after reaching the 5G base station (i.e., gNodeB) and flow towards the elaboration server at the edge, thus reducing latency (green line in Figure J2.2).

The computing infrastructure comprises the computing resources placed at the edge of the network (i.e. in proximity of the gNodeBs) allowing MEC, and computing resources available in a remote cloud. On top of the network and computing infrastructure is a Service and Orchestration layer which is responsible for the management of the deployment and life-cycle of the spatial audio service.

The Service and Orchestration Layer is composed by the MEC Platform Manager and the DASH Service Platform. The MEC Platform Manager is able to provide the upper layer with a uniform northbound interface API decoupling the Service Layer and the Computing Resources Layer. Its functions include:

- ***Life-cycle Management*:** this process takes care of the entire MEC application life-cycle, treating each MEC application instance as a Virtual Network Function (VNF) instance and performing health checks and auto healing for high availability

- ***Policy Enforcement*:** it manages networks and applications connectivity, according to the network policies

- ***Inter-host Management*:** this block is responsible of enabling networking and communications between different MEC hosts, exposing them to the *Service Layer*.

## J2.5 Referred spatialization tools

Bringing rich, dynamic audio environments into AR/VR experiences without affecting performance can be challenging. There are often few CPU resources allocated for audio, especially on mobile, which can limit the number of simultaneous high-fidelity 3D sound sources for complex environments.

Various tools implementing binaural spatialization were investigated and their advantages and disadvantages discussed before making a choice, i.e. Resonance Audio (Google), 3D Tune-In Toolkit, [161], Audio Spatializer [Oculus (Facebook)], Steam Audio (Valve corporation), SOFAlizer [162], Spatial Audio Framework [163].

The choice fell on Resonance Audio, [164]. Resonance Audio is a multi-platform spatial audio SDK, delivering high fidelity at scale. The Resonance Audio SDK uses highly optimized digital signal processing algorithms based on higher order Ambisonics to spatialize hundreds of simultaneous 3D sound sources, without compromising audio quality. The SDKs run on Android, iOS, Windows, MacOS and Linux platforms and provide integration for Unity, Unreal Engine, FMOD, Wwise and DAWs.

Native APIs for C/C++, Java and Objective-C are also provided together with the full source code C++ library.

As part of the open source project, a reference implementation of YouTube's Ambisonic-based spatial audio decoder is provided. Using this implementation, developers can easily render Ambisonic content in their VR media and other applications, while benefiting from Ambisonics open source, royalty-free model. The project also includes encoding, sound field manipulation and decoding techniques, as well as head related transfer functions (HRTFs) used to achieve rich spatial audio that scales across a wide spectrum of device types and platforms. Lastly the entire library of highly optimized DSP classes and functions is available: this includes resamplers, convolvers, filters, delay lines and other DSP capabilities.

## J2.6 Spatialization server application

In order to demonstrate the proposed service a custom application has been developed. The application, written in C++, uses a custom Linux build of the Resonance Audio library and some utility libraries like the `Boost C++ Libraries` in order to handle the TCP socket communication between client and server in the easiest way, [165]. Figure J2.3 summarize the simple behavior of the developed service: a client streaming application (e.g., an AR mobile application) is able to connect to the *spatialization server* that is awaiting for requests from connected clients of spatialization processing.

The sample application starts with the generation of a stereo *sin* tone at $500Hz$ and stores the generated samples into a proper buffer. The *duration* and the *sample_rate* of the generated samples are expected as parameters. Algorithm 3 shows the samples generation process.

**Algorithm 3** Algorithm for samples generation

```
Input: duration, sample_rate
Output: buffer
    Initialisation:
    buffer: ARRAY [duration*sample_rate] OF Byte
    LOOP Process
    for $i = 0$ to $duration * sample\_rate$ do
        buffer[$i$] ← $sin(2\pi * frequency/sample\_rate * i)$
    end for
    return buffer
```

Once a socket endpoint has been set-up the main loop of the spatialization server application acts as summarized by Algorithm 4.

**Algorithm 4** Spatialization server application main loop algorithm

```
Input: TCP_port
    Initialisation:
samples = Algorithm1
socket = init(TCP_port)
    LOOP Process
while true do
    receive audio source position
    while samples is not empty do
        buffer = compute_spatial_stream(samples,x,y,z)
        send buffer through socket
    end while
end while
```

The main loop of the application is waiting for audio source position parameters sent through socket by the connected client and computes the spatialized streams. When a set of spatialized samples is ready it is transmitted to the client. The flow diagram shown in Figure J2.3 summarizes the spatialization server application behavior.

## J2.7 Performance analysis & results

In order to offer a performance analysis of the advantages introduced by MEC, in contrast to what happens with a cloud deployment, as shown in Figure J2.4, in the proposed spatial audio use case we adopt the model proposed in [166].

The main key performance indicator for the overall system performance is represented by the maximum achievable throughput to serve the spatial users which can be expressed as follow:

$$R_{max} \leq \min \left\{ \frac{c \times MSS}{RTT \times \sqrt{PL}} \times N, BR \right\} \tag{J2.1}$$

where $c$ is a constant depending on the specific TCP implementation, ack strategy, loss mechanism and congestion avoidance algorithm, [167], whose typical values are comprised between 0.9 and 1.2; $MSS$ is the maximum TCP segment size; $RTT$ is the round trip time between the users and the spatial audio elaboration node; $PL$ is the packet loss ratio; $N$ is the number of users supported with maximum throughput, and $BR$ is the available data-rate, i.e. the network capacity, which in the following we assume to be $1 Gbps$ per radio access point. As it can be noticed from Eq. J2.1,

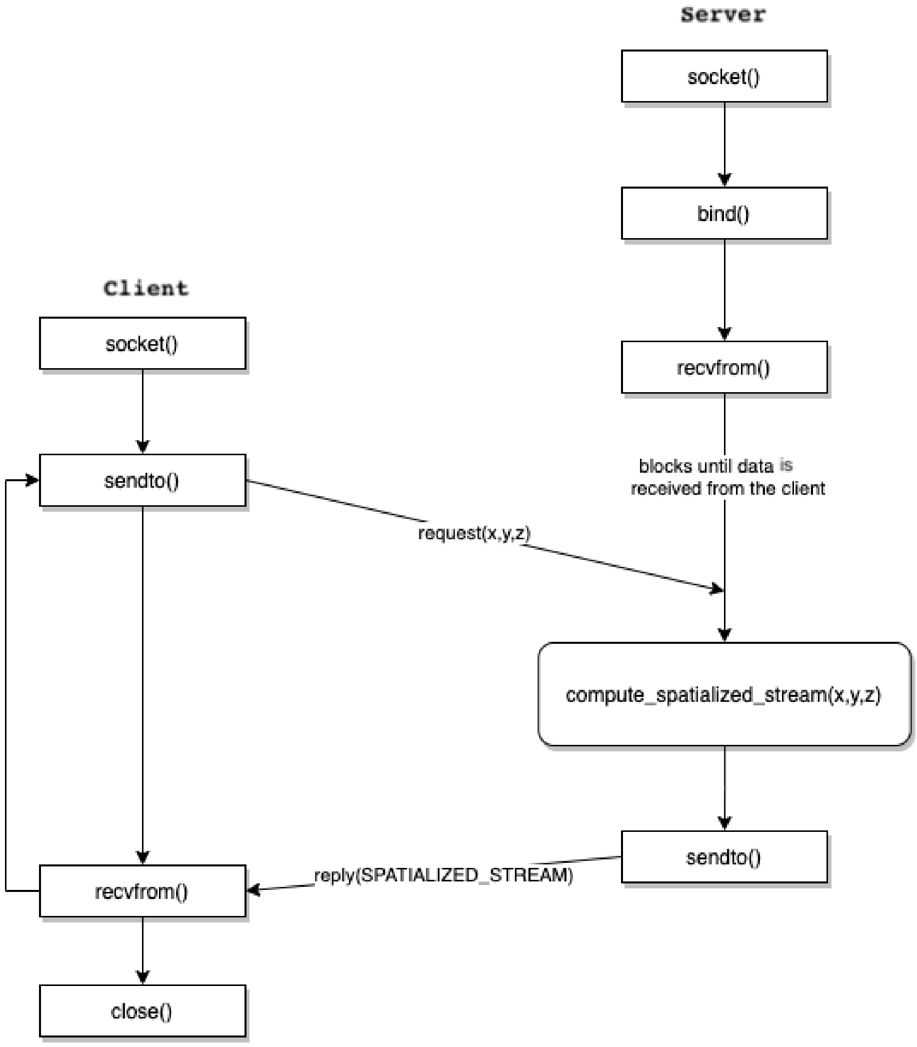

Server
socket()
bind()
recvfrom()
blocks until data is received from the client
Client
socket()
sendto()
request(x,y,z)
compute_spatialized_stream(x,y,z)
sendto()
reply(SPATIALIZED_STREAM)
recvfrom()
close()

Figure J2.3: Flow diagram of the spatialization server application

the *min* operator models a possible bottleneck effect of the network. However, in the considered scenario with private dedicated indoor 5G coverage and high data-rate, this in not likely to happen.

The term $RTT \times \sqrt{PL}$ shows the inverse relation between the experienced throughput and round trip time and packet loss. On this hand, the advantages deriving from MEC are twofold in fact:

- thanks to the possibility to perform the elaboration closer to the users, $RTT$ is reduced
- packets exchanged are not sent over the internet service provider network where packet drops due to congestion can occur generating packet loss, thus MEC has also the advantage to reduce $PL$

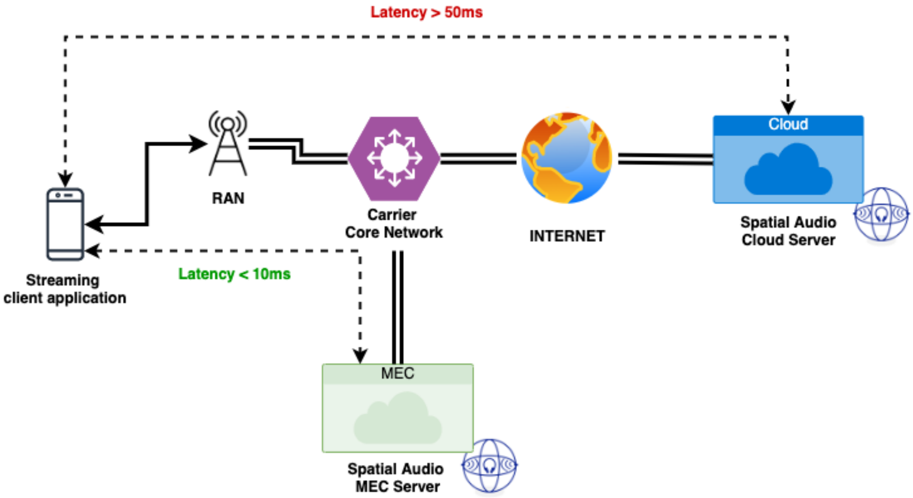


Figure J2.4: MEC vs Cloud service deployment

Given the above, and assuming that each user generates one TCP flow, all the flows have the same size, and all the flows are terminated at the same location, the number of spatial audio users supported without any performance impairment can be expressed as:

$$N_U = \frac{R_{max}}{R_U} \tag{J2.2}$$

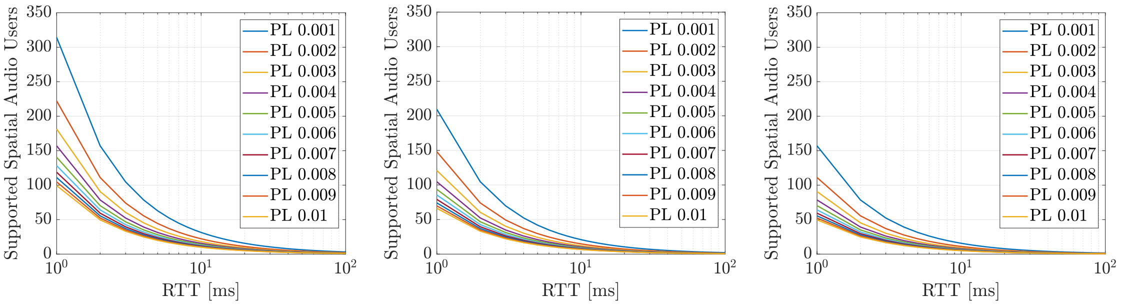
where $R_{max}$ is given by Equation J2.1 and $R_U$ is the throughput per single user. $R_U$ depends on the selected audio quality and can be calculated as:

$$R_U = W \times S \times 2 \tag{J2.3}$$

where $W$ is the sample bit-width whose typical values are 16, 24 and $32bits$; $S$ is the sample rate that we assume equal to $44100Hz$.

Figure J2.5 shows the number of supported spatial audio users per different audio qualities ($16bit$, $24bit$, and $32bit$, respectively) and different network conditions represented by a packet loss ratio varying from 0.01 to 0.001. As expectable, the number of supported users decreases by increasing the quality of the audio, due to the higher required throughput per user. Furthermore, it has an inverse relation with the PL. It is worth mentioning that variations of PL may be related to both the wireless channel conditions (fading, path-loss, presence of obstacles, crowdedness of users in the area) and to congestion into the transport network. In this context, MEC has the two-fold advantage of reducing the RTT and PL related to congestion in the transport network.

The results presented in Figure J2.5 may represent a framework to adopt some adaptive strategies for dynamic placement of audio contents or audio quality adjustment based on packet loss experienced by the users and targeted number of users (as it happens, for example, in Dynamic Adaptive video Streaming over HTTP (DASH), [168]). However, this aspect is out of the scope of this work and represents a future direction.



(a) 16 bit (b) 24 bit (c) 32 bit

Figure J2.5: Comparison of supported users for different audio sample widths

In order to quantify the advantage in terms of supported users for the MEC scenario with respect to the cloud case, we introduce the supported users gain metric $\mathit{\Gamma}$ which is given by

$$\mathit{\Gamma} = \frac{N_{U,MEC}}{N_{U,CLOUD}} \tag{J2.4}$$

where $N_{U,MEC}$ and $N_{U,CLOUD}$ are the number of users supported in the MEC and cloud case, respectively, and are calculated according to Eq. J2.2 and Eq. J2.1 by assuming specific values for $RTT$ and $PL$ for the two scenarios. Thus, Eq. J2.4 can be expressed as follows:

$$\mathit{\Gamma} = \sqrt{\frac{PL_{CLOUD}}{PL_{MEC}}} \times \frac{RTT_{CLOUD}}{RTT_{MEC}} = \sqrt{\alpha} \times \beta \tag{J2.5}$$

where $PL_{MEC}$ and $PL_{CLOUD}$ are $PL$ values for MEC and cloud respectively; $RTT_{MEC}$ and $RTT_{CLOUD}$ are $RTT$ values for MEC and cloud respectively. $\alpha$ and $\beta$ represent the ratio between cloud and MEC PL and RTT, respectively.

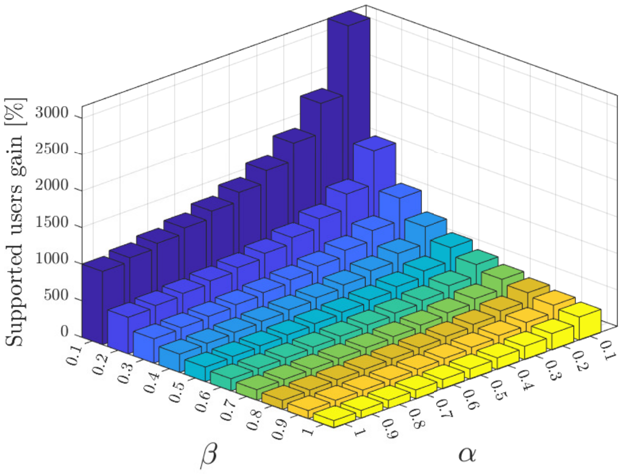


Figure J2.6: Gain in terms of supported spatial audio users

Figure J2.6 shows the performance gain for different values of the parameters $\alpha$ and $\beta$. Results show that gain of such a system in terms of supported spatial audio users grow more rapidly with the reduction of round trip time compared to a reduction of packet loss. In other words, to move the function responsible for the audio spatialization at the edge of the network, is more convenient when the edge elaboration reduces significantly the experienced latency. When the elaboration at the edge introduces a reduction of ten times for both the latency and the packet loss, a MEC approach is able to support a number of users up to 30 times larger with respect to a cloud one.

As $\alpha$ and $\beta$ can be derived by observing the network behavior and user performance, the results shown in Figure J2.6 can be utilized by the MEC platform manager to perform the deployment of the spatial audio service at the edge based on achievable gain and targeted performance.

## J2.8 Conclusion

This paper presented a possible sound spatialization application for cultural heritage enjoyment exploiting many advantages of the 5G architecture. The proposed solution allows to guarantee almost real-time sound spatialization as a function of users' movements for the most generic client device. This can be achieved by exploiting the MEC paradigm, allowing to move the computational load from the client to the edge without involving the core network. Results also demonstrate that this scenario is able to serve an increased number of users with reference to a cloud based one, thus making this solution even more appealing for the CH Italian scenario, which comprises many big museums with a high daily number of visitors. From a more general point of view, the presented application is another demonstration of the fundamental importance of the MEC paradigm as an enabler of new services and business models.

# Part III

# Conclusion

# Concluding remarks and future directions

To orchestrate real and virtual resources with the aim of optimize performances, a huge number of technical and design issues need to be addressed. This dissertation focuses on design and technical challenges that need to be faced to deploy a fully functional system that is able to perform network optimization through orchestration and virtualization.

The first topic addressed in this thesis is the optimization of a Dynamic Adaptive Streaming over HTTP (DASH) video streaming service that needs to be forwarded through a real world Passive Optical Network (PON). Real world network parameters have been measured and a reproducible testing framework has been deployed. The builded framework has been used to validate the increment of performances given by a Multi-access Edge Computing (MEC) server deployed at different locations. The same principles are moved in the context of spatial audio for cultural heritages, where computational power is needed at the edge in addition to the low latency. The presence of a MEC server at the edge shows that MEC architecture is an enabler in that context for both offloading of computational power from the user device and network computation at the edge.

The second topic addressed is the design and deployment of a real network involving Open Radio Access Network (O-RAN), PON, and 5G Service-Based Architecture (SBA) core. A management and orchestration framework is designed to take decision on commissioning or decommissioning network slices. It has been shown that this approach is able to react to network events in only 1 second, being feasible with all kind of applications that are not sensible to this start-up time. Slices provided by the presented orchestration components are able to reduce latency up to 85% and minimize the perceived jitter. The time of 1 second to apply changes into the network is divided in 0.1s (Software Defined Networking (SDN) controller and orchestrator delays) and 0.9s (hardware-specific time to apply changes).

The third topic addressed the possibility of managing the Kubernetes (K8s) scheduling process, making the scheduler aware of the end user's experienced latency to take decisions about service migration. The so developed latency aware scheduler is able to migrate the service seamlessly in a way to optimize experienced application latency.

The last topic regards cybersecurity aspects in O-RAN. Cybersecurity in general is a very hot research topic and implementing security by design is a very hard objective to reach. Studying the state of the art of the O-RAN alliance specifications showed clearly that there is space for malicious attacks to the non Real Time Radio Intelligent controller (RIC) interface. A man-in-the middle attack has been demonstrated and a blockchain-based approach has been proposed to protect policy tamping attacks to the A1 interface.

The edge ecosystem is still being developed and designed and there are many research questions that remain still open. 6G is setting up new standards and objectives both ambitious and challenging. The trend of the edge computing is something 6G standard gives many expectations. The virtualization standardization and the cooperation between different virtualization infrastructure managers will probably raise challenges in the future. The network layer will have a huge impact on how information will be exchanged by Virtual Infrastructure Manager (VIM)s and the heterogeneity of access technologies will call to be exploited. A joint orchestration of PON, Radio Access Network (RAN) and services will be required to seamlessly adapt network and services and to enable service migration from the cloud down to our personal devices decreasing the network latency or up to the central cloud to offload computation and reduce costs.

# References

[1] O-RAN Alliance Working Group 1, "O-RAN.WG1.O-RAN-Architecture-Description-v06.00 - O-RAN Architecture Description v06.00." online, https://orandownloadsweb.azurewebsites.net/download?id=206. November 2021.

[2] O-RAN Alliance Working Group 2, "O-RAN.WG2.R1GAP-v03.00 - A1 interface: General Aspects and Principles v03.00." online, https://orandownloadsweb.azurewebsites.net/download?id=318. July 2022.

[3] K. Miller, "Calculating optical fiber latency." https://www.m2optics.com/blog/bid/70587/calculating-optical-fiber-latency, 2012. [Online; accessed 10-March-2023].

[4] U. Cisco, "Cisco annual internet report (2018–2023) white paper," *Cisco: San Jose, CA, USA*, vol. 10, no. 1, pp. 1–35, 2020.

[5] GSMA, "The state of mobile internet connectivity report 2022." https://www.gsma.com/r/wp-content/uploads/2022/12/The-State-of-Mobile-Internet-Connectivity-Report-2022.pdf?utm_source=website&utm_medium=download-button&utm_campaign=somic22, 2023. [Online; accessed 15-March-2023].

[6] M. P. et Al., "Mobile-edge computing – introductory technical white paper," tech. rep., ETSI, 09 2014.

[7] B. Liang, M. A. Gregory, and S. Li, "Multi-access edge computing fundamentals, services, enablers and challenges: A complete survey," *Journal of Network and Computer Applications*, vol. 199, p. 103308, 2022.

[8] X. Jiang, F. R. Yu, T. Song, and V. C. Leung, "A survey on multi-access edge computing applied to video streaming: Some research issues and challenges," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 2, pp. 871–903, 2021.

[9] F. Richter, “Amazon, microsoft and google dominate cloud market.” `https://www.statista.com/chart/18819/worldwide-market-share-of-leading-cloud-infrastructure-service-providers/`, 2022. [Online; accessed 19-February-2023].

[10] Ookla, “Latency is the next frontier of consumer experience. are you ready?.” `https://www.ookla.com/articles/latency-the-next-frontier-of-consumer-experience?utm_source=Ookla%20Insights&utm_medium=email&utm_campaign=Express_2023-02-08_08:30:00&utm_content=Latency%20is%20the%20Next%20Frontier%20of%20Consumer%20Experience.%20Are%20You%20Ready?`, 2023. [Online; accessed 15-March-2023].

[11] E. Commission, “Iot and the future of edge computing in europe,” 2022.

[12] T. L. Foundation, “State of the edge 2022,” 2022.

[13] Z. Li, M. Kihl, Q. Lu, and J. A. Andersson, “Performance overhead comparison between hypervisor and container based virtualization,” in *2017 IEEE 31st International Conference on advanced information networking and applications (AINA)*, pp. 955–962, IEEE, 2017.

[14] A. Celesti, D. Mulfari, M. Fazio, M. Villari, and A. Puliafito, “Exploring container virtualization in iot clouds,” in *2016 IEEE international conference on Smart Computing (SMARTCOMP)*, pp. 1–6, IEEE, 2016.

[15] M. G. Xavier, M. V. Neves, F. D. Rossi, T. C. Ferreto, T. Lange, and C. A. De Rose, “Performance evaluation of container-based virtualization for high performance computing environments,” in *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pp. 233–240, IEEE, 2013.

[16] M. J. Scheepers, “Virtualization and containerization of application infrastructure: A comparison,” in *21st twente student conference on IT*, vol. 21, 2014.

[17] ETSI, “Multi-access edge computing,” 2023.

[18] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[19] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, “A survey on software-defined networking,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 27–51, 2014.

[20] P. Parol and M. Pawlowski, “Towards networks of the future: Sdn paradigm introduction to pon networking for business applications,” in *2013 Federated Conference on Computer Science and Information Systems*, pp. 829–836, 2013.

[21] S. McGettrick, F. Slyne, N. Kitsuwan, D. B. Payne, and M. Ruffini, “Experimental end-to-end demonstration of shared n:m dual-homed protection in sdn-controlled long-reach pon and pan-european core,” *J. Lightwave Technol.*, vol. 34, pp. 4205–4213, Sep 2016.

[22] P. Alvarez, F. Slyne, C. Bluemm, J. M. Marquez-Barja, L. A. DaSilva, and M. Ruffini, “Experimental demonstration of sdn-controlled variable-rate fronthaul for converged lte-over-pon,” in *Optical Fiber Communication Conference*, p. Th2A.49, Optica Publishing Group, 2018.

[23] R. Bassoli, “Network function virtualization,” in *Computing in Communication Networks*, pp. 119–132, Elsevier, 2020.

[24] Q.-V. Pham *et al.*, “A survey of multi-access edge computing in 5g and beyond: Fundamentals, technology integration, and state-of-the-art,” *IEEE Access*, vol. 8, pp. 116974–117017, 2020.

[25] M. ETSI, “Multi-access edge computing (mec) framework and reference architecture,” *ETSI GS MEC*, vol. 3, p. V2, 2019.

[26] J. Yao, T. Han, and N. Ansari, “On mobile edge caching,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2525–2553, 2019.

[27] G. Forecast, “Cisco visual networking index: global mobile data traffic forecast update, 2017–2022,” *Update*, vol. 2017, p. 2022, 2019.

[28] D. I. Forum, “Abr logic,” 2021.

[29] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, “Bola: Near-optimal bitrate adaptation for online videos,” *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1698–1711, 2020.

[30] K. Spiteri, R. Sitaraman, and D. Sparacio, "From theory to practice: Improving bitrate adaptation in the dash reference player," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 15, no. 2s, pp. 1–29, 2019.

[31] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys Tutorials*, vol. 16, pp. 1617–1634, Third 2014.

[32] G. Ishigaki, S. Devic, R. Gour, and J. P. Jue, "Dynamic bandwidth allocation for pon slicing with performance-guaranteed online convex optimization," in *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2021.

[33] C. Centofanti, A. Marotta, C. Rinaldi, F. Franchi, D. Cassioli, and F. Graziosi, "Improved dash video streaming performance by mec-enabled optical access," in *Asia Communications and Photonics Conference 2021*, p. T4A.144, Optica Publishing Group, 2021.

[34] M. Bjorklund, "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)." RFC 6020, Oct. 2010.

[35] R. Enns, M. Bjorklund, A. Bierman, and J. Schonwalder, "Network Configuration Protocol (NETCONF)." RFC 6241, June 2011.

[36] S. Das and M. Ruffini, "Optimal virtual pon slicing to support ultra-low latency mesh traffic pattern in mec-based cloud-ran," in *2021 International Conference on Optical Network Design and Modeling (ONDM)*, pp. 1–5, June 2021.

[37] S. Das, F. Slyne, and M. Ruffini, "Optimal slicing of virtualised passive optical networks to support dense deployment of cloud-ran and multi-access edge computing," *IEEE Networks*, 2022.

[38] M. P. McGarry and M. Reisslein, "Investigation of the dba algorithm design space for epons," *Journal of Lightwave Technology*, vol. 30, no. 14, pp. 2271–2280, 2012.

[39] M. Zhu, J. Gu, and G. Li, "Pwc-pon: an energy-efficient low-latency dba scheme for time division multiplexed passive optical networks," *IEEE Access*, vol. 8, pp. 206848–206865, 2020.

[40] H. Uzawa, K. Honda, H. Nakamura, Y. Hirano, K.-i. Nakura, S. Kozaki, and J. Terada, “Dynamic bandwidth allocation scheme for network-slicing-based tdm-pon toward the beyond-5g era,” *Journal of Optical Communications and Networking*, vol. 12, no. 2, pp. A135–A143, 2020.

[41] S. Ibrahim, B. He, and H. Jin, “Towards pay-as-you-consume cloud computing,” in *2011 IEEE International Conference on Services Computing*, pp. 370–377, July 2011.

[42] Nokia Network Sharing White Paper, “available online at https://www.nokia.com/networks/solutions/network-sharing/, accessed on Aug. 2022..”

[43] M. Kassis, S. Costanzo, and M. Yassin, “Flexible Multi-Operator RAN Sharing: Experimentation and Validation Using Open Source 4G/5G Prototype,” in *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, pp. 205–210, IEEE, 2021.

[44] C.-C. Tsai, F. J. Lin, and H. Tanaka, “Evaluation of 5G Core Slicing on User Plane Function,” *Communications and Network*, vol. 13, no. 3, pp. 79–92, 2021.

[45] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, “Network slicing and softwarization: A survey on principles, enabling technologies, and solutions,” *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[46] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca, and J. Folgueira, “Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges,” *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80–87, 2017.

[47] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, “Network Slicing in 5G: Survey and Challenges,” *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[48] D. Kim and S. Kim, “Network Slicing as Enablers for 5G Services: State of the Art and Challenges for Mobile Industry,” *Telecommun. Syst.*, vol. 71, p. 517–527, jul 2019.

[49] N. Afraz, F. Slyne, H. Gill, and M. Ruffini, "Evolution of access network sharing and its role in 5g networks," *Applied Sciences*, vol. 9, no. 21, p. 4566, 2019.

[50] S. Das and M. Ruffini, "Optimal virtual pon slicing to support ultra-low latency mesh traffic pattern in mec-based cloud-ran," in *2021 International Conference on Optical Network Design and Modeling (ONDM)*, pp. 1–5, IEEE, 2021.

[51] C. Centofanti, A. Marotta, D. Cassioli, F. Graziosi, N. Sambo, L. Valcarenghi, C. Bernard, and H. Roberts, "Slice Management in SDN PON Supporting Low-Latency Services," in *2022 European Conference on Optical Communications (ECOC)*, IEEE, 2022.

[52] X. Foukas, M. K. Marina, and K. Kontovasilis, "Orion: Ran slicing for a flexible and cost-effective multi-service mobile network architecture," in *Proceedings of the 23rd annual international conference on mobile computing and networking*, pp. 127–140, 2017.

[53] A. Marotta, K. Kondepu, D. Cassioli, C. Antonelli, L. M. Correia, and L. Valcarenghi, "Software defined 5G converged access as a viable techno-economic solution," in *2018 Optical Fiber Communications Conference and Exposition (OFC)*, pp. 1–3, IEEE, 2018.

[54] OAI, "OpenAirInterface: A Flexible Platform for 5G Research," Sep. 2022.

[55] OpenAirInterface 5G Core, Sep. 2022.

[56] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, interfaces, algorithms, security, and research challenges," *arXiv preprint arXiv:2202.01032*, 2022.

[57] S. Mondal and M. Ruffini, "Optical Front/Mid-haul with Open Access-Edge Server Deployment Framework for Sliced O-RAN," *IEEE Transactions on Network and Service Management*, 2022.

[58] S. Das and M. Ruffini, "Optimal virtual PON slicing to support ultra-low latency mesh traffic pattern in MEC-based Cloud-RAN," in *2021 International Conference on Optical Network Design and Modeling (ONDM)*, IEEE, 2021.

[59] R. Schmidt, M. Irazabal, and N. Nikaein, "FlexRIC: an SDK for next-generation SD-RANs," in *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*, pp. 411–425, 2021.

[60] Kubernetes. `http://kubernets.io/`. [Online; accessed 28-Jan-2023].

[61] O. Tomarchio, D. Calcaterra, and G. D. Modica, "Cloud resource orchestration in the multi-cloud landscape: a systematic review of existing frameworks," *Journal of Cloud Computing*, vol. 9, pp. 1–24, 2020.

[62] J. Santos, T. Wauters, B. Volckaert, and F. De Turck, "Towards network-aware resource provisioning in kubernetes for fog computing applications," in *2019 IEEE Conference on Network Softwarization (NetSoft)*, pp. 351–359, IEEE, 2019.

[63] L. Toka, "Ultra-reliable and low-latency computing in the edge with kubernetes," *Journal of Grid Computing*, vol. 19, no. 3, p. 31, 2021.

[64] Ł. Wojciechowski, K. Opasiak, J. Latusek, M. Wereski, V. Morales, T. Kim, and M. Hong, "Netmarks: Network metrics-aware kubernetes scheduler powered by service mesh," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pp. 1–9, IEEE, 2021.

[65] A. C. Caminero and R. Muñoz-Mansilla, "Quality of service provision in fog computing: Network-aware scheduling of containers," *Sensors*, vol. 21, no. 12, p. 3978, 2021.

[66] F. Tonini, C. Natalino, D. A. Temesgene, Z. Ghebretensaé, L. Wosinska, and P. Monti, "Benefits of pod dimensioning with best-effort resources in bare metal cloud native deployments," *IEEE Networking Letters*, 2023.

[67] "MQTT version 5.0 Specification." `https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html`, 2019. [Online; accessed 28-Jan-2023].

[68] C. Centofanti, A. Marotta, C. Rinaldi, F. Franchi, D. Cassioli, and F. Graziosi, "Improved dash video streaming performance by mec-enabled optical access," in *2021 Asia Communications and Photonics Conference (ACP)*, pp. 1–3, IEEE, 2021.

[69] C. Centofanti and W. Tiberti, "Kubernetes latency-aware scheduler." `https://github.com/Kubernetes-scheduler/`, 2023.

[70] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges.," *arXiv preprint arXiv:2202.01032*, 2022.

[71] C. T. Shen and *et al.*, "Security threat analysis and treatment strategy for oran," in *2022 24th IEEE International Conference on Advanced Communication Technology (ICACT)*, 2022.

[72] O-RAN Alliance, "O-ran sc projects." online, `https://wiki.o-ran-sc.org/pages/viewpage.action?pageId=41452927`.

[73] O-RAN Alliance Working Group 2, "O-RAN.WG2.Non-RT-RIC-ARCH-TR-v01.01 - Non-RT RIC: Functional Architecture v01.01." online, `https://orandownloadsweb.azurewebsites.net/download?id=74`. March 2021.

[74] O-RAN Alliance Working Group 3, "O-RAN.WG3.E2GAP-v02.01 - Near-RT RIC: Architecture & E2 General Aspects and Principles v02.01." online, `https://orandownloadsweb.azurewebsites.net/download?id=262`. February 2022.

[75] O-RAN Alliance Working Group 2, "O-RAN.WG2.AIML-v01.03 - O-RAN AI/ML Workflow Description and Requirements v01.03." online, `https://orandownloadsweb.azurewebsites.net/download?id=158`. July 2021.

[76] L. Velasco, M. Signorelli, O. G. De Dios, C. Papagianni, R. Bifulco, J. J. V. Olmos, S. Pryor, G. Carrozzo, J. Schulz-Zander, M. Bennis, R. Martinez, F. Cugini, C. Salvadori, V. Lefebvre, L. Valcarenghi, and M. Ruiz, "End-to-end intent-based networking," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 106–112, 2021.

[77] P. Madadi, C. Lo, J. Jeon, J. Cho, J.-H. Song, and J. C. Zhang, "Open-ran and future intelligent networks," in *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, pp. 1–7, 2022.

[78] A. Arnaz, J. Lipman, M. Abolhasan, and M. Hiltunen, "Toward integrating intelligence and programmability in open radio access networks: A comprehensive survey," *IEEE Access*, vol. 10, pp. 67747–67770, 2022.

[79] B. Brik, K. Boutiba, and A. Ksentini, "Deep learning for b5g open radio access network: Evolution, survey, case studies, and challenges," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 228–250, 2022.

[80] A. Kak, V.-Q. Pham, H.-T. Thieu, and N. Choi, "Poster: Multi-rat network slicing in the open ran era," in *2022 IEEE 30th International Conference on Network Protocols (ICNP)*, pp. 1–2, 2022.

[81] E. Coronado, S. Siddiqui, and R. Riggio, "Roadrunner: O-ran-based cell selection in beyond 5g networks," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–7, 2022.

[82] D. Mimran, R. Bitton, Y. Kfir, E. Klevansky, O. Brodt, H. Lehmann, Y. Elovici, and A. Shabtai, "Security of open radio access networks," *Computers & Security*, vol. 122, p. 102890, 2022.

[83] M. Liyanage, A. Braeken, S. Shahabuddin, and P. Ranaweera, "Open ran security: Challenges and opportunities," 2022.

[84] S. Soltani, M. Shojafar, R. Taheri, and R. Tafazolli, "Can open and ai-enabled 6g ran be secured?," *IEEE Consumer Electronics Magazine*, vol. 11, no. 6, pp. 11–12, 2022.

[85] J. Śliwa and M. Suchański, "Security threats and countermeasures in military 5g systems," in *2022 24th International Microwave and Radar Conference (MIKON)*, pp. 1–6, 2022.

[86] A. Braeken, M. Liyanage, P. Kumar, and J. Murphy, "Novel 5g authentication protocol to improve the resistance against active attacks and malicious serving networks," *IEEE Access*, vol. 7, pp. 64040–64052, 2019.

[87] W. Tiberti, E. Di Fina, A. Marotta, and D. Cassioli, "Impact of man-in-the-middle attacks to the o-ran inter-controllers interface," in *2022 5th IEEE Future Networks World Forum (FNWF'22)*, 2022.

[88] Y. Yuan, J. Yang, R. Duan, I. Chih-Lin, and J. Huang, "Anomaly detection and root cause analysis enabled by artificial intelligence," in *2020 IEEE Globecom Workshops (GC Wkshps*, pp. 1–6, 2020.

[89] D. Dik and M. S. Berger, "Transport security considerations for the open-ran fronthaul," in *2021 IEEE 4th 5G World Forum (5GWF)*, pp. 253–258, 2021.

[90] L. Giupponi and F. Wilhelmi, "Blockchain-enabled network sharing for o-ran in 5g and beyond," *IEEE Network*, vol. 36, no. 4, pp. 218–225, 2022.

[91] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," 2020-08-10 04:08:00 2020.

[92] D. Van Landuyt and W. Joosen, "A descriptive study of assumptions in stride security threat modeling," 2021-11-17.

[93] O-RAN Alliance Security Focus Group, "O-RAN.SFG.Threat-Model-v03.00 - O-RAN Security Threat Modeling and Remediation Analysis v03.00." online, https://orandownloadsweb.azurewebsites.net/download?id=260. April 2022.

[94] "Information security, cybersecurity and privacy protection - Guidance on managing information security risks," standard, International Organization for Standardization, Geneva, CH, Oct. 2022.

[95] O-RAN Alliance Security Focus Group, "O-RAN.SFG.Security-Protocols-Specifications-v03.00 - Security Protocols Specifications v03.00." online, https://orandownloadsweb.azurewebsites.net/download?id=243. November 2021.

[96] X. Wang, Y. L. Yin, and H. Yu, "Finding collisions in the full sha-1," in *Advances in Cryptology – CRYPTO 2005* (V. Shoup, ed.), (Berlin, Heidelberg), pp. 17–36, Springer Berlin Heidelberg, 2005.

[97] "Safecurves: choosing safe curves for elliptic-curve cryptography." online, https://safecurves.cr.yp.to/.

[98] O-RAN Alliance Security Focus Group, "O-RAN.SFG.Non-RT-RIC-Security-TR-v01.00 - Study on Security for Non-RT-RIC v01.00." online, https://orandownloadsweb.azurewebsites.net/download?id=256. March 2022.

[99] O-RAN Alliance Security Focus Group, "O-RAN.SFG.Security-Requirements-Specifications-v03.00 - Security Requirements Specifications v03.00." online, https://orandownloadsweb.azurewebsites.net/download?id=259. March 2022.

[100] O-RAN Alliance Working Group 2, "O-RAN.WG2.A1TP-v02.00 - A1 interface: Transport Protocol v02.00." online, https://orandownloadsweb.azurewebsites.net/download?id=315. July 2022.

[101] "Ettercap - man-in-the-middle attack software." online, https://github.com/Ettercap/ettercap.

[102] O-RAN Alliance Working Group 2, "O-RAN.WG2.A1AP-v03.02 - A1 interface: Application Protocol v03.02." online, https://orandownloadsweb.azurewebsites.net/download?id=276. April 2022.

[103] O-RAN Alliance Working Group 2, "O-RAN.WG2.A1TD-v03.00 - A1 interface: Type Definitions v03.00." online, https://orandownloadsweb.azurewebsites.net/download?id=277. April 2022.

[104] NGMN Alliance, "5g white paper," *Next generation mobile networks, white paper*, vol. 1, 2015.

[105] 5G Infrastructure Association, "5g pan-european trials roadmap strategy," 2017.

[106] A. Palombini, "Storytelling and telling history. towards a grammar of narratives for cultural heritage dissemination in the digital era," *Journal of cultural heritage*, vol. 24, pp. 134–139, 2017.

[107] E. Madirov and S. Absalyamova, "The influence of information technologies on the availability of cultural heritage," *Procedia-Social and Behavioral Sciences*, vol. 188, pp. 255–258, 2015.

[108] A. C. Addison, "Emerging trends in virtual heritage," *IEEE multimedia*, vol. 7, no. 2, pp. 22–25, 2000.

[109] J. Malpas, "Cultural heritage in the age of new media in kalay, y, kvan, t and affleck, j (eds) new heritage: New media and cultural heritage. abingdon," 2008.

[110] M. Roussou and D. Efraimoglou, "High-end interactive media in the museum," in *International Conference on Computer Graphics and Interactive Techniques: ACM SIGGRAPH 99 Conference abstracts and applications*, vol. 8, pp. 59–62, 1999.

[111] F. Bernardini, H. Rushmeier, I. M. Martin, J. Mittleman, and G. Taubin, "Building a digital model of michelangelo's florentine pieta," *IEEE Computer Graphics and Applications*, vol. 22, no. 1, pp. 59–67, 2002.

[112] G. Guidi, L. Micoli, M. Russo, B. Frischer, M. De Simone, A. Spinetti, and L. Carosso, "3d digitization of a large model of imperial rome," in *Fifth International Conference on 3-D Digital Imaging and Modeling (3DIM'05)*, pp. 565–572, IEEE, 2005.

[113] K. Ikeuchi, A. Nakazawa, K. Hasegawa, and T. Oishi, "The great buddha project: Modeling cultural heritage for vr systems through observation.," in *ISMAR*, vol. 3, pp. 7–16, 2003.

[114] E. Pietroni, A. Pagano, and C. Rufa, "The etruscanning project: gesture-based interaction and user experience in the virtual reconstruction of the regolini-galassi tomb," in *2013 digital heritage international congress (DigitalHeritage)*, vol. 2, pp. 653–660, IEEE, 2013.

[115] S. Brusaporci, F. Graziosi, F. Franchi, P. Maiezza, and A. Tata, "Mixed reality experiences for the historical storytelling of cultural heritage," in *From Building Information Modelling to Mixed Reality*, pp. 33–46, Springer, 2020.

[116] IMT-2020 (5G) Promotion Group, "5g vision and requirements," *Chinese Journal of Engineering*, 2014.

[117] D. Raj, S. S. Lekshmi, J. Guruprasad, M. Urmila, T. A. Lakshmi, A. Vinod, T. Swathi, *et al.*, "Enabling technologies to realise smart mall concept in 5g era," in *2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pp. 1–6, IEEE, 2018.

[118] R. Jurva, M. Matinmikko-Blue, V. Niemelä, and S. Nenonen, "Architecture and operational model for smart campus digital infrastructure," *Wireless Personal Communications*, pp. 1–18, 2020.

[119] S. Doukianou, D. Daylamani-Zad, and I. Paraskevopoulos, "Beyond virtual museums: Adopting serious games and extended reality," *Visual Computing for Cultural Heritage*, p. 283, 2020.

[120] D. Borsatti, G. Davoli, W. Cerroni, and C. Raffaelli, "Enabling industrial iot as a service with multi-access edge computing," *IEEE Communications Magazine*, vol. 59, no. 8, pp. 21–27, 2021.

[121] L. D'Errico, F. Franchi, F. Graziosi, A. Marotta, C. Rinaldi, M. Boschi, and A. Colarieti, "Structural health monitoring and earthquake early warning on 5g urllc network," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, pp. 783–786, 2019.

[122] A. Tesei, M. Luise, P. Pagano, and J. Ferreira, "Secure multi-access edge computing assisted maneuver control for autonomous vehicles," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, pp. 1–6, 2021.

[123] X. Jiang, F. R. Yu, T. Song, and V. C. M. Leung, “A survey on multi-access edge computing applied to video streaming: Some research issues and challenges,” *IEEE Communications Surveys Tutorials*, vol. 23, no. 2, pp. 871–903, 2021.

[124] W. Fohl, J. Reichardt, and J. Kuhr, “A system-on-chip platform for hrtf-based realtime spatial audio rendering with an improved realtime filter interpolation,” *International Journal on Advances in Intelligent Systems*, vol. 4, no. 3, pp. 309–317, 2011.

[125] G. Coluccelli, V. Loffredo, L. Monti, M. R. Spada, F. Franchi, and F. Graziosi, “5g italian mise trial: Synergies among different actors to create a “5g road”,” in *2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI)*, pp. 1–4, IEEE, 2018.

[126] A. Lapini, G. Calamai, F. Argenti, and M. Carfagni, “Application of binaural audio techniques for immersive fruition of cultural heritage,” *IOP Conference Series: Materials Science and Engineering*, vol. 364, p. 012099, 06 2018.

[127] D. D’Auria, D. Di Mauro, D. M. Calandra, and F. Cutugno, “A 3d audio augmented reality system for a cultural heritage management and fruition.,” *Journal of Digital Information Management*, vol. 13, no. 4, 2015.

[128] D. Murphy and F. Neff, “Spatial sound for computer games and virtual reality,” in *Game sound technology and player interaction: Concepts and developments*, pp. 287–312, IGI Global, 2011.

[129] D. D’Auria, D. D. Mauro, D. Calandra, and F. Cutugno, “A 3d audio augmented reality system for a cultural heritage management and fruition,” *Journal of Digital Information Management*, vol. 13, p. 203, 2015.

[130] D. P.-Q. et al., “Ghost orchestra a virtual reconstruction of the acoustics of notre dame catherdal.”

[131] m. r. schroeder and b. f. logan, “’colorless’ artificial reverberation,” *journal of the audio engineering society*, vol. 9, pp. 192–197, july 1961.

[132] V. Valimaki, J. D. Parker, L. Savioja, J. O. Smith, and J. S. Abel, “Fifty years of artificial reverberation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1421–1448, 2012.

[133] L. Nacke and M. Grimshaw-Aagaard, “Player-game interaction through affective sound,” *Games Computing and Creative Technologies: Book Chapters*, 01 2011.

[134] M. S. Smith, *Strategies for the Creation of Spatial Audio in Electroacoustic Music*. PhD thesis, University of North Texas, Denton, Texas, 12 2018.

[135] J. Y. Hong, J. He, B. Lam, R. Gupta, and W.-S. Gan, “Spatial audio for soundscape design: Recording and reproduction,” *Applied Sciences*, vol. 7, no. 6, 2017.

[136] T. Kuppanda, N. Degara, D. Worrall, B. Thoshkahna, and M. Müller, “Virtual reality platform for sonification evaluation,” 2015.

[137] M. Geier, S. Spors, and S. Weinzierl, “The future of audio reproduction,” in *International Workshop on Adaptive Multimedia Retrieval*, pp. 1–17, Springer, 2008.

[138] C. P. Brown and R. O. Duda, “A structural model for binaural sound synthesis,” *IEEE transactions on speech and audio processing*, vol. 6, no. 5, pp. 476–488, 1998.

[139] V. Välimäki, J. Parker, L. Savioja, J. O. Smith, and J. Abel, “More than 50 years of artificial reverberation,” in *Audio engineering society conference: 60th international conference: dreams (dereverberation and reverberation of audio, music, and speech)*, Audio Engineering Society, 2016.

[140] W.-S. Gan, S. Peksi, J. He, R. Ranjan, N. D. Hai, and N. K. Chaudhary, “Personalized hrtf measurement and 3d audio rendering for ar/vr headsets,” in *Audio Engineering Society Convention 142*, Audio Engineering Society, 2017.

[141] P. Majdak and M. Noisternig, “Aes69-2015: Aes standard for file exchange-spatial acoustic data file format,” in *Audio Engineering Society*, 2015.

[142] mySOFA, “Head related transfer functions (hrtf) locating sounds in 3d with your headphones.” `https://mysofa.audio/index.html`. [Online; accessed 20-June-2021].

[143] K. Hartung, J. Braasch, and S. J. Sterbing, “Comparison of different methods for the interpolation of head-related transfer functions,” in *Audio Engineering*

*Society Conference: 16th International Conference: Spatial Sound Reproduction*, Audio Engineering Society, 1999.

[144] W. Zhang, P. N. Samarasinghe, H. Chen, and T. D. Abhayapala, "Surround by sound: A review of spatial audio recording and reproduction," *Applied Sciences*, vol. 7, no. 5, 2017.

[145] C. Pike and F. Melchior, "An assessment of virtual surround sound systems for headphone listening of 5.1 multichannel audio," in *Audio Engineering Society Convention 134*, Audio Engineering Society, 2013.

[146] A. McKeag and D. S. McGrath, "Sound field format to binaural decoder with head tracking," in *Audio Engineering Society Convention 6r*, Audio Engineering Society, 1996.

[147] M. Gorzel, A. Allen, I. Kelly, J. Kammerl, A. Gungormusler, H. Yeh, and F. Boland, "Efficient encoding and decoding of binaural sound with resonance audio," in *Audio Engineering Society Conference: 2019 AES International Conference on Immersive and Interactive Audio*, Audio Engineering Society, 2019.

[148] r. shukla, i. t. radu, m. sandler, and r. stewart, "real-time binaural rendering with virtual vector base amplitude panning," *journal of the audio engineering society*, march 2019.

[149] B. Cowan and B. Kapralos, "Spatial sound for video games and virtual environments utilizing real-time gpu-based convolution," in *Proceedings of the 2008 Conference on Future Play: Research, Play, Share*, pp. 166–172, 2008.

[150] B. Kapralos and N. Mekuz, "Application of dimensionality reduction techniques to hrtfs for interactive virtual environments," in *Proceedings of the International Conference on Advances in Computer Entertainment Technology*, ACE '07, (New York, NY, USA), p. 256–257, Association for Computing Machinery, 2007.

[151] Molex and dimensional research, "Timing is everything: carriers and the state of 5G. A Survey of R&D, Engineering, and Product Stakeholders," March 2021. [Online; accessed 16-July-2021].

[152] B. Ethirajulu, "How 5g and edge computing can enhance virtual reality." [Online; accessed 16-July-2021].

[153] B. O. Donnel, “How 5g could make augmented reality and mixed reality more real,” March 2020. [Online; accessed 16-July-2021].

[154] “Immersive experiences with 5g, information can surround us in a 3d digital world,” March 2020. [Online; accessed 16-July-2021].

[155] J. Stanislow, “5g impact on virtual reality (vr) and augmented reality (ar),” March 2020. [Online; accessed 16-July-2021].

[156] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, “A survey on mobile augmented reality with 5g mobile edge computing: Architectures, applications, and technical aspects,” *IEEE Communications Surveys Tutorials*, vol. 23, no. 2, pp. 1160–1192, 2021.

[157] R.-S. Schmoll, S. Pandi, P. J. Braun, and F. H. Fitzek, “Demonstration of vr / ar offloading to mobile edge cloud for low latency 5g gaming application,” in *2018 15th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pp. 1–3, 2018.

[158] K. Katsaros, D. Gkounis, D. Kaleshi, B. Thomas, J. Harris, H. Falaki, and D. Simeonidou, “Enhancing tourist experiences through 5g - the 5g smart tourism case study,” in *2019 IEEE 2nd 5G World Forum (5GWF)*, pp. 471–476, 2019.

[159] X. Qiao, P. Ren, S. Dustdar, L. Liu, H. Ma, and J. Chen, “Web ar: A promising future for mobile augmented reality—state of the art, challenges, and insights,” *Proceedings of the IEEE*, vol. 107, no. 4, pp. 651–666, 2019.

[160] X. Qiao, P. Ren, G. Nan, L. Liu, S. Dustdar, and J. Chen, “Mobile web augmented reality in 5g and beyond: Challenges, opportunities, and future directions,” *China Communications*, vol. 16, no. 9, pp. 141–154, 2019.

[161] M. Cuevas-Rodríguez, L. Picinali, D. González-Toledo, C. Garre, E. de la Rubia-Cuestas, L. Molina-Tanco, and A. Reyes-Lecuona, “3d tune-in toolkit: An open-source library for real-time binaural spatialisation,” *PLOS ONE*, vol. 14, pp. 1–37, 03 2019.

[162] Various, “Sofalizer for unity.” `https://github.com/sofacoustics/SOFAlizer-for-Unity`. [Online; accessed 01-April-2021].

[163] Various, "Spatial_audio_framework." `https://github.com/leomccormack/Spatial_Audio_Framework`. [Online; accessed 01-April-2021].

[164] Various, "Resonance audio sdk." `https://github.com/resonance-audio/resonance-audio`. [Online; accessed 16-June-2021].

[165] B. Schäling, *The boost C++ libraries.* Boris Schäling, 2011.

[166] F. Vatalaro and G. Ciccarella, "A network paradigm for very high capacity mobile and fixed telecommunications ecosystem sustainable evolution," *IEEE Access*, vol. 8, pp. 135075–135090, 2020.

[167] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the tcp congestion avoidance algorithm," *SIGCOMM Comput. Commun. Rev.*, vol. 27, p. 67–82, July 1997.

[168] T. C. Thang, Q.-D. Ho, J. W. Kang, and A. T. Pham, "Adaptive streaming of audiovisual content using mpeg dash," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 1, pp. 78–85, 2012.