



**EXPERIMENTATION AND VALIDATION OPENNESS FOR LONGTERM
EVOLUTION OF VERTICAL INDUSTRIES IN 5G ERA AND BEYOND**

[H2020 - Grant Agreement No.101016608]

Deliverable D5.1

System level evaluation and KPI analysis (Intermediate)

Editor B. García (UMA)

Contributors TID, UMA, NCSR, INTRA, COS, INF, MAG, ATOS,
LNV, IEA, UPV, GMI, ININ, CAF, IQBT, FOGUS,
8BELLS, PAL, ZORT, IMM, UMS

Version 1.0

Date August 31st, 2022

Distribution PUBLIC (PU)

Telefonica
Telefónica
Investigación y Desarrollo



Atos

INTRASOFT
INTERNATIONAL

COSMOTÉ

Lenovo

envolve
ENTREPRENEURSHIP

**UNIVERSIDAD
DE MÁLAGA**

**UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA**

**GMI
AERO**

**Internet
INSTITUTE**

CAFA

InQbit
The Q-Bit Innovation

FOGUS
INNOVATION & SERVICES

**INFO
LYSIS**

EIGHTBELLS
INNOVATION & SERVICES

**PAL
ROBOTICS**

Qucomm
5G tech & beyond

IMMERSION
Imagination. Interaction...

**Autonomy-as-a-Service
Ulife**

DISCLAIMER

This document contains information, which is proprietary to the EVOLVED-5G ("Experimentation and Validation Openness for Longterm evolution of VERTICAL industries in 5G era and beyond) Consortium that is subject to the rights and obligations and to the terms and conditions applicable to the Grant Agreement number: 101016608. The action of the EVOLVED-5G Consortium is funded by the European Commission.

Neither this document nor the information contained herein shall be used, copied, duplicated, reproduced, modified, or communicated by any means to any third party, in whole or in parts, except with prior written consent of the EVOLVED-5G Consortium. In such case, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced. In the event of infringement, the consortium reserves the right to take any legal action it deems appropriate.

This document reflects only the authors' view and does not necessarily reflect the view of the European Commission. Neither the EVOLVED-5G Consortium as a whole, nor a certain party of the EVOLVED-5G Consortium warrant that the information contained in this document is suitable for use, nor that the use of the information is accurate or free from risk, and accepts no liability for loss or damage suffered by any person using this information.

The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

REVISION HISTORY

Revision	Date	Responsible	Comment
0.1	June 07, 2022	B. Garcia (UMA)	Table of Content
0.5	July 20, 2022	B. Garcia (UMA)	First integration
0.6	July 26, 2022	B. Garcia (UMA)	Additional content, second editing pass
0.7	July 28, 2022	B. García (UMA)	Version for internal review
0.8	August 08, 2022	G. Makropoulos (NCSRD) H. Koumaras (NCSRD)	Internal review and revision of comments
0.9	August 15, 2022	D. Artuñedo (TID) J. Garcia (TID)	Internal review and revision of comments
1.0	August 29, 2022	B. García (UMA)	Final version

LIST OF AUTHORS

Partner ACRONYM	Partner FULL NAME	Name & Surname
<i>TID</i>	<i>TELEFONICA INVESTIGACIÓN Y DESARROLLO</i>	<i>Javier García Alejandro Molina David Artuñedo</i>
<i>UMA</i>	<i>UNIVERSITY OF MÁLAGA</i>	<i>Almudena Díaz Bruno García Francisco Luque M^a del Mar Moreno M^a del. Mar Gallardo Pedro Merino</i>
<i>NCSRD</i>	<i>NATIONAL CENTER FOR SCIENTIFIC RESEARCH “DEMOKRITOS”</i>	<i>H. Koumaras G. Makropoulos D. Fragkos A. Gogos</i>
<i>INTRA</i>	<i>Netcompany Intrasoft</i>	<i>Angela Dimitriou</i>
<i>COS</i>	<i>COSMOTE Mobile Telecommunications S.A</i>	<i>F. Setaki I. Messogiti</i>
<i>INF</i>	<i>INFOLYSIS</i>	<i>T. Dounia C. Sakkas</i>
<i>FOGUS</i>	<i>FOGUS INNOVATIONS & SERVICES P.C.</i>	<i>D.Tsolkas A-S.Charismiadis</i>

GLOSSARY

Abbreviations/Acronym	Description
3GPP	<i>3rd Generation Partnership Project</i>
5G NR	<i>5G New Radio</i>
AF	<i>Application Function</i>
AMF	<i>Access and Mobility Management Function</i>
API	<i>Application Programming Interface</i>
CAPIF	<i>Common API Framework</i>
CI/CD	<i>Continuous Integration / Continuous deployment</i>
COTS	<i>Commercial off-the-shelf</i>
DS	<i>Device Side</i>
E2E	<i>End to End</i>
EC	<i>European Commission</i>
ELCM	<i>Experiment Life-Cycle Manager</i>
EN_DC	<i>E-UTRAN New Radio – Dual Connectivity</i>
EPC	<i>Evolved Packet Core</i>
ETSI	<i>European Telecommunications Standards Institute</i>
FOF	<i>Factories of the Future</i>
gNodeB / gNB	<i>Next Generation (5G) Base Station</i>
GPS	<i>Global Positioning System</i>
GUI	<i>Graphical User Interface</i>
ICMP	<i>Internet Control Message Protocol</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
KPI	<i>Key Performance Indicator</i>
LTE	<i>Long Term Evolution</i>
MANO	<i>Management and Orchestration</i>
MIMO	<i>Multiple-Input Multiple-Output</i>
MCS	<i>Modulation Coding Scheme</i>
NEF	<i>Network Exposure Function</i>
NS	<i>Network Service</i>
NSA	<i>Non-Standalone</i>
NSD	<i>Network Service Descriptor</i>
NSI	<i>Network Slice Instance</i>
NST	<i>Network Slice Template</i>
NW-TT	<i>Network-Side TSN Translator</i>
OFDM	<i>Orthogonal Frequency Division Multiplexing</i>
OSM	<i>Open-Source MANO</i>
P4	<i>Programming Protocol-independent Packet Processors</i>
PCF	<i>Point Coordination Function</i>
PCP	<i>Priority Code Point</i>
PRB	<i>Physical Resource Block</i>
QAM	<i>Quadrature Amplitude Modulation</i>
QoS	<i>Quality of Service</i>



D5.1 - System level evaluation and KPI analysis (Intermediate)

GA Number 101016608

RAN	<i>Radio Access Network</i>
RRH	<i>Remote Radio Head</i>
RTT	<i>Round-Trip Time</i>
SA	<i>Standalone</i>
SMF	<i>Session Management Function</i>
SUT	<i>System Under Test</i>
TDD	<i>Time Division Duplex</i>
TSN	<i>Time-Sensitive Networking</i>
TT	<i>TSN Translator</i>
UDP	<i>User Datagram Protocol</i>
UE	<i>User Equipment</i>
UPF	<i>User Plane Function</i>
URLLC	<i>Ultra-Reliable Low-Latency Communication</i>
VM	<i>Virtual Machine</i>
VIM	<i>Virtual Infrastructure Manager</i>
VNF	<i>Virtual Network Function</i>
VNFD	<i>Virtual Network Function Descriptor</i>

EXECUTIVE SUMMARY

The purpose of this document is to present the initial results of the system level evaluations performed to the EVOLVED-5G platforms during the first phase (i.e., Release A) of the project. It also includes the results of the verification tests and the initial analysis of the performance of the software tools developed within the context of the project, namely the CAPIF Tool and NEF emulator.

The deliverable is the result of the work carried out in Tasks 5.1 of EVOLVED-5G project, where main goals is to apply the verification and validation methodology in order to test and quantify a set of Key Performance Indicators (KPIs), in order to assess the capabilities of the EVOLVED-5G infrastructure and software components.

Based on the above, the main contribution of this deliverable is to provide a description of the initial results obtained during the initial validation phase of the platforms (Malaga and Athens) and software components, as well as to provide details about the implementation of the verification and validation processes in the platforms. This deliverable includes:

- A small recap of the EVOLVED-5G methodology.
- A description and the results of the initial performance assessment of the validation platforms.
- Details about the verification and validation of the CAPIF Tool and NEF Emulator, as well as the obtained results.

A summary of the obtained results can be seen in the following table:

Throughput	NCSRD Demokritos Site	Downlink Median	338.83 Mbps
		Uplink (2 UL slots) Median	43.6 Mbps
		Uplink (8 UL slots) Median	210.5 Mbps
	Cosmote Site	Downlink Median	628.7 Mbps
		Uplink Median	52.7 Mbps
	Málaga Site	Downlink Median	472.6 Mbps
Round Trip Time	NCSRD Demokritos Site	Median	31.54 ms
		Median (low latency)	7.26 ms
	Cosmote Site	Median	14.4 ms
	Málaga Site	Median	12.546 ms
		Median (TSN)	14.5 ms
Average Access Time	NEF Emulator	Average of all tests	25.115 ms
	CAPIF Tool	Average of all tests	34.027 ms

Deliverable D5.4 “System level evaluation and KPI analysis (Final)”, to be delivered in month M34, will complement this deliverable and present the evolution of the platform’s results towards the end of the EVOLVED-5G project.

TABLE OF CONTENTS

1	INTRODUCTION	1
1.1.	Document Purpose	1
1.2.	Document Structure	1
1.3.	Target Audience	1
2	Initial Platform Assessment	3
2.1	The EVOLVED-5G Experimentation Methodology	3
2.2	The EVOLVED-5G Platforms	3
2.2.1	The Athens Platform	3
2.2.2	The Málaga Platform	6
2.3	Baseline Tests and Results	9
2.3.1	Athens Platform Tests and Results	9
2.3.2	Málaga Platform Tests and Results	17
3	Component-Level Evaluation	23
3.1	EVOLVED-5G Software Components	23
3.2	Functional Testing	23
3.2.1	NEF Emulator results	23
3.2.2	CAPIF Tool results	28
3.3	Performance Testing	33
3.3.1	NEF Emulator	36
3.3.2	CAPIF Tool	37
4	CONCLUSION	39
5	REFERENCES	40
6	ANNEXES	42
6.1	Test Case Template	42
6.2	Athens platform Test Case Templates	43
6.2.1	DL throughput (NCSRD Demokritos)	43
6.2.2	UL throughput (NCSRD Demokritos)	44
6.2.3	Best UL throughput (NCSRD Demokritos)	45
6.2.4	DL throughput (Cosmote)	46
6.2.5	UL throughput (Cosmote)	48
6.2.6	RTT (NCSRD Demokritos)	49
6.2.7	RTT low latency (NCSRD Demokritos)	50
6.2.8	RTT (Cosmote – NCSRD Demokritos)	51



D5.1 - System level evaluation and KPI analysis (Intermediate)

GA Number 101016608

6.3	UMA platform Test Case Templates	53
6.3.1	DL throughput (UMA)	53
6.3.2	RTT (UMA)	54

LIST OF FIGURES

Figure 1. Coordination & MANO layer VMs - OpenStack	5
Figure 2. Athens Platform interconnection	6
Figure 3. Coordination Layer in the Málaga platform.....	7
Figure 4. Asus I007D, Askey 5G NR ODU RTL6305 and Askey 5G NR USB dongle NDQ1300	8
Figure 5. TSN architecture	8
Figure 6. NCSR D site testbed setup	9
Figure 7. Cosmote - NCSR D sites testbed setup	10
Figure 8. DL throughput primary results (NCSR D).....	12
Figure 9. DL throughput complementary results (NCSR D).....	12
Figure 10. UL throughput primary results (NCSR D).....	13
Figure 11. UL throughput complementary results (NCSR D)	13
Figure 12. Best UL throughput primary results (NCSR D).....	13
Figure 13. Best UL throughput complementary results (NCSR D)	14
Figure 14. DL throughput primary results (COSMOTE + NCSR D)	14
Figure 15. DL throughput complementary results (COSMOTE + NCSR D)	15
Figure 16. UL throughput primary results (COSMOTE + NCSR D)	15
Figure 17. UL throughput complementary results (COSMOTE + NCSR D)	15
Figure 18. RTT - 64byte packet size (NCSR D)	16
Figure 19. RTT – low latency - 64byte packet size (NCSR D).....	16
Figure 20. RTT - 64byte packet size (COSMOTE + NCSR D).....	17
Figure 21. UMA Radio deployment	17
Figure 22. Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM throughput per iteration (UMA)	18
Figure 23. Micro cells 5G SA MIMO 4x4 256 QAM throughput temporal evolution (UMA)	18
Figure 24. Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT per iteration (UMA)	20
Figure 25. 2221Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT temporal evolution (UMA)	20
Figure 26. 5GCore usage statistics part 1 (UMA)	21
Figure 27. 5GCore usage statistics part 2 (UMA)	21
Figure 28. TSN over 5G - delay of the test	22
Figure 29. TSN over 5G - jitter of the test	22
Figure 30. NEF Emulator Results – Test Report.....	28
Figure 31. __init__.robot file.....	30
Figure 32. <component_name>.robot file	31
Figure 33. CAPIF Tool Results – Test Report	33
Figure 34. Common performance tests implementation	35

LIST OF TABLES

Table 1. 5G SA Configuration at UMA testbed	17
Table 2 Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM throughput statistical analysis (UMA)	19
Table 3 Complementary radio measurements for throughput test (UMA)	19
Table 4 Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT statistical analysis (UMA)	20
<i>Table 5 Complementary radio measurements for RTT tests (UMA)</i>	21
Table 6 Testing plan targeting the MonitoringEvent and AsSessionWithQoS APIs	24
Table 7 Testing plan targeting the API Invoker and API Publisher of CAPIF	28
Table 6 NEF Emulator performance test result	36
Table 9 CAPIF Tool performance test result	37

1 INTRODUCTION

1.1. DOCUMENT PURPOSE

The main goal of this document, titled “*System level evaluation and KPI analysis (Intermediate)*” is to present the results obtained during the evaluation of the EVOLVED-5G platforms (Athens and Málaga) towards the end of the first phase (i.e., Release A) of the project. These measurements reflect the improvements stemming from the updates performed to the platforms during the first 20 months of the project, and have been realized through the experimentation methodology defined in Work Package 2 as well as by utilizing tools developed in Work Packages 3 and 4.

1.2. DOCUMENT STRUCTURE

The document is divided into two main sections:

- **Section 2. Initial Platform Assessment:** This section is divided into three sub-sections:
 - Section 2.1: The EVOLVED-5G Experimentation Methodology presents a summary of the experimentation methodology followed by the EVOLVED-5G project.
 - Section 2.2: The EVOLVED-5G Platforms presents a complete and updated description of the EVOLVED-5G infrastructure platforms (Malaga and Athens).
 - Section 2.3: Baseline Tests and Results summarizes the results obtained during the first evaluation of the platforms (i.e., Release A).
- **Section 3. Component-Level Evaluation:** Section 3 is devoted to the evaluation of the CAPIF Tool and NEF Emulator as software components developed within the context of EVOLVED-5G and is separated into three sub-sections:
 - For context purposes, Section 3.1: EVOLVED-5G Software Components presents a brief description for both the CAPIF Tool and NEF Emulator.
 - Section 3.2: Functional Testing describes the process and results of the functional verification of the two components.
 - Section 3.3: Performance Testing describes the performance (non-functional) testing of the two components.

1.3. TARGET AUDIENCE

The release of the deliverable is public, intending to showcase the current results and status of the platforms and software components of EVOLVED-5G. From specific to broader, different target audiences for D5.1 are identified below:

- **Project Consortium:** To validate the evolution of the EVOLVED-5G platforms, the correctness of the evaluation methodology defined in WP2, and the usability of the tools implemented in WP 3 and 4, as well as to serve as the baseline results to the evolution to be presented in Deliverable D5.4.
- **Industry 4.0/Industry 4.0 developers, FoF (Factories of the Future) and other vertical industries and groups:** To showcase the performance and available features in the EVOLVED-5G platforms, which may raise awareness and interest in other industrial partners in the project achievements.
- **The scientific audience, general public and the funding EC Organization:** The scientific audience can get access to the performance results measured in three separate sites



D5.1 - System level evaluation and KPI analysis (Intermediate)

GA Number 101016608

that form the two EVOLVED-5G platforms, which can be used as a baseline for future investigation. This deliverable also documents the work carried out by the Project Consortium and justifies the effort reported in the corresponding activities.

2 INITIAL PLATFORM ASSESSMENT

2.1 THE EVOLVED-5G EXPERIMENTATION METHODOLOGY

The EVOLVED-5G experimentation methodology is an adaptation of the methodology defined in the 5Genesis project [1], which is based on the definition of Test Cases. Test Cases follow the template shown in Annex 6.1, which includes:

- A short description of the test.
- A listing of any necessary pre-conditions and assumptions that need to be verified before the test execution.
- A description of the target KPI, including any measurement methods or calculations required for obtaining it, and
- The sequence of steps to follow, either manually or automatically, during the test execution. A more complete description of the Test Cases and the methodology can be found in Deliverable D2.2 [2], section 6.4.

Once defined, Test Cases are implemented in the particular testing environment where the tests will be conducted. The process includes any preparation needed for meeting the Test Case pre-conditions (such as installing and configuring any hardware or software requirement), the implementation of any additional functionality required as indicated by each step in the test sequence (especially in the case of automated tests) or any partial testing required to ensure that the Test Case can be correctly executed in the testing environment.

2.2 THE EVOLVED-5G PLATFORMS

The EVOLVED-5G project makes use of two different platforms located in Athens (composed by two sites: NCSR Demokritos and Cosmote) and Málaga. The two platforms provide 5G radio capabilities and make use of the Open5Genesis framework for the coordination of the experiments.

2.2.1 The Athens Platform

For the deployment of the Athens platform, the Open5Genesis infrastructure components have been installed and integrated. The full process for the Open5Genesis Suite integration is described in Deliverables D5.2 [19] and D5.4 [20] of the 5Genesis Project.

The 5Genesis approach dictates three layers that smoothly cooperate to provide the experiment platform. Briefly these layers are:

- **Management and Orchestration (MANO)** layer, which handles functionality related to virtualization, network slices and virtual resources management.
- **Coordination layer**, which is responsible for the overall coordination of the experiments, including experiments' life cycle management, KPIs monitoring and analytic results presentation.
- **Infrastructure layer**, which handles user traffic providing 5G Core Network's connectivity.

In order to achieve a modular approach, **a set of virtual machines have been deployed both for Coordination and MANO layers** in a cloud computing infrastructure manager (OpenStack). In

total, seven virtual machines (VMs) have been deployed as well as properly configured to serve each of the components required for the setup of the experimentation platform.

Following a bottom-up approach, deployments of virtual machines per layer are described in the following paragraphs.

For the **MANO layer implementation**, the deployed VMs are as follows:

- Initially ETSI open-source MANO (OSM) has been deployed to implement network function virtualization management and orchestration. This component manages Network Services (NS) and therefore Virtualized network Functions (VNF). Network service descriptors and connected virtual network function descriptors are onboarded and referenced by the slice manager, in order to create network slices as required by the experiments.
- As a next step a VM with Open5Genesis Slice Manager has been deployed. Slice Manager is configured to interconnect with the OSM. Network slice templates (NSTs) are defined in the slice manager, referencing NSs in OSM. Basic MANO functionalities, e.g., network services lifecycle management, are delegated to Slice Manager.

Additionally, for the Network Service instantiation upon network slice deployment, two separate virtual infrastructure managers (VIMs), one at the core network and one at the edge of NCSRD premises, have been integrated with OSM and Slice Manager. Virtual machines required for experiments' execution are instantiated in these infrastructure managers. For example, machines running as probes of the 5Genesis service like iPerf server for throughput measurements, or Ping server tool for Round Trip Time or delay (RTT) measurements are instantiated in these VIMs as part of the Network Services required for the experiment's execution.

Athens' platform Coordination layer consists of five virtual machines, serving each of the Open5Genesis required components for the experiment's coordination. Those virtual machines cooperating smoothly for the experiment's creation, life cycle management and results retrieval and presentation are the following:

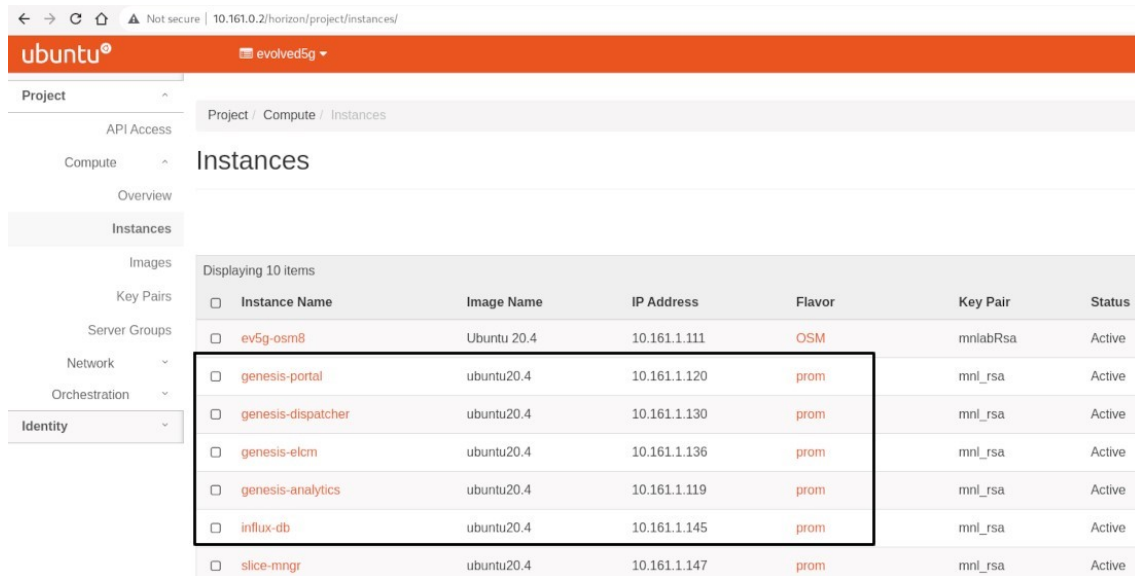
- A machine which serves as the experiment's metrics persistence and graphical presentation component. For this purpose, an InfluxDB [13], a time series database where all experiments defined metrics and KPIs are persisted once experiments have been executed, and Grafana [14] monitoring software have been installed.
- An Analytics component which provides methods for analyzing and offline learning on the data and is responsible for the monitoring of the platform. It retrieves data from the InfluxDB component and enables users to perform KPI statistical validation. Statistical analysis is presented in meaningful dashboards exposed by the specific platform thus, giving access to performance indicators statistical data as for example median, min, max, q1, q3 values. The Analytics component also provides the possibility to a user to perform a cross-correlation across fields of the same or different experiments.
- The Experiment Life Cycle Manage, or ELCM, which is the "heart" of the coordination layer of the 5GENESIS architecture and is responsible for the scheduling and execution of experiments. It handles the life cycle of an experiment keeping the experiment in an internal queue until all the required resources for the experiment are available. It uses independent executors to run the experiment and recovers the generated results. In the same machine the OpenTAP software component is installed to which the execution of automated tests is delegated. OpenTAP communicates with both deployed VMs and

User Equipment (UEs) at the infrastructure layer instructing the experiment sequence and finally retrieving the results.

- The Dispatcher component, which is the entry point of the system, offering the functionalities to an Experimenter through a single interface. These functionalities are known as the Open APIs, being able to interact with the key features of the underlying modules (as described above) without actually exposing them.
- The Open5Genesis Portal which provides an intuitive Graphical User Interface (GUI) for all experiments infrastructure stakeholders in order to create, run and monitor (in real time) experiments execution. Once experiment has been completed, links to KPIs and metrics' graphical and statistical dashboards are provided in order to obtain insights on the results.

The **Infrastructure layer** is comprised of two testbeds which are described in Section 2.3.1.

In the following Figure 1 all VMs instantiated, for the Athens experimentation platform, in OpenStack VIM are presented. The five VMs squared in black box compose the Coordination Layer components and the two remaining (ev5g-osm8 and slice-mngr) are those included within the MANO Layer.



Instance Name	Image Name	IP Address	Flavor	Key Pair	Status
ev5g-osm8	Ubuntu 20.4	10.161.1.111	OSM	mmlabRsa	Active
genesis-portal	ubuntu20.4	10.161.1.120	prom	mml_rsa	Active
genesis-dispatcher	ubuntu20.4	10.161.1.130	prom	mml_rsa	Active
genesis-eicm	ubuntu20.4	10.161.1.136	prom	mml_rsa	Active
genesis-analytics	ubuntu20.4	10.161.1.119	prom	mml_rsa	Active
influx-db	ubuntu20.4	10.161.1.145	prom	mml_rsa	Active
slice-mngr	ubuntu20.4	10.161.1.147	prom	mml_rsa	Active

Figure 1. Coordination & MANO layer VMs - OpenStack

All virtual machines are deployed on Linux (Ubuntu 20.4 flavor) operating system and resources allocated are the same for all of them (2 CPUs, 4GB RAM, 30 GB disk size) except from the OSM8 virtual machine for which 2 CPUs, 8GB RAM, 40 GB disk size are allocated. Last, but not least, once Open5Genesis machines were deployed they were configured accordingly in order to ensure their interconnection and thus proper information flow. Figure 2 depicts the configured networking connections within infrastructure's components.

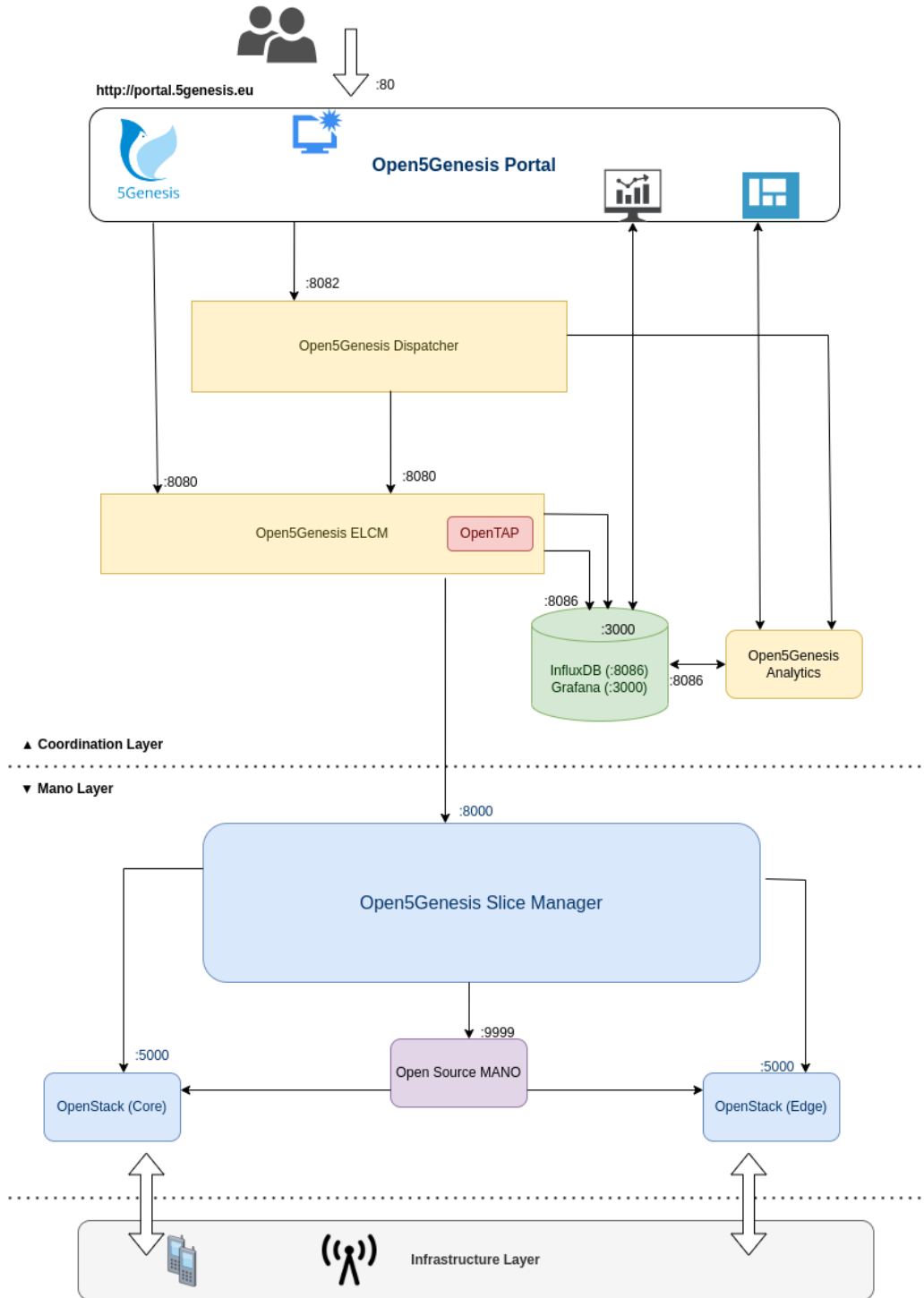


Figure 2. Athens Platform interconnection

2.2.2 The Málaga Platform

As in the Athens platform, the Open5Genesis Suite has been integrated in the Málaga Platform and is being used for the coordination of experiments and the processing of results. The same architecture, as described in the above section is in use, however, in the case of Málaga the Coordination layer has been deployed in a single physical computer (Windows 10 Pro, Intel Core i7-7700, 32GB of RAM), with components installed either directly in the host OS, or inside separate Linux virtual machines using VirtualBox, as it can be seen in Figure 3. The Slice Manager and underlying MANO layer are not currently configured due to an ongoing

reorganization of the virtualization infrastructure in the Málaga Platform. It was decided to prioritize the transition to Kubernetes over VNFs deployment automation, since support for containerization is critical for the validation of NetApps, and other ad-hoc solutions are available in the case a virtual machine needs to be available during the tests.

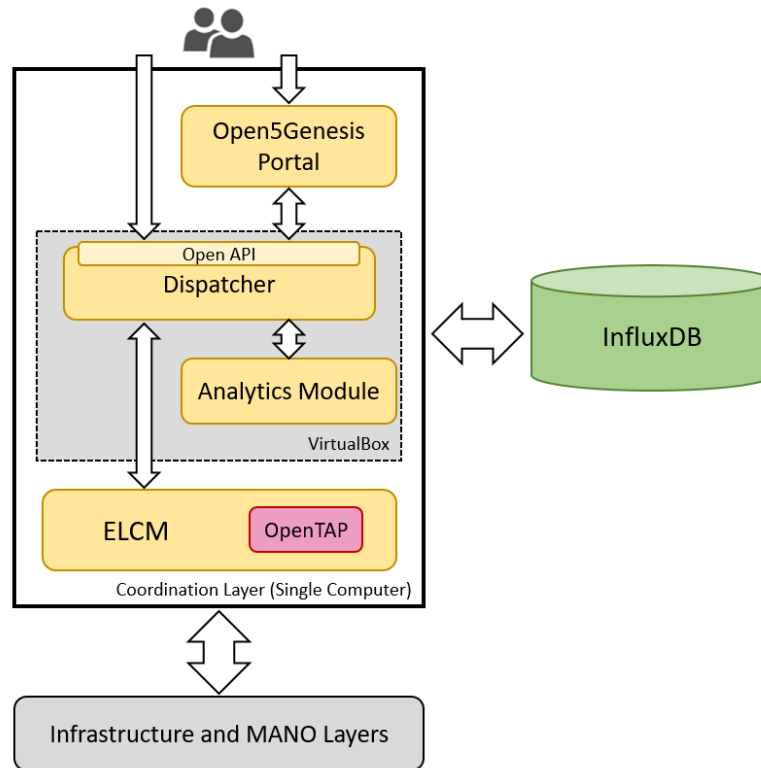


Figure 3. Coordination Layer in the Málaga platform

2.2.2.1 Radio evolution, user equipment and devices

In Malaga's site and within Universidad of Malaga (UMA) premises, the Nokia radio infrastructure has been upgraded with the new software version of radio equipment 5G21B. This release includes all the new features required in order to meet the latest 3GPP Release 15 requirements.

Additionally, the 4 Remote Radio Heads (RRHs) that provide outdoor 5G Stand Alone (SA) have been configured with 4x4 Multiple Input Multiple Output (MIMO) and a bandwidth of 50 MHz. The 4x4 MIMO feature provides improved cell coverage and DL throughput by increasing the maximum number of the gNB TX antennas and Data Layers (DL) MIMO per UE to four. Additionally, UMA has acquired new UEs and modems for testing the SA mode and millimeter wave (mmWave) performance, which can be seen in Figure 4:

- **Asus I007D:** Supports mmWave bands 257(26 GHz) and 258(28 GHz). Supports Standalone (SA) & Non-Standalone (NSA) network and is compatible with the Nemo Handy measurement solution [12].
- **Askey 5G NR ODU RTL6305:** Supports mmWave bands 5G: n257, n258. Supports SA & NSA network. Support Long Term Evolution (LTE) 4x4 MIMO & 5G Sub 6GHz 4x4 MIMO.
- **Askey 5G NR USB dongle NDQ1300:** Supports 5G New Radio (NR) Sub 6GHz and 5G NR E-UTRAN New Radio – Dual Connectivity (EN_DC).



Figure 4. Asus 1007D, Askey 5G NR ODU RTL6305 and Askey 5G NR USB dongle NDQ1300

2.2.2.2 Time-Sensitive Networking (TSN) over 5G

Time-Sensitive Networking includes a set of standards focused on improving the reliability of network communication and the transmission of data with very low latency, created by the IEEE 802.1 Time-Sensitive Networking Task Group [17]. TSN is usually implemented on top of fixed networks, however, the efforts documented in this section are geared to the implementation of such capabilities over a 5G mobile network.

The TSN infrastructure in Malaga is mainly composed of one TSN bridge, two TSN end stations and the TSN translators (TT): Network-Side (NW) TT next to the User Plane Function (UPF) and Device-Side (DS) TT close to the 5G UE. The architecture is shown in Figure 5 and was explained in more detail in Deliverable D3.1 [7]. However, since then, it has been improved in terms of implementation: The main updates are coming in the TSN translators (DS-TT and NW-TT), which support traffic prioritization through the 5G network and time synchronization.

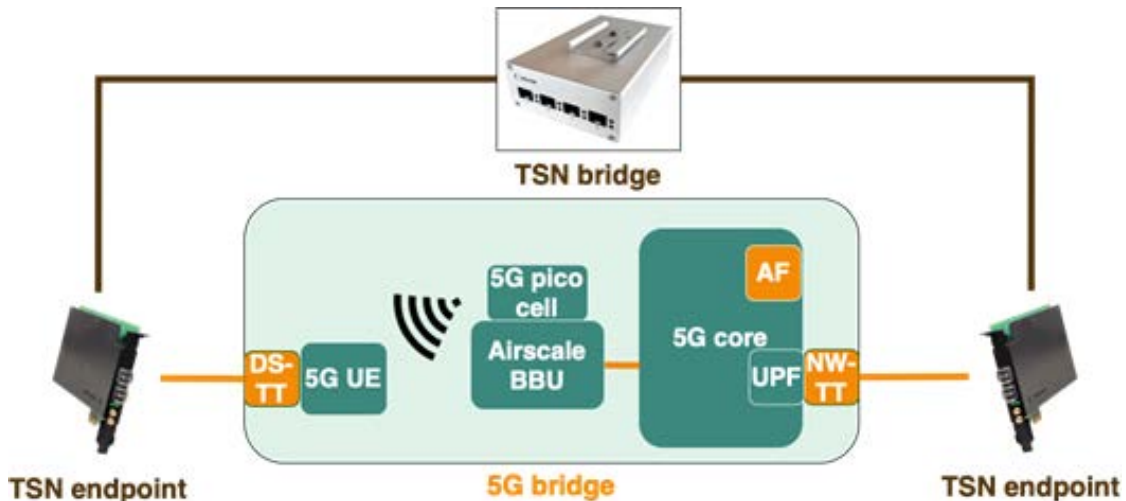


Figure 5. TSN architecture

Both TSN translators, NW-TT and DS-TT, have been developed using the Programming Protocol-independent Packet Processors (P4) language [15], which allows us to modify the packet headers (specifically 802.1Q) in order to add or remove them and send the packets from one TSN endpoint to other TSN endpoint through the 5G network. These TSN translators will consider the priority mapping between the TSN domain (using the Priority Code Point-PCP field in the 802.1Q header) and the 5G network (using 5G QoS Identifiers (5QI)).

The other key challenge is time synchronization through the 5G network. Since we are on 3GPP release 15, as a first step, we address the time synchronization configuring two master clocks at the edges of the network with a single Global Positioning System (GPS) signal, so we assume that the two TSN endpoints are synchronized. Finally, the Application Function (AF), which is the main TSN development coming from the EVOLVED-5G project, will manage the 5G network configuration to attain the requirements imposed by the traffic. The AF is a critical component in order to demonstrate the possibility of coexistence between TSN and 5G. The Ultra-Reliable Low-Latency Communication (URLLC) capabilities make 5G a suitable candidate for deterministic and time-sensitive wireless communication and, in turn, TSN has been selected by Institute of Electrical and Electronics Engineers (IEEE) as the standard Ethernet-based technology for converged networks of Industry 4.0. The AF, together with the Policy Control Function (PCF) will manage the QoS alignment between the 5G and TSN domains.

2.3 BASELINE TESTS AND RESULTS

2.3.1 Athens Platform Tests and Results

The initial assessment of the Athens platform is analyzed in the following section. The first stage of the evaluation process is grounded on components described in Section 3.1.1., where different aspects of the platform are taken into consideration. As a first step, the tests are described in detail using the test cases templates taken from the 5Genesis project [16] and customized for EVOLVED-5G. Afterwards, the execution of the experiments is realized using the Open5Genesis experimentation framework and final results are conducted from the Analytics framework, fulfilling high granularity. The assessment involves both NCSR D Demokritos and Cosmote sites comprising the Athens platform as described in D2.2 [2]. For NCSR D Demokritos site, Figure 6 illustrates the test setup for all experiments between endpoints A and B. The experiments include throughput for both downlink and uplink and end to end (E2E) RTT for “standard” and low latency Radio Access Network (RAN) configurations. On the other hand, Cosmote site enables a complete multi-domain 5G NSA solution where the RAN part is deployed within Cosmote’s premises and both Evolved Packet Core (EPC) and the Open5Genesis experimentation framework are hosted on NCSR D Demokritos premises. Moreover, the two sites are connected through the GRNET on top of the IEEE 802.1ad standard also known as QinQ.

A simplified overview of the testbed setup is depicted in Figure 7. Details of the overall experimentation phase are presented below.

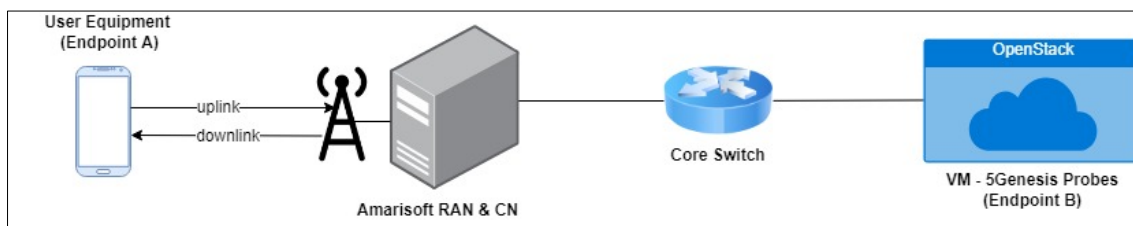


Figure 6. NCSR D site testbed setup

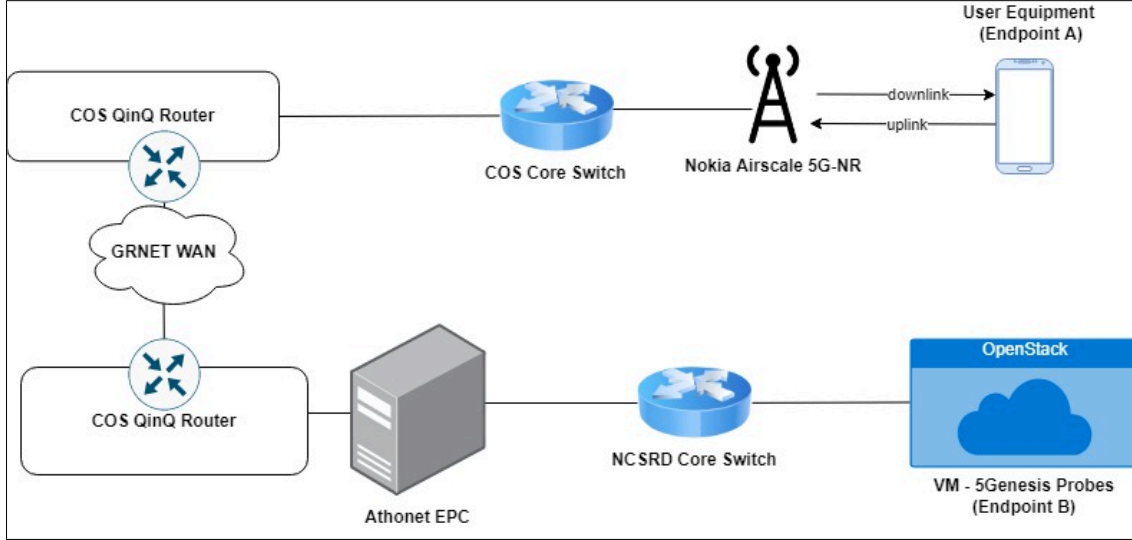


Figure 7. Cosmote - NCSR D sites testbed setup

2.3.1.1 Throughput

Throughput experiments have been executed based on test case templates NCSR D_Downlink, NCSR D_uplink, NCSR D_best_uplink, evaluating downlink, uplink and best uplink respectively. More details are described on Annex 6.2 including the scenario, the testing infrastructure, the target KPI and the test case sequence implying the execution steps. The System Under Test (SUT) involves the commercial Amarisoft Classic (i.e., both 5G-NR and 5GC Rel. 16), one Commercial off-the-shell (COTS) UE and a VM that integrates the 5Genesis iPerf probe. All the tests were conducted in a lab environment with perfect channel conditions leading to an approximate 26 Modulation and Coding Scheme (MCS) value. Finally, the results of each experiment are compared to the theoretical value based on equation (1), adopted from 3GPP TS 38.306 [3]:

$$\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{\text{Layers}}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{\text{PRB}}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$$

Where:

- j is the number of the aggregated component carriers which is **1** since one component carrier is used
- $v_{\text{Layers}}^{(j)}$ is the maximum number of Multiple-input / multiple output (MIMO) layers, which is **2** for downlink and **1** for uplink
- $Q_m^{(j)}$ is the modulation order, which is **8** considering MCS 26
- $f^{(j)}$ is the scaling factor and can take the values **1**, 0.8, 0.75 and 0.4, which is **1** since there is only one component carrier
- μ is the numerology as defined in TS 38.211, which is **1** for 30 KHz Subcarrier Spacing (SCS)
- T_s^{μ} is the average Orthogonal Frequency Division Multiplexing (OFDM) symbol duration

in a subframe for numerology μ , assuming normal cyclic prefix, which is $T_s^{\mu} = \frac{10^{-3}}{14 \cdot 2^{\mu}} = 3.571 \times 10^{-5} \text{ sec} \approx \mathbf{35 \mu s}$

- $N_{\text{PRB}}^{BW(j),\mu}$ is the maximum number of Physical Resource Blocks (PRB) for selected $BW^{(j)}$ with numerology μ , as defined in 5.3 TS 38.101-1 and 5.3 TS 38.101-2

- 12 is the number of subcarriers for 1 PRB
- $OH^{(j)}$ is the overhead for control channels and takes the following values:
 - 0.14, for frequency range FR1 for DL
 - 0.18, for frequency range FR2 for DL
 - 0.08, for frequency range FR1 for UL
 - 0.10, for frequency range FR2 for UL
- $R_{\max} = 948/1024$. However, for MCS 26, $R = 916.5$

During the placement phase the slice manager determines, based on the experiment referenced NST, the NSs to be deployed per location (core, edge) and reads the Network Service Descriptors (NSDs) and Virtual Network Function Descriptors (VNFDs), referenced by the slice manager NST, registered in the OSM to calculate required resources and determine whether these resources are available. In the case that these resources are available it requests these resources. This is the placement time depicted in the Grafana dashboard.

If the placement process finishes successfully, the Slicing Lifecycle manager requests the activation of the reserved resources. This is the provisioning phase during which it creates required OSM and VIM isolated tenants for the newly created slice. It requests the set up and configuration (VMs and networking, according to NSDs and VNFDs registered in the OSM) of the required resources, within the isolated tenants, which are required for the deployment phase. The deployment time corresponds to the total time needed to have the VMs, which are part of the slice network service, to be up and running, i.e., this is the time needed for the Network Slice Instance (NSI) to have been successfully deployed.

2.3.1.1.1 DL Throughput Results (NCSRD Demokritos site)

For downlink throughput, the experiment begins with the iPerf client, the VM that hosts the 5Genesis iPerf probe (endpoint B), bursting User Datagram Protocol (UDP) traffic towards the UE (endpoint A). On the other hand, the UMA iPerf android application, acting as an iPerf server, is installed on the device capturing all the necessary results. Figure 8 presents the throughput results (i.e., in Mbps) executed for three successive iterations, each of them including 60 samples. It is worth noting that the average values for the three iterations are the following: **median** - 338.83 Mbps, **max** - 355 Mbps, **min** - 306.6 Mbps, **q3** - 343.3 Mbps, **q1** – 333.8 Mbps.

Based on the above equation the theoretical throughput for the corresponding radio configuration described in Annex 6.2 is 386 Mbps. Note that in the above calculation process, block error rate probability is not considered, therefore the result of the experiment is sufficiently close to the theoretical value.

Complementary results in regard to transport layer and the slice manager have been captured and they are also illustrated in Figure 9. With regards to the transport layer includes both UDP jitter in msec and UDP packet loss on the server side, resulting to an average of 0.033 msec and 19.16% respectively. The high packet loss percentage arises from the fact that the UDP bandwidth is set to 400 Mbps, thus the radio channel can handle 338.83 Mbps on average. The values of the slice manager, and more specifically for slicing placement, provisioning and deployment time, are 320 msec, 8.13 sec and 41.8 sec respectively.



Figure 8. DL throughput primary results (NCSRD)



Figure 9. DL throughput complementary results (NCSRD)

2.3.1.1.2 UL Throughput Results (NCSRD Demokritos site)

For uplink throughput, two experiments are defined for two different slot configurations. The first configuration is the same as in the downlink throughput experiment with 2 uplink slots in a period of 5ms. In order to maximize the uplink speed, the second experiment uses 8 uplink slots in a period of 5ms. The details of the radio configurations are described in Annex 6.2.2 6.2.3. The conducted results are presented in Figure 10 and Figure 12 for the first and for the second (best uplink) configurations, respectively. In the uplink direction, the experiments begin with UMA iPerf android application acting as a client, thus the UDP traffic starts from the device (endpoint A) towards the VM (endpoint B). The average values for first configuration involving three iterations are the following: **median** – 43.6 Mbps, **max** – 45.06 Mbps, **min** – 40.7 Mbps, **q3** – 43.9 Mbps, **q1** – 42.88 Mbps. In addition, the average values for the second configuration for best uplink are: **median** – 210.5 Mbps, **max** – 220.3 Mbps, **min** – 192.3 Mbps, **q3** – 214.3Mbps, **q1** – 206 Mbps.

The theoretical values resulting from the above equation are 56 Mbps for the first configuration and 220 Mbps for the best uplink. Moreover, it should be pointed out that in the best uplink case, throughput is improved by 166.9 Mbps on average. UDP jitter and packet loss measurements are also considered in both experiments for uplink. The results are depicted in Figure 11 and Figure 13 respectively and the average values are 0.53 msec jitter, 0% packet loss for the first configuration and 0.11 msec jitter, 0.01% packet loss for the second configuration. Note that, adjusting the UDP bandwidth close to the expected result leads to close to zero,

packet loss values. The values for the slicing placement, provisioning and deployment time are 290 msec, 7.44 sec, 41 sec and 280 msec, 7.25 sec, 30.8 sec, for the first and second configuration respectively.

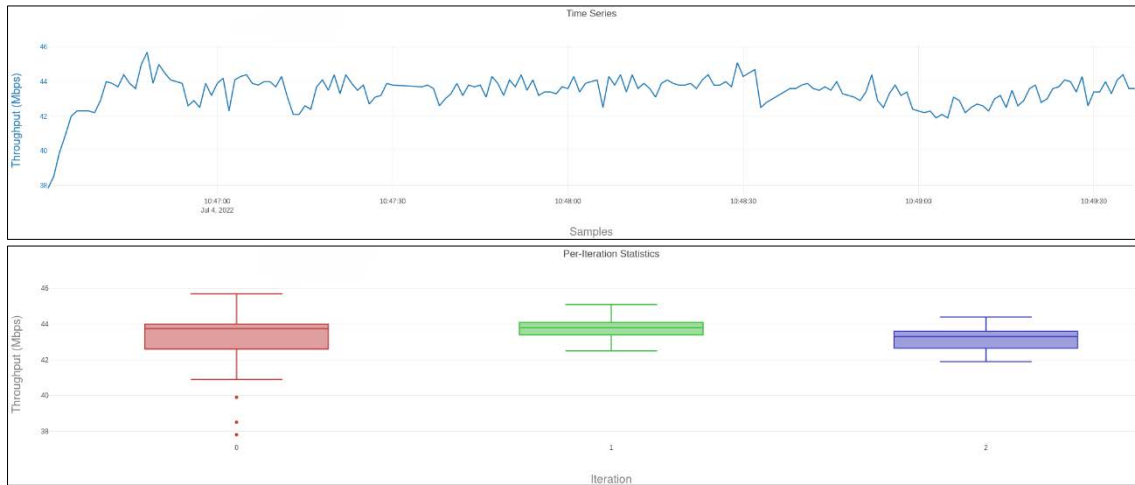


Figure 10. UL throughput primary results (NCSRD)



Figure 11. UL throughput complementary results (NCSRD)

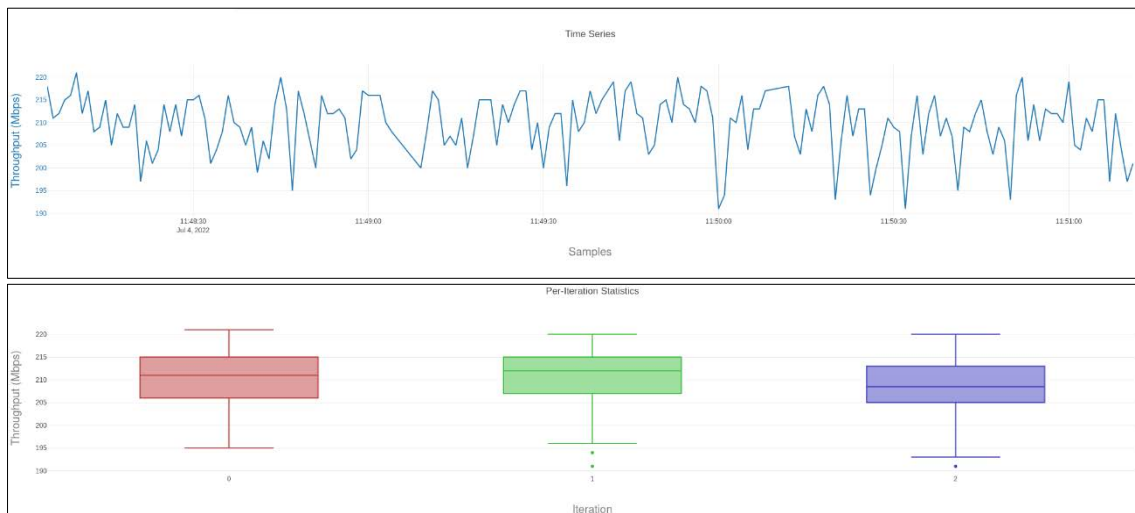


Figure 12. Best UL throughput primary results (NCSRD)



Figure 13. Best UL throughput complementary results (NCSRD)

2.3.1.1.3 DL Throughput Results (Cosmote site)

In accordance with the abovementioned methodology presented in section 2.3.1.1, the throughput experiments were executed in a similar way as in “NCSRD Demokritos” site, considering slight differences on the topology. The details of the experiment are described in the corresponding test case in Annex 6.2.4. For downlink throughput, the experiment begins with the iPerf client, the VM that hosts the 5Genesis iPerf probe (endpoint B), bursting UDP traffic towards the UE (endpoint A) that lays on Cosmote premises together with the RAN. The UMA iPerf android application, acting as an iPerf server, is also installed on the device. Figure 14 presents the throughput results (i.e., in Mbps) executed for three successive iterations, with each of them including 60 samples. It is worth noting that the average values for the three iterations are the following: **median** – 628.7 Mbps, **max** - 635 Mbps, **min** – 578.3 Mbps, **q3** - 631 Mbps, **q1** – 625 Mbps. Packet loss and jitter median values are 14% and 0.017 msec, presented in Figure 15. The throughput results for downlink are adequate, as according to the vendor (NOKIA), for the specific configuration and TDD pattern selected, the maximum values expected are 650 Mbps for downlink for both TCP and UDP traffic. The values for the slicing placement, provisioning and deployment time are 410 msec, 7.24 sec and 30.8 sec respectively.

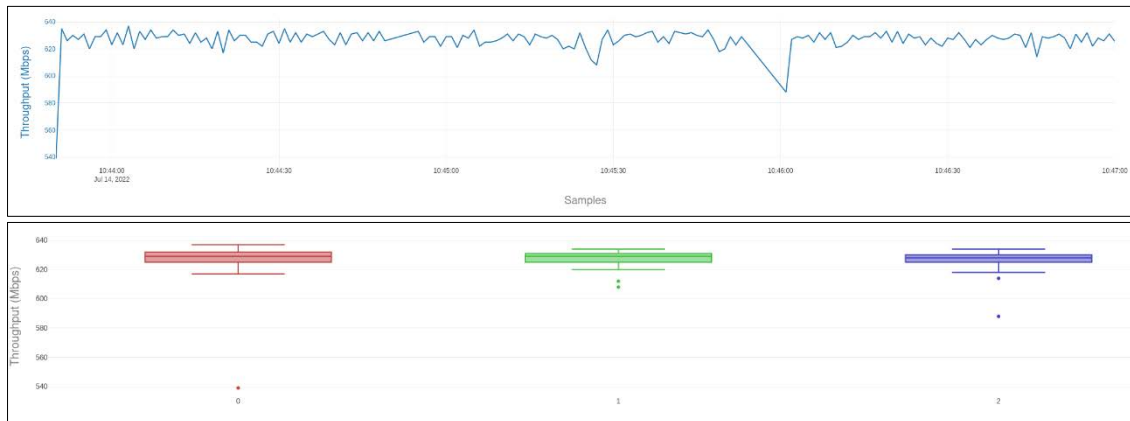


Figure 14. DL throughput primary results (COSMOTE + NCSRD)



Figure 15. DL throughput complementary results (COSMOTE + NCSRD)

2.3.1.1.4 UL Throughput Results (Cosmote site)

The same topology as described in downlink throughput is used also for the uplink throughput results. The details of the experiment are described in Annex 6.2.5. In this case, the experiments begin with UMA iPerf android application acting as a client in Cosmote site, thus the UDP traffic starts from the device (endpoint A) towards the VM (endpoint B) instantiated on OpenStack within Demokritos' premises. The average values involving three iterations, presented in Figure 16, are the following: **median** – 52.7 Mbps, **max** – 52.9 Mbps, **min** – 47.4 Mbps. Complementary measurements are presented in Figure 17 and are 0% packet loss and 0.43 msec jitter on average. The values for the slicing placement, provisioning and deployment time are 440 msec, 7.78 sec and 31.3 sec respectively, as shown in Figure 17

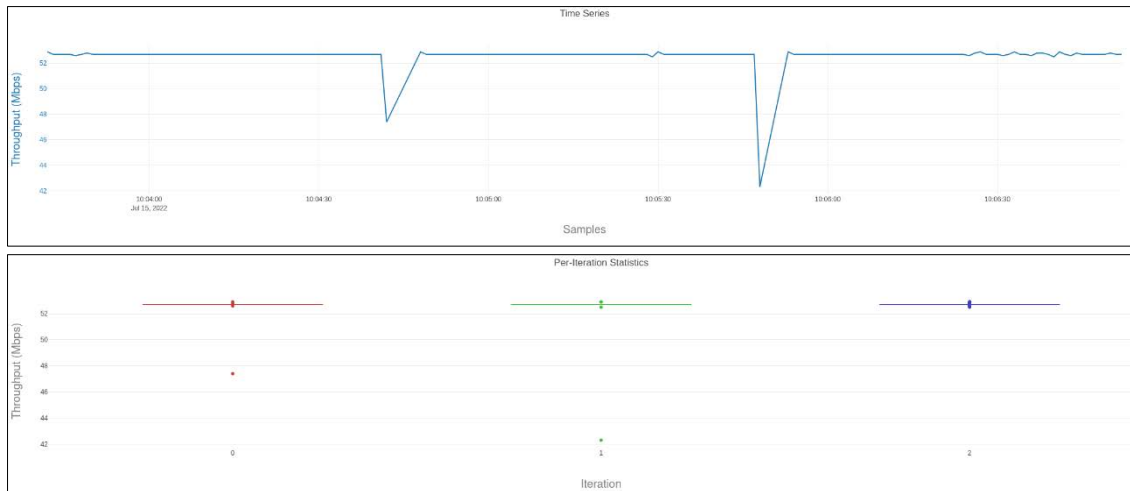


Figure 16. UL throughput primary results (COSMOTE + NCSRD)



Figure 17. UL throughput complementary results (COSMOTE + NCSRD)

2.3.1.2 Round Trip Time (RTT)

2.3.1.2.1 RTT results (NCSRD)

Delay experiments were executed based on test case templates NCSRD_RTT, NCSRD_RTT_low_latency evaluating the RTT between a UE and the VM deployed on OpenStack. More details are described on Annex 6.2.6 6.2.7 including the scenario, the testing infrastructure, the target KPI and the test case sequence implying the execution steps. As in throughput experiments, the SUT involves the commercial Amarisoft Classic (i.e., both 5G-NR and 5GC Rel. 16), one COTS UE and a VM that integrates the 5Genesis ping probe. All the tests were conducted in a lab environment with perfect channel conditions leading to an approximate 26 MCS) value and a packet size of 64 byte.

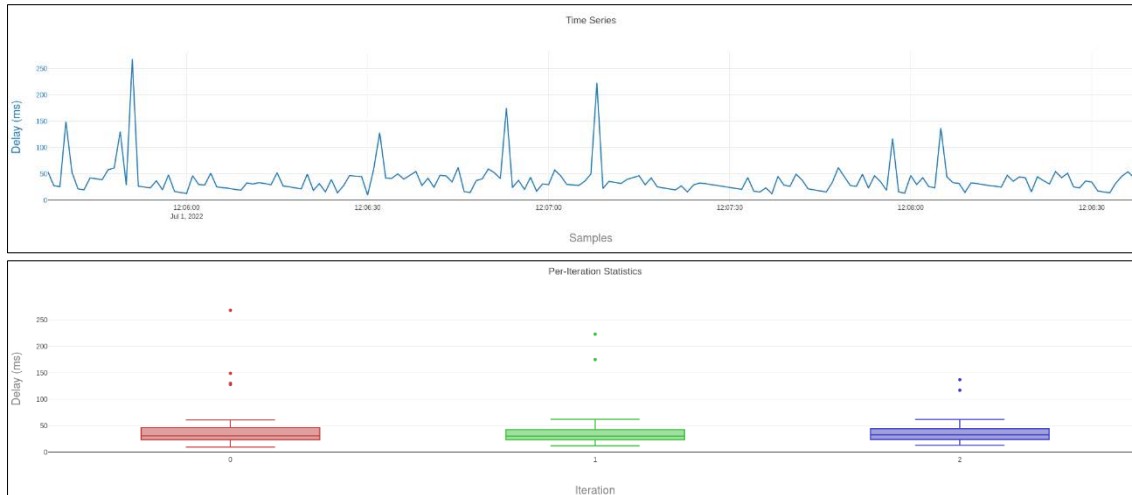


Figure 18. RTT - 64byte packet size (NCSRD)

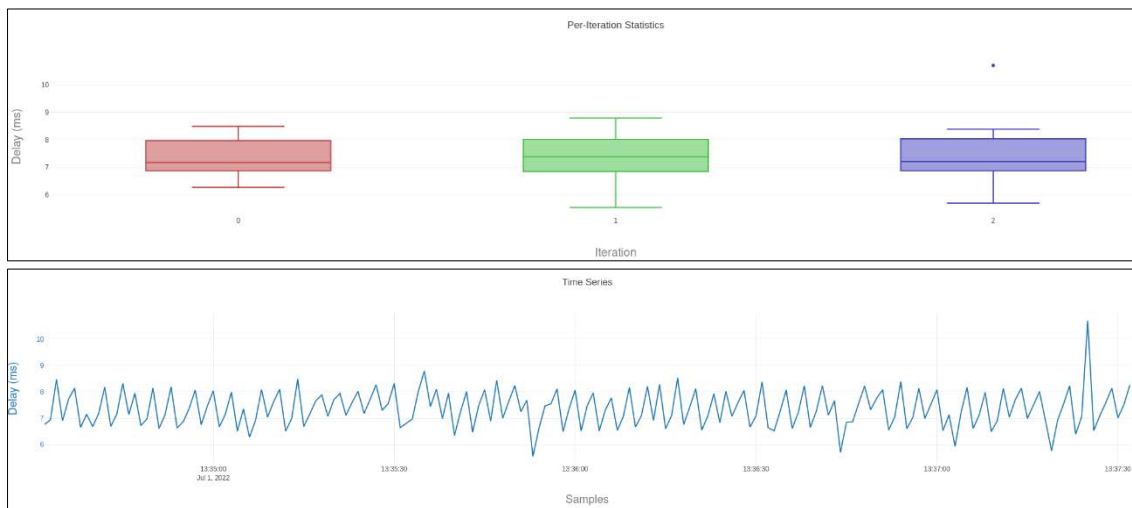


Figure 19. RTT – low latency - 64byte packet size (NCSRD)

As it can be seen from Figure 18, the average values for three iterations are: max – 209 msec, q3 – 44.58, median – 31.45, q1 – 23.85, min – 11.76. For the low latency experiment the scheduling request period has been reduced from 10 to 0.5 msec and symmetric slot configuration is used. The conducted results are illustrated in Figure 19 and the average values are: max – 9.32 msec, q3 – 8.01, median – 7.26, q1 – 6.87, min – 5.84. It is worth noting that for the low latency configuration the median value is reduced by 24.10 msec.

2.3.1.2.2 RTT results (Cosmote)

Delay experiments are executed based on the topology presented in Figure 20. The details of the experiment are well described in Annex 6.2.8. The average values for three iterations including 60 samples are: max – 83.36 msec, q3 – 17.73 msec, media - 14.4 msec, q1 – 13 msec, min – 11.3 msec.

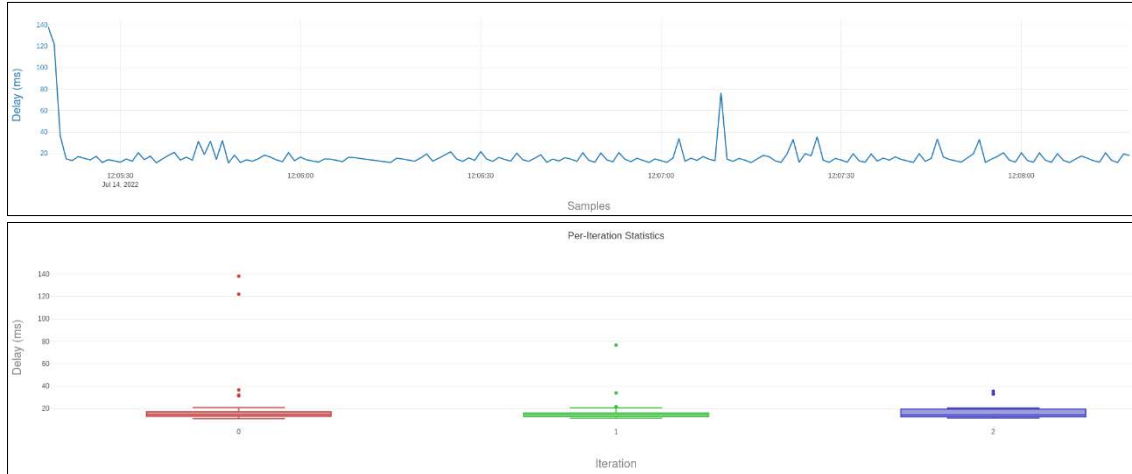


Figure 20. RTT - 64byte packet size (COSMOTE + NCSRD)

2.3.2 Málaga Platform Tests and Results

In this section results of the evaluation of the 5G SA FR1 deployment (see Figure 21) available at the UMA testbed are provided. This deployment includes 4 5GNR TDD micro cells in FR1 band n78 at 3.5GHz.

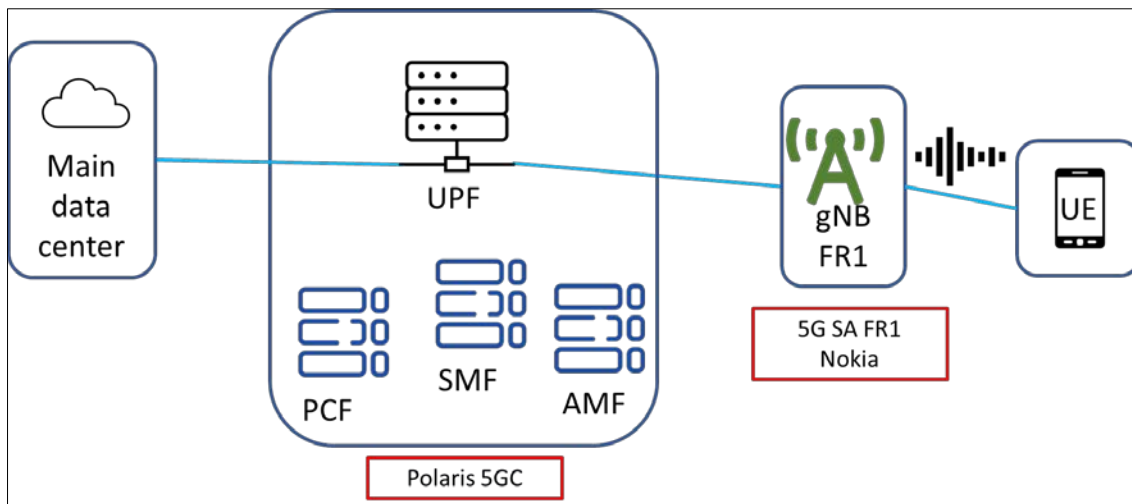


Figure 21. UMA Radio deployment

Table 1 summarizes the configuration applied in the network during the assessment. Each one of the cells have an associated channel bandwidth of 50 MHz. In the downlink, a 4x4 MIMO with 256QAM modulation enables the selected scheduling configuration to reach 728 Mbps per carrier.

Table 1. 5G SA Configuration at UMA testbed

Band	N78
Mode	TDD

Bandwidth	50 MHz
Carrier-components	1 Carrier
MIMO-layer	4 layers
DL MIMO mode	4x4
Beams	Single beam
Subcarrier-spacing	30 kHz
Uplink/Downlink slot ratio	1/4

2.3.2.1 DL Throughput Results

The throughput tests have been performed between the main compute node of the testbed and a 5G UE based on the UMA iPerf agents. The iPerf server is deployed at the UE and the iPerf client is running at the main compute node. The execution of the tests has been automated via the Open5GENESIS Suite and OpenTAP.

The iPerf client generates a Transmission Control Protocol (TCP) traffic pattern with two TCP flows. The TCP window has been configured at both side with a value of 15 Mbits. A total of 25 iterations have been executed where the duration of each iteration is of 180 seconds. The results of each one of the iterations are depicted in Figure 22.

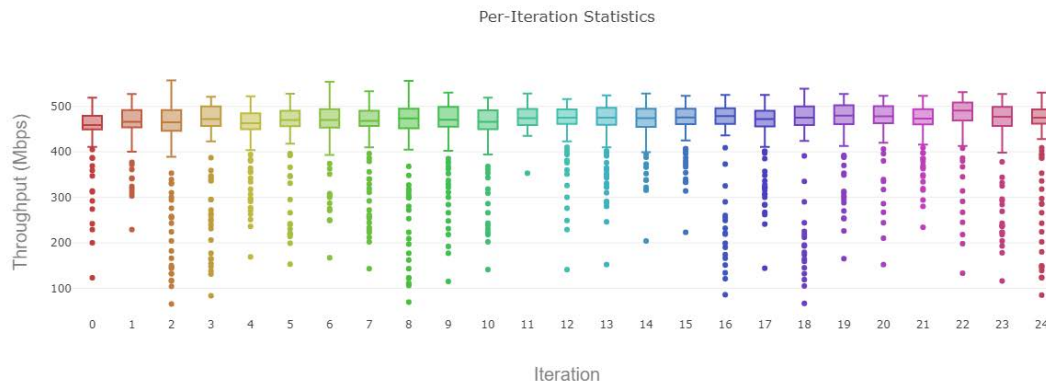


Figure 22 Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM throughput per iteration (UMA)

The temporal evolution of the throughput is presented in Figure 23.

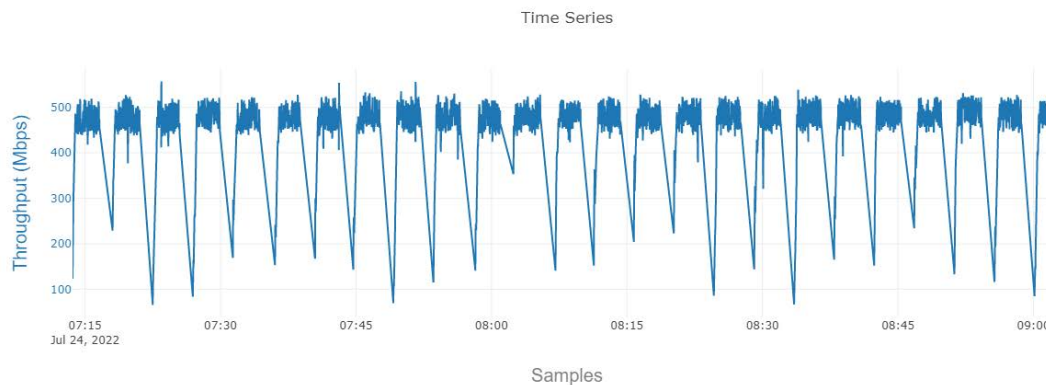


Figure 23 Micro cells 5G SA MIMO 4x4 256 QAM throughput temporal evolution (UMA)

The statistical analysis of the results is provided in Table 2.

Table 2 Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM throughput statistical analysis (UMA)

Mean: 460.092 +/- 3.775 Mbps
Standard deviation: 63.426 +/- 7.783 Mbps
Median: 472.600 +/- 2.620 Mbps
Min: 146.832 +/- 26.897 Mbps
Max: 530.080 +/- 4.725 Mbps
25% Percentile: 457.15 +/- 2.102 Mbps
75% Percentile: 494.61 +/- 2.393 Mbps
5% Percentile: 300.416 +/- 31.958 Mbps
95% Percentile: 515.052 +/- 1.542 Mbps

The mean throughput obtained is 460 Mbps, however, the theoretical throughput for the configuration applied and described in Table 1 is 700 Mbps. The reason why the observed downlink throughput is less than expected is that the rank indicator reported by the UE is 3 for half the time, which means 4X4 MIMO is not performing with the best efficiency.

The analysis of the results reveals a quite stable behavior. In order to contextualize these results radio measurements were also collected and are reported in Table 3. The radio measurements show good radio propagations conditions.

Table 3 Complementary radio measurements for throughput test (UMA)

NR RSRP max	-43.6 dBm
NR RSRP min	-46.4 dBm
NR RSRP avg	-45.6 dBm
NR RSRQ max	-10.2 dBm
NR RSRQ min	-10.5 dBm
NR RSRQ avg	-10.3 dBm

2.3.2.2 Round Trip Time Results

The gNodeBs have activated a feature called proactive scheduling that enables the scheduler to generate a configurable number of additional uplink grants and thus, avoid the latency associated with the scheduling request procedure.

The tests are based on the UMA Ping agent and have been performed between the main compute node and the 5G UE. The tests have been automated via the Open5GENESIS Suite. The configured Internet Control Message Protocol (ICMP) packet size is of 56 bytes. A total of 25 iterations with a duration of 180 seconds each have been executed and the results are shown in Figure 24.

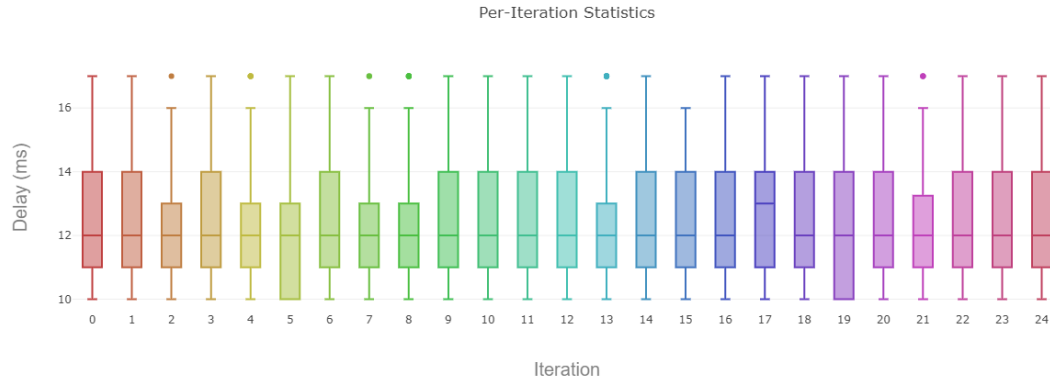


Figure 24 Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT per iteration (UMA)

The temporal evolution of the RTT measurements is shown in Figure 25.

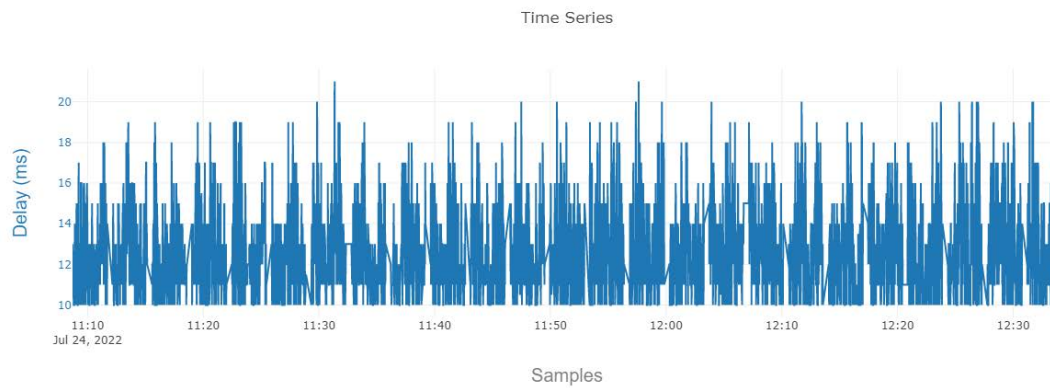


Figure 25 2221Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT temporal evolution (UMA)

The statistical analysis of the results is provided in Table 4.

Table 4 Micro cells 5G SA MIMO 4x4 50 MHz 256 QAM RTT statistical analysis (UMA)

Mean: 13.713+/- 0.284 ms
Standard deviation: 9.054 +/- 2.381 ms
Median: 12.546000 +/- 0.115363 ms
Min: 10.008000 +/- 0.011429 ms
Max: 19.752000 +/- 0.291422 ms
25% Percentile: 11.324 +/- 0.094 ms
75% Percentile: 14.344 +/- 0.173 ms
5% Percentile: 10.2824 +/- 0.048 ms
95% Percentile: 17.0884 +/- 0.279ms

The analysis of the measurements also shows a stable behavior of the infrastructure regarding the delay. The radio conditions (see Table 5) are very similar to the one reported during the throughput tests. The mean value obtained can supports the cycle time demanded by the process automation industrial use case as stated in [18].

Table 5 Complementary radio measurements for RTT tests (UMA)

NR RSRP max	-43.6 dBm
NR RSRP min	-46.4 dBm
NR RSRP avg	-45.6 dBm
NR RSRQ max	-10.2 dBm
NR RSRQ min	-10.5 dBm
NR RSRQ avg	-10.3 dBm

Finally, the 5G Core was also monitored during the tests to check its correct performance. Figure 26 and Figure 27 illustrate some of the relevant monitoring parameters during tests.

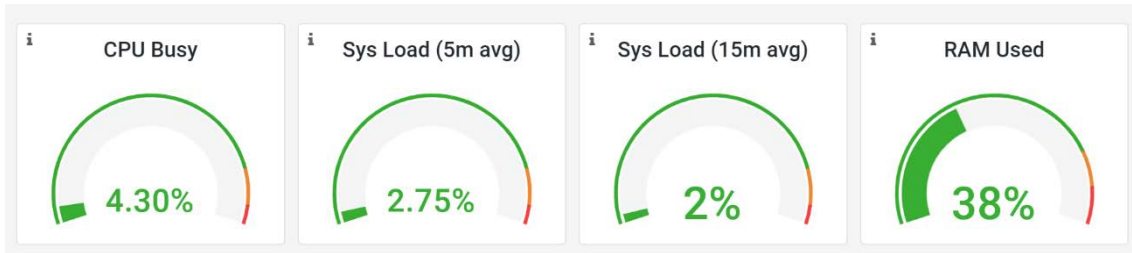


Figure 26 5GCore usage statistics part 1 (UMA)

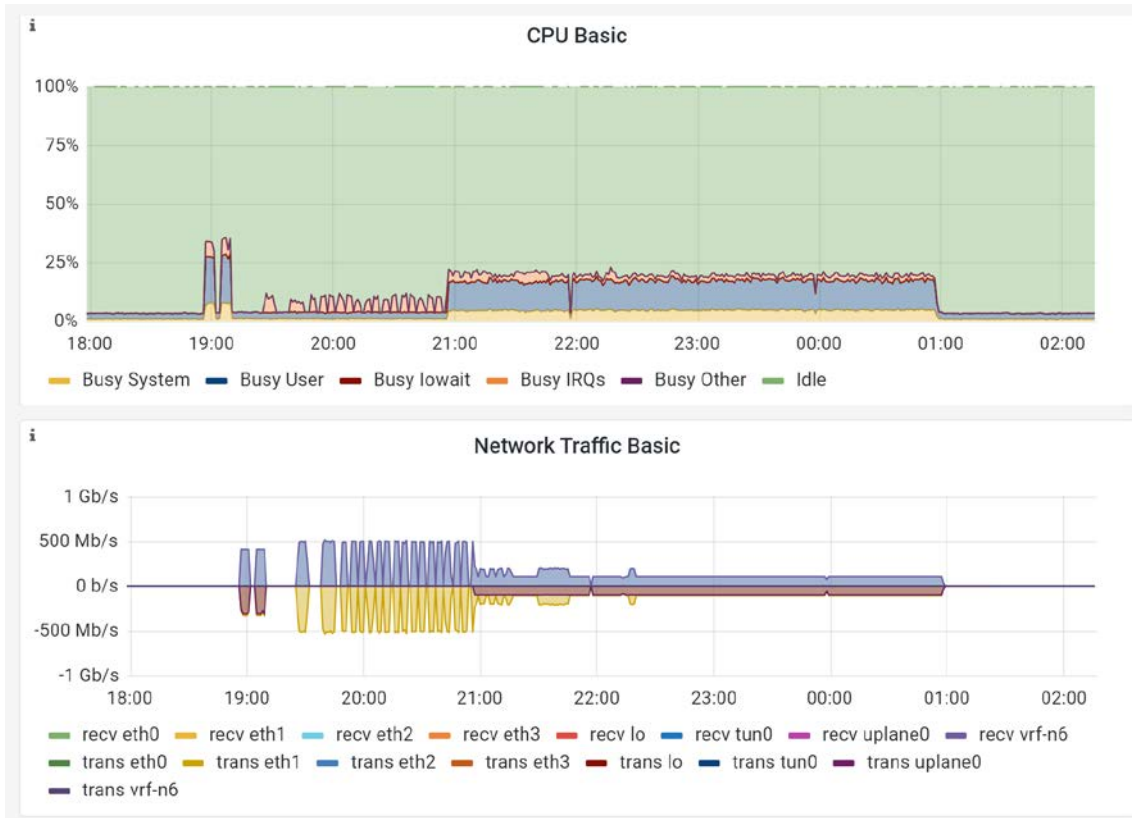


Figure 27 5GCore usage statistics part 2 (UMA)

2.3.2.3 TSN over 5G results

In this section initial stage of results of TSN over 5G using the current components in the Málaga platform and previously mentioned in in Section 3.1.2.2 are presented. Note that these results are far from the results obtained in wired TSN networks, however, they will be improved with

the ongoing development of the different components, such as the TSN translators and the time synchronization.

The test consists of sending UDP (including 802.1Q headers within TSN domain) traffic from one TSN endpoint to other TSN endpoint with the 5G network acting as a TSN bridge. Focused is made on measuring the delay and jitter of the traffic, since these KPIs directly affect the perceived Quality of Service and reliability of the communication.

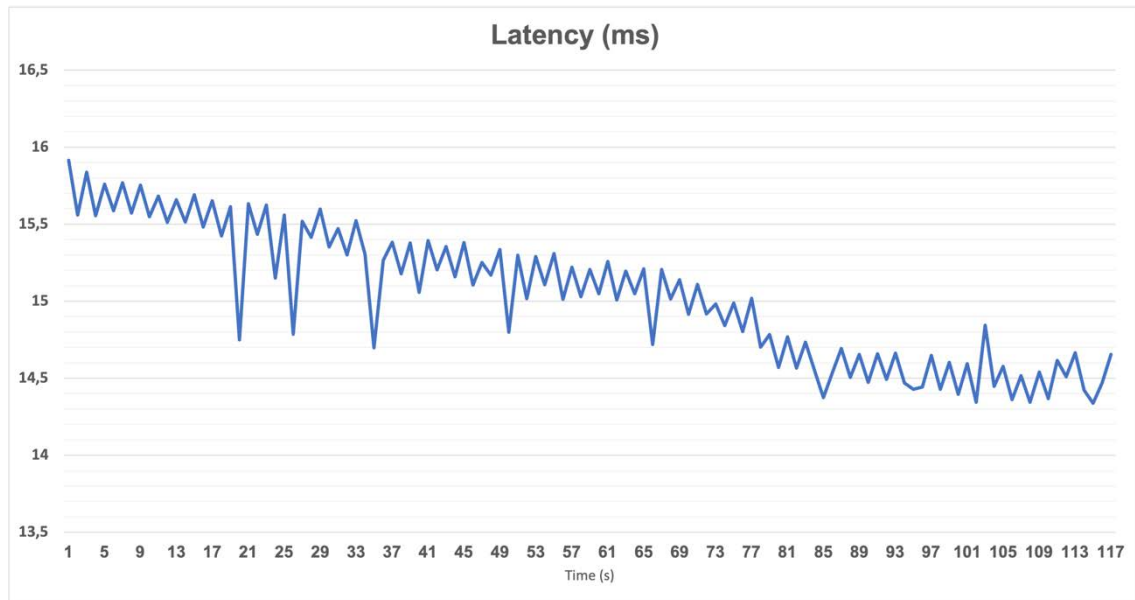


Figure 28. TSN over 5G - delay of the test

Figure 28 shows the latency (in ms) during the test, the results start at around 16 ms and they converge at ~80s around 14,5 ms. Figure 29 depicts the jitter in the same test, in this case, results are between 9-10 ms.

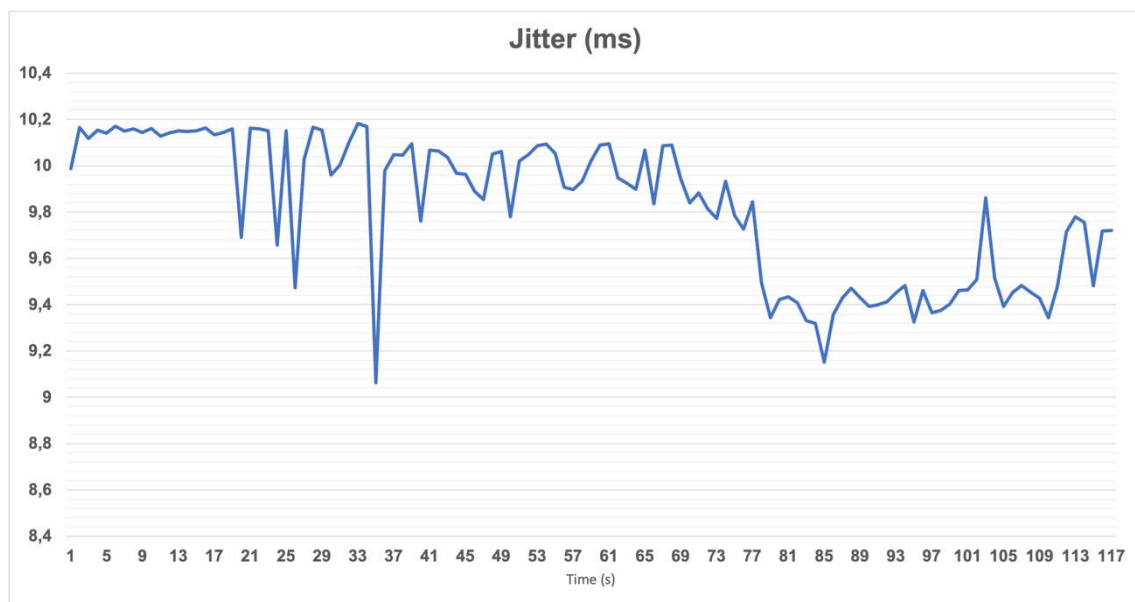


Figure 29. TSN over 5G - jitter of the test

3 COMPONENT-LEVEL EVALUATION

3.1 EVOLVED-5G SOFTWARE COMPONENTS

The NEF emulator exposes Northbound APIs to NetApps following the 3GPP TS 29.522 [4] and TS 29.122 [5] specifications. At the time of writing this deliverable, the current version of the NEF emulator supports the *MonitoringEvent* and *AsSessionWithQoS* APIs. The details of the architectural concerns, implementation aspects and the technologies used to develop the NEF emulator can be found on D3.1 [7]. In addition, a thorough description NEF APIs, including the two APIs that are currently supported can be found on D4.1 [9]. In order to ensure the proper functionality of the APIs, various test cases have been defined, developed and executed, as described in Section 3.2 (Functional Testing). On top of that, performance tests are also considered including the access time and the success rate of multiple requests. The details of the definition and the results of these sets of tests are described in section 3.3 (Performance Testing)

EVOLVED-5G is developing a CAPIF Core Function following 3GPP TS 23.222 [4] and TS 29.222 [5]. This module is described in deliverables D2.4 [6] and D3.2 [8]. CAPIF Core Function tool has been incorporated to Athens and Málaga platforms as the API Exposure Layer to expose platform APIs, namely, NEF Emulator APIs.

CAPIF offers API management services for API Invokers, in our case, the NetApps. The main services offered are:

- API Invoker registration services, which allows NetApps to register in CAPIF Core Function to consume CAPIF services.
- API Publish service, which allows API Exposure services, in our case, NEF Emulator, to publish their APIs for NetApps to discover them.
- API Discovery Service, that allows NetApps to Discover APIs registered in CAPIF, such as NEF Emulator APIs.

CAPIF Core Function has been developed following 3GPP CAPIF APIs YAMLS published in a GitHub repository [11] where all Release 17 3GPP APIs are published. In order to guarantee that API contracts and CAPIF Core Function functionality works properly, a number of tests cases have been developed and automated, so that in every deployment of CAPIF Core Function during Validation or Certification processes, behavior and compliance with standards of CAPIF Core Function is guaranteed.

3.2 FUNCTIONAL TESTING

3.2.1 NEF Emulator results

The NEF emulator constitutes the backend entity that exposes the necessary NEF APIs for facilitating the development of a NetApp without the requirement of NEF exposure availability through a real 5G network. For the NEF emulator that has been developed in the scope of the project, a testing plan has also been introduced (https://github.com/EVOLVED-5G/NEF_emulator/tree/main/docs/test_plan).

The testing plan targets the *MonitoringEvent* and *AsSessionWithQoS* APIs, which are exposed by NEF and exploited by the EVOLVED-5G NetApps. The list of tests defined is presented in the following table:

Table 6 Testing plan targeting the MonitoringEvent and AsSessionWithQoS APIs

TEST	Entity	NEF API
Create subscription by Authorized NetApp	NEF_API_MONITORING_EVENT_API	201 NetApp creates a subscription successfully to the Monitoring Event API for a registered UE.
One-time request to the Monitoring Event API by Authorized NetApp	NEF_API_MONITORING_EVENT_API	200 NetApp sends a one-time response request to the Monitoring Event API for a registered UE.
Create subscription when there is already an active subscription for a registered UE	NEF_API_MONITORING_EVENT_API	409 Conflict / There is already an active subscription for UE with external id 'externalId'.
Create subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT_API	401 Unauthorized.
Read all active subscriptions by Authorized NetApp	NEF_API_MONITORING_EVENT_API	200 With a list of active subscriptions from the Monitoring Event API.
Read all active subscriptions by Authorized NetApp (no active subscriptions available)	NEF_API_MONITORING_EVENT_API	204 No Content.
Read individual subscription by Authorized NetApp	NEF_API_MONITORING_EVENT_API	200 Individual subscription by the NetApp from Monitoring Event API is successfully retrieved.
Read individual subscription by Authorized NetApp with invalid subscription id	NEF_API_MONITORING_EVENT_API	404 Not Found.
Read all active subscriptions by unAuthorized NetApp	NEF_API_MONITORING_EVENT_API	401 Unauthorized
Read individual subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT_API	401 Unauthorized
Update individual subscription by Authorized NetApp	NEF_API_MONITORING_EVENT_API	200 Individual subscription by the NetApp from Monitoring Event API is successfully updated.
Update individual subscription by Authorized NetApp with invalid subscription id	NEF_API_MONITORING_EVENT_API	404 Not Found

Update individual subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT_API	401 Unauthorized
Delete individual subscription by Authorized NetApp	NEF_API_MONITORING_EVENT_API	200 Individual subscription by the NetApp from Monitoring Event API is successfully deleted.
Delete individual subscription by Authorized NetApp with invalid subscription id	NEF_API_MONITORING_EVENT_API	404 Not Found
Delete individual subscription by unAuthorized NetApp	NEF_API_MONITORING_EVENT_API	401 Unauthorized
Create subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	201 The NetApp created a subscription successfully to the AsSessionWithQoS for a registered UE.
Create subscription when there is already an active subscription for a registered UE	NEF_API_AS_SESSION_WITH_QOS_API	409 Conflict / There is already an active subscription for UE with (ipv4, ipv6, mac address)
Create subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	401 Unauthorized
Read all active subscriptions by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	200 with subscriptions retrieved successfully by the NetApp from the AsSessionWithQoS API
Read all active subscriptions by Authorized NetApp (no active subscriptions available)	NEF_API_AS_SESSION_WITH_QOS_API	404 Not Found
Read individual subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	200 Individual subscription by the NetApp from AsSessionWithQoS API is successfully retrieved
Read individual subscription by Authorized NetApp with invalid subscription id	NEF_API_AS_SESSION_WITH_QOS_API	404 Not Found
Read all active subscriptions by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	401 Unauthorized
Read individual subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	401 Unauthorized

Update individual subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	200 Individual subscription by the NetApp from AsSessionWithQoS API is successfully updated
Update individual subscription by Authorized NetApp with invalid subscription id	NEF_API_AS_SESSION_WITH_QOS_API	404 Not Found
Update individual subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	401 Unauthorized
Delete individual subscription by Authorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	200 Individual subscription by the NetApp from AsSessionWithQoS API is successfully deleted
Delete individual subscription by Authorized NetApp with invalid subscription id	NEF_API_AS_SESSION_WITH_QOS_API	404 Not Found
Delete individual subscription by unAuthorized NetApp	NEF_API_AS_SESSION_WITH_QOS_API	401 Unauthorized

The functional tests of NEF have been implemented taking advantage of the Robot Framework [10]. They are available in the GitHub repository of NEF under the branch “validation-tests” (https://github.com/EVOLVED-5G/NEF_emulator/tree/validation-tests).

In this branch of the repository, three folders exist, each one related to some aspect of NEF testing:

- **Pipelines/**: This folder contains the Jenkins pipeline for automatically deploying NEF services if necessary (i.e., if it is not already deployed) and for running the Robot Framework tests.
- **Tests/**: This folder contains the actual tests, including the code of the tests, the test cases, the relevant resources and custom Python libraries. In particular:
 - Features:
 - test cases for each service.
 - robot framework code for the tests referencing also relevant libraries and resources from the respective directories.
 - Libraries:
 - auxiliary code for testing to cover Robot Framework functionalities.
 - Resources:
 - configuration parameters, mainly keywords and variables referenced by the tests.
- **Tools/**: This folder contains the code for containerizing the tests, i.e., code and configuration parameters to generate the Robot Docker image (to be used by Jenkins pipelines) and for deploying a Jenkins pipeline that uploads the Robot Docker image to the JFrog Artifactory of EVOLVED-5G.

The test cases are categorized by the specific API of NEF, which is denoted by the relevant path that follows the /features/ directory (i.e., tests/features/<API_NAME>). Inside that directory a file named /<API_NAME>.robot contains the code for testing each endpoint of the respective API.

The generation of a robot image is realized through the execution of the following command in the path /tools/robot:

```
docker build --no-cache -t ${ROBOT_IMAGE_NAME}:${ROBOT_VERSION} .
```

This builds a Docker image with the name and version provided. An example follows:

```
docker build --no-cache -t dockerhub.hi.inet/dummy-netapp-testing/robot-test-image:2.0 .
```

If Jenkins is desired to be used, the image is built and then it gets pushed to dockerhub.hi.inet through the pipeline defined in the directory /tools/robot too, which is named robot-image.groovy.

Once the image is built, Robot Framework tests can be executed either locally or remotely via Jenkins. The first option is used during the development to test NEF faster by deploying the Docker image at the system where the development takes place. The second option takes advantage of the EVOLVED-5G CI/CD platform and can be configured to use a deployment of NEF wherever it is available. The deployment of the Docker image is similar with a differentiation in launching the respective container in the second case, as illustrated below:

The Docker command consists of the following:

- “docker run -t --network=“host” --rm”: this will call Docker and set some useful options:
 - Run: launch Docker image on Docker.
 - -t: use tty
 - -name: gives the name to the container created.
 - --network: this means Docker image will use the same networks than host.
 - --rm: this option will remove image after end execution of tests from Docker environment. Reduces resource usage.
- “-v \${ROBOT_<DIR>_DIRECTORY}:/opt/robot-tests/<DIR>”: these options will attach local directories to volumes defined under robot Docker image. It has only 3 volumes:
 - /opt/robot-tests/tests: At this volume Docker image will expect to attach tests directory of repository, including all robot code.
 - /opt/robot-tests/results/AsSessionWithQoSAPI: At this volume robot will store reports generated after execution of this entity’s tests. If reports need to be stored after execution, a folder on host and pass this folder as argument in the call when running tests (--outputdir /opt/robot-tests/results/AsSessionWithQoSAPI) must be provided. Otherwise, those reports will be missed. If the folder is not included in the call reports will be stored in the folder where the code of tests is stored.
 - /opt/robot-tests/results/MonitoringEventAPI: same as the above but for different entity.
- “\${ROBOT_IMAGE_NAME}:\${ROBOT_VERSION}”: This part indicates the Docker image and version previously defined in the Docker image generation. In the example, the Docker is pulled from the EVOLVED-5G artifactory.
- Options after Robot Image selection:
 - -c: This allows the execution of commands inside the container created.

After running all the tests, the following report is produced:

```
+ docker run -t --name robot --network-host --rm -v /home/contint/pro-dcip-evols-01/workspace/nef_emulator_validation/nef_emulator_validation_testing/tests:/opt/robot-tests/tests -v /home/contint/pro-dcip-evols-01/workspace/nef_emulator_validation_testing/results:/opt/robot-tests/results/ASessionWithQoSAPI -v /home/contint/pro-dcip-evols-01/workspace/nef_emulator_validation_testing/results/MonitoringEventAPI:/opt/robot-tests/results/MonitoringEventAPI -e NGINX_HOSTNAME=http://localhost:8080 dockerhub.hi.inet/dummy-netapp-testing/robot-test-image:2.0 /bin/bash -c robot --outputdir /opt/robot-tests/results/ASessionWithQoSAPI /opt/robot-tests/tests/features/NEF_Monitoring_Event_API/monitoring_event_api.robot
robot --outputdir /opt/robot-tests/results/MonitoringEventAPI /opt/robot-tests/tests/features/NEF_Monitoring_Event_API/monitoring_event_api.robot

Nef Subscriptions Api :: This file contains the Test Cases for the As Sessi...

Create Nef subscription                                Response body: {'ipv4Addr': '10.0.0.3', 'notificationDestination': 'http://localhost:80/api/v1/utis/session-with-qos/callback', 'snsal': {'sst': 1, 'sd': '000001'}, 'dnn': 'province1.mnc01.mcc202.gprs', 'qosReference': 9, 'altQoSReferences': [0], 'usageThreshold': {'duration': 0, 'totalVolume': 0, 'downloadVolume': 0, 'uplinkVolume': 0}, 'qosMonInfo': {'reqQosMonParams': ['DOWNLINK'], 'repFreqs': ['EVENT_TRIGGERED'], 'latThreshDL': 0, 'latThreshUL': 0, 'latThreshRp': 0, 'waitTime': 0, 'repPeriod': 0}, 'ipv4Addr': '::1', 'macAddr': '22-00-00-00-00-01', 'link': 'http://localhost:8080/net/api/v1/3gpp-as-session-with-qos/v1/3/subscriptions/624aed17cated5db2c221db'}
Create Nef subscription                                | PASS |

Create Nef subscription with already active            ....There is already an active subscription for UE
Create Nef subscription with already active            | PASS |

Create subscription by unauthorized NetApp              ....Unauthorized NetApp
Create subscription by unauthorized NetApp              | PASS |
```

Figure 30. NEF Emulator Results – Test Report

3.2.2 CAPIF Tool results

As described in the introduction, along with CAPIF Core Function, EVOLVED-5G has developed a number of test cases to validate that after each deployment of CAPIF Core Function tool, API contracts are following 3GPP specifications (TS 23.222 [4] and TS 29.222 [5]) and that CAPIF Core Function behaves properly. These tests simulate *API Invoker* and *API Publisher* entities and invoke CAPIF APIs to test several conditions, checking that the response provided by CAPIF Core Function is the appropriate one. The defined tests are summarized in the following table:

Table 7 Testing plan targeting the API Invoker and API Publisher of CAPIF

TEST	Entity	CAPIF API
Register NetApp	CAPIF_API_Invoker_Management_API	201 API invoker on-boarded successfully.
Register NetApp Already registered	CAPIF_API_Invoker_Management_API	403 Forbidden
Update Registered NetApp	CAPIF_API_Invoker_Management_API	200 API invoker details updated successfully.
Update Not Registered NetApp	CAPIF_API_Invoker_Management_API	404 Not found.
Delete Registered NetApp	CAPIF_API_Invoker_Management_API	204 The individual API Invoker matching onboardingId was offboarded.
Delete Not Registered NetApp	CAPIF_API_Invoker_Management_API	404 Not Found.
Publish API by Authorised API Publisher	CAPIF_Publish_Service_API	201 API Published
Publish API by NON Authorised API Publisher	CAPIF_Publish_Service_API	401 Unauthorised
Retrieve all APIs Published by Authorised apfld	CAPIF_Publish_Service_API	200 Definition of all service API(s) published by the API publishing function.
Retrieve all APIs Published by NON Authorised apfld	CAPIF_Publish_Service_API	401 Unauthorized
Retrieve single APIs Published by Authorised apfld	CAPIF_Publish_Service_API	200 Definition of serviceApild service API published by the API publishing function.

Retrieve single APIs non Published by Authorised apfld	CAPIF_Publish_Service_API	404 Not Found
Retrieve single APIs Published by NON Authorised apfld	CAPIF_Publish_Service_API	401 Unauthorized
Update API Published by Authorised apfld with valid serviceApild	CAPIF_Publish_Service_API	200 Definition of service API updated successfully.
Update APIs Published by Authorised apfld with invalid serviceApild	CAPIF_Publish_Service_API	404 Not Found
Update APIs Published by NON Authorised apfld	CAPIF_Publish_Service_API	401 Unauthorized
Delete API Published by Authorised apfld with valid serviceApild	CAPIF_Publish_Service_API	204 The individual published service API matching the serviceAPild is deleted.
Delete APIs Published by Authorised apfld with invalid serviceApild	CAPIF_Publish_Service_API	404 Not Found
Delete APIs Published by NON Authorised apfld	CAPIF_Publish_Service_API	401 Unauthorized
Discover Published service APIs by Authorised API Invoker	CAPIF_Discover_Service_API	200 With Collection of Service API Descriptions
Discover Published service APIs by Non Authorised API Invoker	CAPIF_Discover_Service_API	401 Unauthorized
Discover Published service APIs by not registered API Invoker	CAPIF_Discover_Service_API	403 Forbidden
Discover Published service APIs by registered API Invoker with 1 result filtered	CAPIF_Discover_Service_API	200 Ok with 1 api returned
Discover Published service APIs by registered API Invoker filtered with no match	CAPIF_Discover_Service_API	200 Ok with empty list returned
Discover Published service APIs by registered API Invoker not filtered	CAPIF_Discover_Service_API	200 Ok with 2 api returned

All these tests have been implemented using Robot Framework tool [10] and are available at the EVOLVED-5G Github repository: https://github.com/EVOLVED-5G/CAPIF_API_Services, where different folders are created, one related to some aspect of CAPIF. These folders are:

- Docs: Here all the documentation related to the Test Plan definition created for CAPIF is stored.
- Iac: This folder contains all needed information to deploy infrastructure of services at OpenShift, in this case, Terraform scripts.
- Pac: It contains pipelines to be used by Jenkins for any operation, like deploy/destroy at OpenShift, generation of Docker images for testing or launch test.
- Services: All services involved at CAPIF deployment, including auxiliary services like jwt, nginx, easyrsa server, etc.
- Tests: The Robot code for testing is under this folder, where Test Cases and all related code developed (like Python custom libraries and resources) are stored.
- Tools: This folder contains information to generate Robot Docker image (to be used mainly by Jenkins pipelines) and also script to generate from Swagger the initial template of CAPIF services.

The Tests folder contains all developed code of robot to execute Test Plan defined. Under tests folder a directory structure to split in a logical way all code needed is presented:

- Features:
 - Here are the Test Cases for each service.
 - Each folder (including root one) include a `__init__.robot` that setup configuration for all directories contained below it.
 - The code used is Robot flavor.
 - This code will use also code inside Libraries and Resources
- Libraries:
 - At this folder Python is used as an auxiliary code for testing.
 - This is a usual way to develop code needed for testing that need complex logic where Robot code is discouraged, because High level syntax only increase the complexity, for example, get an object from dictionary.
- Resources:
 - All auxiliary code developed using Robot, mainly Keywords and Variables for all Test Cases implementations.

Test Cases are split based on each component of the CAPIF. Each one will be stored under `tests/features/<COMPONENT_NAME>` folder at repository and will have 2 files:

- `__init__.robot` : This file contains all needed Settings for that specific component, for example, Force Tags that will setup the tag for all test of that component.



```
CAPIF_API_Services_Evolved / tests / features / CAPIF_Api_Evolved
Load In Interactive Console | Jorge Moratinos, 5 months ago | 1 author
1 *** Settings ***           Jorge Moratinos, 5 months ago
2 Force Tags                 capif_api_events
```

Figure 31. `__init__.robot` file

- `<component_name>.robot` : This file contains all code for each Test Case of that component, setting up the tags for each test.



Figure 32. `<component_name>.robot` file

The `__init__.robot` file Force Tags simplifies the way tests are launched. Indeed, when launching all tests of one CAPIF component, the addition of those tags is the only necessary requirement.

For the generation of robot image, just the execution of the following command under repository folder `/tools/robot` (where Dockerfile is stored) is needed:

```
docker build --no-cache . -t ${ROBOT_IMAGE_NAME}:${ROBOT_VERSION}
```

This will build a Docker image with name and version indicated. For example, the command could be something like:

```
docker build --no-cache . -t dockerhub.hi.inet/5ghacking/5gnow-robot-test-image:3.0
```

To use Jenkins, the usual way is to build that and push this new image to `dockerhub.hi.inet`. The pipeline that manages that process is presented at the repository under `/pac/Jenkinsfile-tools.groovy` (https://github.com/EVOLVED-5G/CAPIF_API_Services/blob/develop/pac/Jenkinsfile-tools.groovy)

Once the image is built, there are 2 ways to execute Robot Tests:

- Local machine: This is useful during development, when a quick way of testing (either on local or remote environments) with CAPIF is enabled by using a local Docker image built previously.
 - This is allowed by building a local Docker image and launching it with the needed input parameters. (This is better for local development).
 - Alternatively, the image uploaded to `dockerhub.hi.inet` can also be downloaded.
- Jenkins: This is useful to raise up a complete ci/cd environment enabling the deployment of CAPIF services at OpenShift and launching tests on that deployment pipeline.
 - To allow this, the robot Docker image must be uploaded to Dockerhub, usually executing the tool build pipeline.

The way to invoke Docker image is the same, but in Jenkins a Groove pipeline is used instead. However, the step to launch tests using Docker image is the same:

```
docker run -t --network="host" --rm \  
-v ${ROBOT_TESTS_DIRECTORY}:/opt/robot-tests/tests \  
-v ${ROBOT_RESULTS_DIRECTORY}:/opt/robot-tests/results \  
${ROBOT_IMAGE_NAME}:${ROBOT_VERSION} \  
--variable NGINX_HOSTNAME:${NGINX_HOSTNAME} \  
${ROBOT_TESTS_INCLUDE} ${ROBOT_TEST_OPTIONS}
```

As seen in the above screenshot, the Docker command has 4 parts:

- “docker run -t --network=“host” --rm”: this will call Docker and set some useful options:
 - Run: launch Docker image on Docker.
 - -t: use tty
 - --network: this means Docker image will use the same networks than host.
 - --rm: this option will remove image after end execution of tests from Docker environment.
- “-v <LOCAL_DIRECTORY>:<DOCKER_VOLUME>”: this option will attach local directory to volumes defined under robot Docker image. It only has 2 volumes:
 - /opt/robot-tests/tests: At this volume Docker image will expect to attach tests directory of repository, including all robot code.
 - /opt/robot-tests/results: At this volume robot will store reports generated after execution of tests. To get access to these reports, a folder on the host must be provided, otherwise these reports will be lost.
- “\${ROBOT_IMAGE_NAME}:\${ROBOT_VERSION}”: This part indicates the Docker image and version previously generated that Docker will run.
- Options after Robot Image selection: The command after Docker image information will be sent as a part of robot command executed inside Docker. This means input variables can be placed:
 - --variable: This allows setting variables used by robot tests cases developed as input at call.
 - --include: This option sets tags to be executed where selected tests will execute robot.

To check if all CAPIF services are running properly in local machine after executing run.sh, the following command should be used:

```
./check_services_are_running.sh
```

This shell script will return 0 if all services are running properly.

After running all the tests, a report is produced gathering test results:

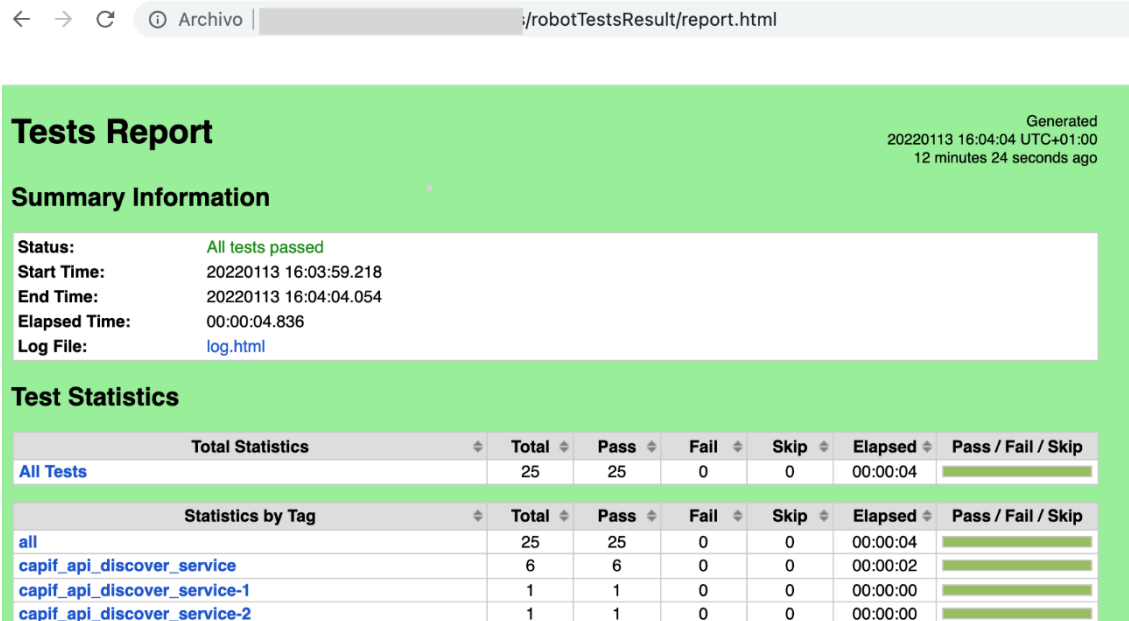


Figure 33. CAPIF Tool Results – Test Report

3.3 PERFORMANCE TESTING

For the analysis of the performance of the software components a standardized Test Case has been defined, with an emphasis on the collection of access time metrics while stressing the components with a large number of requests to all of the endpoints of interest. All of the separate tests follow the same template, though the actual implementation of each test has some differences depending on the workflow and information required by the particular endpoint.

EVOLVED-5G Test Case Template	-ID number-	Generic Endpoint Performance Test	Access Time
Scenario (storyline)	<p><i>Description of the motivation and the scope of the test in a qualitative level. What is the reference scenario in a real (industrial) environment that we want to capture with this test?</i></p> <p>The objective of the test is to measure the mean access time of the tested. The test also ensures that the endpoint is able to reliably provide the correct response.</p>		
Testing Infrastructure (Pre-conditions)	<p><i>Information related to all the parameters that affect the values of the KPI/KVIs/KVIs to be measured, network deployment and environment conditions, etc. [Any requirement that needs to be done before execution of this test case. A list of test specific pre-conditions that need to be met by the SUT including information about equipment configuration, traffic descriptor]</i></p> <ul style="list-style-type: none"> The set of software and hardware components involved + their configuration The service type + the traffic that is involved in the process The component that exposes the tested endpoint is running and listening for requests at a known address The component that exposes the tested endpoint is prepared with the minimal set of configuration values required for the testing process 		
Target KPI/KVI	<p><i>Here goes the definition of the target KPI/KVI. Each test case targets only one KPI/KVI (main KPI/KVI). However, secondary measurements from complementary KPI/KVIs can be added as well. The definition of the main KPI/KVI specializes the related target metric (the ID of the</i></p>		

	<p><i>related target metric is declared in the first row of this template). More precisely, the definition of the main KPI/KVI /KVI declares:</i></p> <ul style="list-style-type: none"> <i>The definition of the KPI/KVI+ (if applicable) a secondary list of KPI/KVIs useful to interpret the values of the target KPI/KVI.</i> <i>The reference points from which the measurement(s) will be performed</i> <i>The reference protocol stack level where the measurement is performed</i> <i>Target values + theoretical value space</i> <p>The target KPI is the Mean Access Time of the tested endpoint, which is defined as endpoint the mean delay between the time in which a client sends a request to the tested endpoint till the time in which the client receives a correctly formed response to the request.</p> <p>In order to obtain a statistically meaningful result the endpoint is tested 100 times. The endpoint must provide a response with a delay that is below a certain threshold in all request, in order to consider the test as Successful.</p>
Test Case Sequence	<p>Specializes the measurement process (methodology) of the metric for the selected underlay system. In this field:</p> <ul style="list-style-type: none"> The steps to be followed for performing the measurements are specified The iterations required, the monitoring frequency, etc., are declared. <ul style="list-style-type: none"> ○ The testing framework configures in the component all the required values that could not be prepared as part of the pre-conditions, such as creation of test users or entities, or checks if such configuration has already been performed by a previous test. ○ The following actions are repeated 100 times in order to obtain statistically meaningful results while also stressing the tested component: <ol style="list-style-type: none"> 2.1. The testing framework prepares any necessary data payloads for the tested endpoint, and/or makes use of other endpoints in order to prepare the component for receiving a request in the tested endpoint. 2.2. The testing framework sends a request to the tested endpoint, measuring the time required to receive a response. 2.3. The testing framework checks the received response: <ul style="list-style-type: none"> - If it is not well formatted or otherwise unexpected the test is finalized and considered Failed. - If the response is the expected, but the delay exceeds the defined threshold, the test is considered Failed, but continues in order to calculate a more accurate mean access time. 2.4. The measured delay is used to calculate the mean access time 2.5. If necessary, the testing framework performs any necessary cleanup before the next iteration starts. ○ Once all iterations have been completed (or an error has been detected): <ul style="list-style-type: none"> • If all iterations have been completed, and all the measured delays are below the defined threshold, then the test is considered Successful. • If all iterations have been completed, but any of them had a delay above the defined threshold, then the test is considered Failed. • If any of the iterations has not been completed (due to receiving an unexpected response or because of a runtime error), then the test is considered Failed.

Performance tests of the software component have been implemented using Robot Framework [10]. For all the tests, the basic functionality for the timing management and the calculation of the mean access time have been encapsulated in a reusable keyword (which are akin to methods in programming languages), with another keyword dedicated to the final verdict calculation. All of the implemented tests follow the template as depicted in the figure below, in which the defined keywords can be also shown.

*** Keywords ***

Handle Timing

```
[Arguments]    ${elapsed}    ${iteration}    ${average}    ${success}
${timespan}    Evaluate \
                ${elapsed.seconds}+(${elapsed.microseconds}/1000000.0)
Log To Console    <<<[${TEST_NAME}]${iteration}=${timespan}>>>
IF    ${timespan} < ${THRESHOLD}
    Log To Console    Success
    ${success}    Evaluate    ${success}+${1}
ELSE
    Log To Console    Fail
END
IF    ${iteration} < ${1}
    ${average}=    Set Variable    ${timespan}
ELSE
    ${average}=    Evaluate \
                    ((${average}*(${iteration}))+${timespan})/${iteration+1}
END
[Return]    ${success}    ${average}
```

Handle End Results

```
[Arguments]    ${success}    ${average}
Log To Console    \
    <<<[${TEST_NAME}];Success=${success}/${ITERATIONS};Average=${average}>>>
IF    ${success} < ${ITERATIONS}
    Fail    Detected response times above threshold
END
```

*** Test Cases ***

Example Endpoint Test

```
# Prepare general variables for the test
${success}=    Set Variable    ${0}
${average}=    Set Variable    ${0}

FOR    ${index}    IN RANGE    ${ITERATIONS}
    Log To Console    Iteration: ${index}

    # Prepare any required payloads or make use of additional
    # endpoints to prepare the component (step 2.1)

    # Step 2.2
    ${resp}=    GET    <endpoint>    headers=${header}

    # Handle Timing performs the required calculations (steps 2.3
    # and 2.4)
    ${success}    ${average}    Handle Timing \
        ${resp.elapsed}    ${index}    ${average}    ${success}

    # Any cleanup necessary before the next iteration
    # is performed here (step 2.5)
END

Handle End Results    ${success}    ${average}
```

Figure 34. Common performance tests implementation

In order to support the execution of the tests additional functionality that is more tailored to each of the software components have been defined as Robot Framework [10] keywords, in order to allow the reusability of common actions such as the creation of test users or the retrieval of access tokens, as well as for more specific needs such as the definition of a test cell in the NEF Emulator. This functionality is described in detail in the following sections.

3.3.1 NEF Emulator

Aside from the basic functionality defined in the previous section, certain actions have been implemented in order to automatically generate a valid test scenario for the NEF emulator. This includes:

- The creation of new test users in the NEF Emulator.
- The retrieval of access tokens for use in further requests.
- The creation of additional gNodeBs and cells.
- The creation of new UEs, and the configuration of movement paths.
- The initialization and finalization of the movement of a UE.

Using this functionality, the test suite defined in Robot Framework is able to check if the required entities are already configured and, if not, to automatically create them before the actual execution of the tests.

The following tests for analyzing the performance of the following features have been implemented during the initial validation phase:

- Monitoring Events API:
 - List Active Event Subscription Performance
 - Event Subscription Creation Performance
 - Event Subscription Read Performance
 - Event Subscription Update Performance
 - Event Subscription Delete Performance
- Session with QoS API:
 - List Active QoS Subscription Performance
 - QoS Subscription Creation Performance
 - QoS Subscription Read Performance
 - QoS Subscription Update Performance
 - QoS Subscription Delete Performance

Table 6 shows the results obtained while testing the NEF Emulator. For this test a threshold value of 500 milliseconds was selected. This value can be considered fast enough for many uses and gives margin for sporadic network issues.

Table 8 NEF Emulator performance test result.

API	Test	Average access time (mS)	Success ratio
Monitoring Events API	List Active Event Subscription	9.046	100% - Success
	Event Subscription Creation	15.463	100% - Success
	Event Subscription Read	9.54	100% - Success
	Event Subscription Update	11.683	100% - Success
	Event Subscription Delete	10.346	100% - Success
	List Active QoS Subscription	36.298	100% - Success

Session with QoS API	QoS Subscription Creation	43.313	100% - Success
	QoS Subscription Read	37.885	100% - Success
	QoS Subscription Update	39.954	100% - Success
	QoS Subscription Delete	37.622	100% - Success

3.3.2 CAPIF Tool

The usage of the endpoints of the CAPIF Tool does not depend much on the existence of a set of entities. For this reason, it was only necessary to implement keywords for the creation of test users, the retrieval of tokens, and for clearing all the test data after the finalization of a test. The test suite defined for CAPIF includes the following APIs and endpoints, covering all the functionality provided by the CAPIF Tool in the first validation phase:

- API Invoker Management:
 - Register Invoker Performance
 - Update Invoker Performance
 - Delete Invoker Performance
- Publish Service API:
 - Create Service API Performance
 - List All Service APIs Performance
 - Read Single Service API Performance
 - Update Service API Performance
 - Delete Service API Performance
- Service API Discover:
 - Discover All Published Services Performance
 - Discover Filtered Published Services Performance
- Events API:
 - Create Event Subscription Performance
 - Delete Event Subscription Performance
- Security API:
 - Create Security Context Performance
 - Update Security Context Performance
 - Retrieve Security Context Performance
 - Delete Security Context Performance

Table 9 shows the results obtained for the CAPIF Tool tests with a threshold value of 500 milliseconds.

Table 9 CAPIF Tool performance test result.

API	Test	Average access time (mS)	Success ratio
API Invoker Management	Register Invoker	46.151	100% - Success
	Update Invoker	50.147	100% - Success
	Delete Invoker	31.416	100% - Success
	Create Service API	34.123	100% - Success

Publish Service API	List All Service APIs	29.312	100% - Success
	Read Single Service API	31.09	100% - Success
	Update Service API	34.24	100% - Success
	Delete Service API	29.049	100% - Success
Service API Discover	Discover All Published Services	32.566	100% - Success
	Discover Filtered Published Services	30.954	100% - Success
Events API	Create Event Subscription	30.876	100% - Success
	Delete Event Subscription	32.005	100% - Success
Security API	Create Security Context	34.325	100% - Success
	Update Security Context	36.138	100% - Success
	Retrieve Security Context	32.129	100% - Success
	Delete Security Context	29.920	100% - Success

4 CONCLUSION

This deliverable presents the initial results of the first evaluation trials of the EVOLVED-5G platforms and the software components developed in the context of the project, providing a view on the status and performance of the infrastructure after the conclusion of the first phase (i.e., release A) of the project. This deliverable also showcases the use of the experimentation methodology and components that are part of the results stemming from Work Packages 2, 3 and 4.

With regards to the performance measured in the different EVOLVED-5G sites, the results can be considered good and demonstrate that the platforms can support use cases that demand a throughput of up to 500 MBps and latency over 10 ms. Further improvements related to Release 16, or the usage of mmW can allow us to reduce the latency below 10 ms. The results obtained when using the TSN deployment on the Málaga platform are already comparable to those achieved in wired TSN networks, and can be further improved as the configuration and implementation work carries on.

With regards to the software components, the NEF Emulator and CAPIF Tool have achieved outstanding results, with a 100% success rate in all tests and an extremely fast access times on all endpoints. Functional tests that cover 100% of the developed APIs are also in place, which allows us to continue the development of the components while avoiding regressions.

5 REFERENCES

- [1] A. Díaz Zayas, G. Caso, Ö. Alay, P. Merino, A. Brunstrom, D. Tsolkas, and H. Koumaras, "A Modular Experimentation Methodology for 5G Deployments: The 5GENESIS Approach," *Sensors*, vol. 20, no. 22, p. 6652, Nov. 2020 [Online]: <http://dx.doi.org/10.3390/s20226652>
- [2] EVOLVED-5G D2.2 - Design of NetApps development and evaluation environments [Online]: https://evolved-5g.eu/wp-content/uploads/2021/11/EVOLVED-5G-D2.2-v1.0_final.pdf
- [3] 3GPP TS 38.306 (V16.7.0), User Equipment (UE) radio access capabilities, Release 16, December 2021
- [4] 3GPP TS 23.222 Common API Framework for 3GPP Northbound APIs. [Online]: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3337>, Retrieve 07/22
- [5] 3GPP TS 29.222 Common API Framework for 3GPP Northbound APIs [Online]: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3450>
- [6] EVOLVED-5G, "D4.2 EVOLVED-5G Factory of the Future (FoF) NetApps" [Online]: https://evolved-5g.eu/wp-content/uploads/2022/06/EVOLVED-5G-D4.2_v1.0.pdf
- [7] EVOLVED-5G, "D3.1 Implementations and integrations towards EVOLVED-5G framework realisation" [Online]: <https://evolved-5g.eu/wp-content/uploads/2022/01/EVOLVED-5G-D3.1-v1.0.pdf>
- [8] EVOLVED-5G, Deliverable D3.2 "NetApp Certification Tools and Marketplace development(intermediate)" [Online]: https://evolved-5g.eu/wp-content/uploads/2022/07/EVOLVED-5G-D3.2_FINAL.pdf
- [9] EVOLVED-5G, "D4.1 5G Exposure Capabilities for Vertical Applications (Intermediate)" [Online]: https://evolved-5g.eu/wp-content/uploads/2022/03/EVOLVED-5G-D4.1_v2.0-final.pdf
- [10] Robot Framework – Homepage [Online]: <https://robotframework.org>
- [11] CAPIF standard specification YAMLS [Online]: https://github.com/jdegre/5GC_APIs.
- [12] Nemo Outdoor 5G NR Drive Test Solution [Online]: <https://www.keysight.com/es/en/product/NTA00002B/nemo-outdoor-5g-nr-drive-test-solution.html>
- [13] InfluxData – Homepage [Online]: <https://www.influxdata.com/>
- [14] Grafana Labs – Homepage [Online]: <https://grafana.com/>
- [15] Programming Protocol-independent Packet Processors (P4) Open Source Programming Language – Homepage [Online]: <https://p4.org/>
- [16] 5GENESIS, "D2.4 Final report on facility design and experimentation planning" [Online]: https://5genesis.eu/wp-content/uploads/2020/07/5GENESIS_D2.4_v1.0.pdf
- [17] IEEE 802.1 Time-Sensitive Networking Task Group – Homepage [Online]: <https://www.ieee802.org/1/pages/tsn.html>
- [18] 5G-ACIA, 5G Alliance for Connected Industries and Automation, White Paper 5G for Connected Industries and Automation Second Edition, February 2019.
- [19] 5Genesis, "D5.2 System-Level Tests and Verification (Release B)" [Online]: https://5genesis.eu/wp-content/uploads/2021/10/5GENESIS_D5.2_v1.0.pdf

- [20] 5Genesis, “D5.4 Documentation and supporting material for 5G stakeholders (Release B)” [Online]: https://5genesis.eu/wp-content/uploads/2021/08/5GENESIS-D5.4_v1.0.pdf

6 ANNEXES

6.1 TEST CASE TEMPLATE

EVOLVED-5G Test Case Template	-ID number-	-Title-	- Target Metric (KPI family) -
Scenario (storyline)	Description of the motivation and the scope of the test in a qualitative level. What is the reference scenario in a real (industrial) environment that we want to capture with this test?		
Testing Infrastructure (Pre-conditions)	Information related to all the parameters that affect the values of the KPIs to be measured, network deployment and environment conditions, etc. [Any requirement that needs to be done before execution of this test case. A list of test specific pre-conditions that need to be met by the SUT including information about equipment configuration, traffic descriptor]		
	<ul style="list-style-type: none"> The set of software and hardware components involved + their configuration The service type + the traffic that is involved in the process 		
Target KPI	Here goes the definition of the target KPI. Each test case targets only one KPI (main KPI). However, secondary measurements from complementary KPIs can be added as well. The definition of the main KPI specializes the related target metric (the ID of the related target metric is declared in the first row of this template). More precisely, the definition of the main KPI declares:		
	<ul style="list-style-type: none"> The definition of the KPI + (if applicable) a secondary list of KPIs useful to interpret the values of the target KPI. The reference points from which the measurement(s) will be performed The reference protocol stack level where the measurement is performed Target values + theoretical value space 		
Test Case Sequence	Specializes the measurement process (methodology) of the metric for the selected underlay system. In this field:		
	<ul style="list-style-type: none"> The steps to be followed for performing the measurements are specified The iterations required, the monitoring frequency, etc., are declared. 		

6.2 ATHENS PLATFORM TEST CASE TEMPLATES

6.2.1 DL throughput (NCSRD Demokritos)

EVOLVED-5G Test Case Template	-NCSRD_Downlink-	-DL Throughput-	- Throughput (Mbps) -
Scenario (storyline)	<p>This test evaluates the data rate of a 5G SA network in the downlink direction. The main goal of this test is to assess the throughput of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site) and compare the results with theoretical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.</p>		
Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> OnePlus 8 PRO 5G (COTS UE) Amarisoft RAN (5G NR Rel. 16) Amarisoft Core (5GC Rel. 16) Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> UMA iPerf (Android Application) OpenTAP for automated testing (iPerf TAP plugin) Open5Genesis iPerf probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> VM with Open5Genesis iPerf probe is up and running COTS UE has 5G connectivity UMA iPerf application is installed on the COTS UE All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> n78 band ARFCN 632628, 3489.42 MHz 50 MHz channel bandwidth 30 KHz SCS TDD, 7 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols 256QAM modulation in DL 2x2 MIMO layers 		
Target KPI	<p>The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:</p> $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{\text{Layers}}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\text{max}} \cdot \frac{N_{\text{PRB}}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$		

Test Case Sequence	<ol style="list-style-type: none"> 1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start UMA iPerf android application in server mode 4. Instructing VM iPerf probe to generate traffic towards the UE 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (UE), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards
---------------------------	---

6.2.2 UL throughput (NCSRD Demokritos)

EVOLVED-5G Test Case Template	- NCSRD_uplink -	-UL Throughput-	- Throughput (Mbps) -
Scenar io	<p>This test evaluates the data rate of a 5G SA network in the uplink direction. The main goal of this test is to assess the throughput of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site) and compare the results with theoretical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.</p>		
Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> • OnePlus 8 PRO 5G (COTS UE) • Amarisoft RAN (5G NR Rel. 16) • Amarisoft Core (5GC Rel. 16) • Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> 1. VM with Open5Genesis iPerf probe is up and running 2. COTS UE has 5G connectivity 3. UMA iPerf application is installed on the COTS UE 4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> • n78 band • ARFCN 632628, 3489.42 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, 7 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols • 256QAM modulation in UL • SISO layer 		

Target KPI	<p>The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:</p> $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{\text{Layers}}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\text{max}} \cdot \frac{N_{\text{PRB}}^{BW^{(j)}, \mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$
Test Case Sequence	<ol style="list-style-type: none"> 1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start iPerf probe on VM in server mode 4. Instructing UMA iPerf android application to generate traffic towards the iPerf probe on VM 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (VM), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards

6.2.3 Best UL throughput (NCSRD Demokritos)

EVOLVED-5G Test Case Template	- NCSRD_best_uplink -	-UL Throughput-	- Throughput (Mbps) -
Scenar io	<p>This test evaluates the data rate of a 5G SA network in the uplink direction. The main goal of this test is to assess the throughput of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site) and compare the results with theoretical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.</p>		

Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> OnePlus 8 PRO 5G (COTS UE) Amarisoft RAN (5G NR Rel. 16) Amarisoft Core (5GC Rel. 16) Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> UMA iPerf (Android Application) OpenTAP for automated testing (iPerf TAP plugin) Open5Genesis iPerf probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> VM with Open5Genesis iPerf probe is up and running COTS UE has 5G connectivity UMA iPerf application is installed on the COTS UE All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> n78 band ARFCN 632628, 3489.42 MHz 50 MHz channel bandwidth 30 KHz SCS TDD, 1 DL slots, 8 UL slots, 1 special slot with 10 DL symbols and 2 UL symbols 256QAM modulation in UL SISO layer
Target KPI	<p>The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:</p> $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{Layers}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$
Test Case Sequence	<ol style="list-style-type: none"> Instantiation of the slice, deployment of VM running the iPerf probe Run script to ensure that the service on VM is running Start iPerf probe on VM in server mode Instructing UMA iPerf android application to generate traffic towards the iPerf probe on VM Stop iPerf probe on VM Retrieve experiment results from the iPerf server (VM), Extract KPIs and persist to database Iterate steps 3-7 three times Generate statistical analysis and graphical timeline dashboards

6.2.4 DL throughput (Cosmote)

EVOLVED-5G	-COS_Download-	-DL Throughput-	- Throughput (Mbps) -
------------	----------------	-----------------	-----------------------

Test Case Template	
Scenario (storyline)	<p>This test evaluates the data rate of a 5G SA network in the downlink direction. The main goal of this test is to assess the throughput of the 5G infrastructure that lays on the Athens platform (i.e., both Cosmote and NCSR Demokritos sites) and compare the results with theoretical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.</p>
Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> • Samsung Galaxy S20 5G (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Athonet Core (EPC) • Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> 1. VM with Open5Genesis iPerf probe is up and running 2. COTS UE has 5G connectivity 3. UMA iPerf application is installed on the COTS UE 4. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> • n78 band • ARFCN 646000, 3690 MHz • 100 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUUDD (tdLTE) • 256QAM modulation in DL • 2x2 MIMO layers
Target KPI	<p>The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:</p> $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{\text{Layers}}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{PRB}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$

Test Case Sequence	<ol style="list-style-type: none"> 1. Instantiation of the slice, deployment of VM running the iPerf probe 2. Run script to ensure that the service on VM is running 3. Start UMA iPerf android application in server mode 4. Instructing VM iPerf probe to generate traffic towards the UE 5. Stop iPerf probe on VM 6. Retrieve experiment results from the iPerf server (UE), 7. Extract KPIs and persist to database 8. Iterate steps 3-7 three times 9. Generate statistical analysis and graphical timeline dashboards
---------------------------	---

6.2.5 UL throughput (Cosmote)

EVOLVED-5G Test Case Template	- COS_uplink -	-UL Throughput-	- Throughput (Mbps) -
Scenario	<p>This test evaluates the data rate of a 5G SA network in the uplink direction. The main goal of this test is to assess the throughput of the 5G infrastructure that lays on the Athens platform (i.e., both Cosmote and NCSRD Demokritos sites) and compare the results with theoretical values. Furthermore, the functionality of the overall Open5Genesis framework is evaluated, including slice deployment, placement and provisioning times.</p>		
Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> • Samsung Galaxy S20 5G (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Athonet Core (EPC) • Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> • UMA iPerf (Android Application) • OpenTAP for automated testing (iPerf TAP plugin) • Open5Genesis iPerf probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> 5. VM with Open5Genesis iPerf probe is up and running 6. COTS UE has 5G connectivity 7. UMA iPerf application is installed on the COTS UE 8. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> • n78 band • ARFCN 646000, 3690 MHz • 100 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUUDD (tdLTE) • 64QAM modulation in UL • SISO layer 		

Target KPI	<p>The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:</p> $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{\text{Layers}}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\text{max}} \cdot \frac{N_{\text{PRB}}^{BW^{(j)}, \mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$
Test Case Sequence	<ol style="list-style-type: none"> 19. Instantiation of the slice, deployment of VM running the iPerf probe 20. Run script to ensure that the service on VM is running 21. Start iPerf probe on VM in server mode 22. Instructing UMA iPerf android application to generate traffic towards the iPerf probe on VM 23. Stop iPerf probe on VM 24. Retrieve experiment results from the iPerf server (VM), 25. Extract KPIs and persist to database 26. Iterate steps 3-7 three times 27. Generate statistical analysis and graphical timeline dashboards

6.2.6 RTT (NCSRD Demokritos)

EVOLVED- 5G Test Case Template	-NCSRD_RTT-	-RTT-	- Delay (ms) -
Scenario	This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site)		

Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> OnePlus 8 PRO 5G (COTS UE) Amarisoft RAN (5G NR Rel. 16) Amarisoft Core (5GC Rel. 16) Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> OpenTAP for automated testing (ping TAP plugin) Open5Genesis ping probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> VM with Open5Genesis ping probe is up and running COTS UE has 5G connectivity All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> n78 band ARFCN 632628, 3489.42 MHz 50 MHz channel bandwidth 30 KHz SCS TDD, 7 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols 256QAM modulation in UL SISO layer 20 msec scheduling request period
Target KPI	<p>The target KPI of this test is to measure the RTT in msec. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer. No complementary measurements are considered in this experiment.</p>
Test Case Sequence	<ol style="list-style-type: none"> Instantiation of the slice, deployment of VM running the ping probe Run script to ensure that the service on VM is running Instructing ping probe (VM) to send ICMP echo requests to the target UE Stop ping probe on VM Retrieve experiment results from the ping server (VM), Extract KPIs and persist to database Iterate steps 3-7 three times Generate statistical analysis and graphical timeline dashboards

6.2.7 RTT low latency (NCSRD Demokritos)

EVOLVED-5G Test Case Template	- NCSRD_RTT_low_latency -	-RTT-	- Delay (ms) -
Scenario	<p>This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the Athens platform (i.e., NCSRD Demokritos site)</p>		

Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> OnePlus 8 PRO 5G (COTS UE) Amarisoft RAN (5G NR Rel. 16) Amarisoft Core (5GC Rel. 16) Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> OpenTAP for automated testing (ping TAP plugin) Open5Genesis ping probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> VM with Open5Genesis ping probe is up and running COTS UE has 5G connectivity All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> n78 band ARFCN 632628, 3489.42 MHz 50 MHz channel bandwidth 30 KHz SCS TDD, 2 DL slots, 2 UL slots, 1 special slot with 6 DL symbols and 4 UL symbols (i.e., in 2.5 msec) 256QAM modulation in UL SISO layer 0.5 msec scheduling request period
Target KPI	The target KPI of this test is to measure the RTT in msec. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer. No complementary measurements are considered in this experiment.
Test Case Sequence	<ol style="list-style-type: none"> Instantiation of the slice, deployment of VM running the ping probe Run script to ensure that the service on VM is running Instructing ping probe (VM) to send ICMP echo requests to the target UE Stop ping probe on VM Retrieve experiment results from the ping server (VM), Extract KPIs and persist to database Iterate steps 3-7 three times Generate statistical analysis and graphical timeline dashboards

6.2.8 RTT (Cosmote – NCSR Demokritos)

EVOLVED-5G Test Case Template	-NCSRDT_RTT-	-RTT-	- Delay (ms) -
Scenario	This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the Athens platform, considering the multi domain deployment, including both Cosmote and Demokritos premises that are interconnected through GRNET.		

Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> • Samsung Galaxy S20 5G (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Athonet Core (EPC) • Dell Laptop <p>Software components</p> <ul style="list-style-type: none"> • UMA iPerf (Android Application) • OpenTAP for automated testing (ping TAP plugin) • Open5Genesis ping probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> 1. VM with Open5Genesis ping probe is up and running 2. COTS UE has 5G connectivity 3. All the necessary test descriptors are properly defined (i.e., NSD on Slice Manager, test case on ELCM, tap plan on OpenTAP) <p>Radio Configuration:</p> <ul style="list-style-type: none"> • n78 band • ARFCN 646000, 3690 MHz • 100 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD (tdLTE) • 256QAM modulation in DL • 2x2 MIMO layers
Target KPI	<p>The target KPI of this test is to measure the RTT in msec. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer. No complementary measurements are considered in this experiment.</p>
Test Case Sequence	<ol style="list-style-type: none"> 1. Instantiation of the slice, deployment of VM running the ping probe 2. Run script to ensure that the service on VM is running 3. Instructing ping probe (VM) to send ICMP echo requests to the target UE 4. Stop ping probe on VM 5. Retrieve experiment results from the ping server (VM), 6. Extract KPIs and persist to database 7. Iterate steps 3-7 three times 8. Generate statistical analysis and graphical timeline dashboards

6.3 UMA PLATFORM TEST CASE TEMPLATES

6.3.1 DL throughput (UMA)

EVOLVED-5G Test Case Template	-UMA_Downlink-	-DL Throughput-	- Throughput (Mbps) -
Scenario (story line)	<p>This test evaluates the data rate of a 5G SA network in the downlink direction. The main goal of this test is to assess the throughput of the 5G infrastructure that lays on the UMA platform and compare the results with theoretical values.</p>		
Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> One plus 9 5G (COTS UE) Nokia Airtscale RAN (5G NR Rel. 15) Athonet Core (EPC) <p>Software components</p> <ul style="list-style-type: none"> UMA iPerf (Android Application) OpenTAP for automated testing (iPerf TAP plugin) Open5Genesis iPerf probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> VM with Open5Genesis iPerf probe is up and running COTS UE has 5G connectivity UMA iPerf application is installed on the COTS UE Network is configured <p>Radio Configuration:</p> <ul style="list-style-type: none"> n78 band ARFCN 651666, 3774.990 MHz 50 MHz channel bandwidth 30 KHz SCS TDD, DDDSUUUDD (tdLTE) 256QAM modulation in DL 4x4 DL MIMO layers 		
Target KPI	<p>The target KPI of this test is to measure the downlink throughput (i.e., Mbps). Primary results such as mean, standard deviation, median, min and max values will be provided. The protocol to generate network traffic is UDP, thus secondary KPIs such as packet loss rate (%) and jitter (ms) are included. Since the UMA iPerf android application is used, the protocol layer where the measurement is performed is the application layer. Finally, a comparison between the conducted results and the theoretical value is provided. The calculation formula for the theoretical calculation, adopted from [TS 38306-g70], is described below:</p> $\text{data rate (in Mbps)} = 10^{-6} \cdot \sum_{j=1}^J \left(v_{\text{Layers}}^{(j)} \cdot Q_m^{(j)} \cdot f^{(j)} \cdot R_{\max} \cdot \frac{N_{\text{PRB}}^{BW(j),\mu} \cdot 12}{T_s^{\mu}} \cdot (1 - OH^{(j)}) \right)$		

Test Case Sequence	<ol style="list-style-type: none"> 1. Start UMA iPerf android application in server mode 2. Start iPerf client to generate traffic towards the UE 3. Stop iPerf probe on VM 4. Retrieve experiment results from the iPerf server (UE), 5. Extract KPIs and persist to database 6. Iterate steps 2-4 25 times 7. Generate statistical analysis and graphical timeline dashboards
---------------------------	--

6.3.2 RTT (UMA)

EVOLVED-5G Test Case Template	-UMA_RTT-	-RTT-	- Delay (ms) -
Scenario	This test evaluates the end-to-end RTT of a 5G SA network. The main goal of this test is to assess the delay of the 5G infrastructure that lays on the UMA platform.		
Testing Infrastructure (Pre-conditions)	<p>Hardware and Software components:</p> <p>Hardware components</p> <ul style="list-style-type: none"> • Samsung Galaxy S20 5G (COTS UE) • Nokia Airscale RAN (5G NR Rel. 15) • Athonet Core (EPC) <p>Software components</p> <ul style="list-style-type: none"> • OpenTAP for automated testing (ping TAP plugin) • Open5Genesis ping probe <p>Pre-conditions:</p> <ol style="list-style-type: none"> 4. VM with Open5Genesis ping probe is up and running 5. COTS UE has 5G connectivity 6. Network is configured <p>Radio Configuration:</p> <ul style="list-style-type: none"> • n78 band • ARFCN 651666, 3774.990 MHz • 50 MHz channel bandwidth • 30 KHz SCS • TDD, DDDSUUDDD (tdLTE) • 256QAM modulation in DL • 4x4 DL MIMO layers 		
Target KPI	The target KPI of this test is to measure the RTT in msec. Primary results such as mean, standard deviation, median, min and max values will be provided. Since the ping software is used, which operates by means of ICMP packets, the protocol layer where the measurement is performed is the network layer.		
Test Case Sequence	<ol style="list-style-type: none"> 1. Start ping probe install at the UE to send ICMP echo requests to the main compute node 2. Stop ping probe on UE 3. Retrieve experiment results from the ping UE 4. Extract KPIs and persist to database 5. Iterate steps 1-3 three times 6. Generate statistical analysis and graphical timeline dashboards 		