



6G-SANDBOX

Supporting Architectural and technological
Network evolutions through an intelligent, secured
and twinning enabled Open eXperimentation
facility

Deliverable D5.4 – Final Version of
Experimentation Framework

Date: 30/06/2025

Version: V1.0

Grant Agreement	101096328
Document number	D5.4
Document title	Final Version of Experimentation Framework
Lead Beneficiary	KEYD
Editor(s)	Filip Ivanovic (KEYD)
Author(s)	Filip Ivanovic, (KEYD), Almudena Diaz Zayas (UMA), Dimitris Tsolkas (FOG), Jesus Macias Portela, Alvaro Curto Merino (TID), Setaki Fotini (COS), Anastasius Gavras (EURE), Vaios Koumaras (INF), Bjoern Riemer (FOKUS), Apostolis Salkintzis (LNV), George Makropoulos (NCSR)
Dissemination level	Public
Contractual date of delivery	30 th June 2025
Status	Final
Version	V1.0
File name	6G-SANDBOX_D5.4_V1.0.pdf

Document History

Version

V1.0 Initial Release

TABLE OF CONTENTS

Abstract.....	4
Keywords	4
Glossary	4
1 Introduction	5
1.1 Document Structure and Experimentation Methodology Overview	5
2 Experiment Requirements and Trial Network Design	5
3 Trial Network Creation	6
3.1 Trial Network Report.....	6
3.2 Trial Network Operations	9
3.3 Sample Trial Network integrated with External Equipment	10
4 Running Experiments	12
4.1 Running Experiments with ELCM	12
4.2 Running Experiments with OpenTAP Campaign Manager	15
5 Post-Processing	16
5.1 General Post Processing Methodology	16
5.2 Platform Specific Testing Post Processing methodology.....	17
6 Data Collection and Storage	17
6.1 Methodology.....	17
6.2 Metadata handling.....	18
7 Conclusion.....	18
8 References	19
9 Annex 1: Full Deployment Report of Trial Network “ EUCNCDemo”	21

Table of Figures

Figure 1: Trial Network markdown report example 7

Figure 2: Trial Network markdown rendered reporting the successful deployment of component ‘vm_kvm-ubuntu22_04’ inside the Trial Network ‘eucncdemo’ 7

Figure 3: Template report for 6G-Sandbox library components..... 8

Figure 4: Cover of a Trial Network Report PDF file. The full report can be seen in Annex 1 9

Figure 5: Diagram of the Trial Network design for EUCNC 2025 demo 10

Figure 6: EUCNC 2025 Trial Network Descriptor..... 12

Figure 7: ELCM graphical interface for the creation of experiments 13

Figure 8: ELCM graphical interface for the execution of experiments 13

Figure 9: Editing the tasks within an ELCM test case 14

Figure 10: The main test execution interface of the OpenTAP Campaign manager..... 16

Figure 11: Data reusability in 6G-SANDBOX..... 17

Figure 12: A Common metadata template is used to store metadata to a single Metadata Repository System for all SNS-JU projects 18



ABSTRACT

6G-SANDBOX defines a large-scale testing framework for 6G enabled networking, and as such its testing methodology contains many novel processes and technologies. This document is the project's deliverable that presents the final description of the experimental methodology employed within 6G-SANDBOX. The document follows the format of a walk-through guide; it is structured in sections ordered following the workflow of the experimentation process execution, each process step being detailed in a separate section. The experiment definition, trial network creation, experiment execution, data post processing and data storage/management are all covered in the document, in such a way that an experimenter can use this document to understand and appropriately execute an experiment within 6G-SANDBOX.

KEYWORDS

6G, Testing, Methodology, Trial Networks, Automation, Experiment Life Cycle Management

GLOSSARY

6G-Library: A repository containing templates and definitions used to generate Trial Network reports.

Bastion VM: A secure access point to the TN providing VPN, DNS, and SSH gateway functionalities.

Component: A unit within a Trial Network, such as a VM or service, deployed with specific functionality.

Descriptor: YAML configuration defining components, dependencies, and parameters of a TN.

Experiment Life Cycle Manager (ELCM): A graphical user interface designed by the University of Malaga for the purpose of creating and executing experimentation

InfluxDB: A time series database developed by InfluxData for the purposes of storage and retrieval of time series data

Task: In ELCM – An individual executable function in the ELCM infrastructure

Test Case: In ELCM – A set of tasks compiled together to under one Execution ID

Test Campaign: In OpenTAP – A full set of experimentation instructions, comprising of various test plans and generating a large number of KPIs

Test Plan: In OpenTAP – A set of test execution commands comprising various test steps designed to generate a small number of KPIs

Test Step: In OpenTAP – An individual test execution command, consisting of anything from flow control to a tool command to initiation of post processing code

TNLCM: Trial Network Lifecycle Manager, the orchestrator that deploys, monitors, and finalizes the TN.

Trial Network (TN): A modular environment composed of deployable 6G Library components for network experimentation.

Trial Network Report: A markdown-based summary file generated for each component during deployment.

1 INTRODUCTION

6G-SANDBOX designs a full stack, top to bottom testing solution for 6G enabled networks and network components, and as such the methodology and theory behind testing conducted within 6G-SANDBOX is quite complex. It has also been shaped and molded over time from its original inception at the point of the proposal of 6G-SANDBOX, by the evolving needs and capabilities of the project partners and infrastructure. This deliverable therefore serves as the final reporting on the 6G-SANDBOX experimentation methodology, describing the process that was and is to be followed in the conduction of all experiments within 6G-SANDBOX

1.1 DOCUMENT STRUCTURE AND EXPERIMENTATION METHODOLOGY OVERVIEW

The deliverable is structured in an illustrative way, presenting the key steps of the 6G-SANDBOX experimentation methodology in the order that these steps are to be conducted. This means that this deliverable can also serve as a guide to experimenters and platforms admins, supporting them to conduct experiments on a 6G-SANDBOX platform. With this in mind, what follows in this subsection is an overview of the Experimentation Methodology within 6G-SANDBOX, as well as the corresponding section in the Deliverable that the methodology step is discussed in.

1. Experiment Ideation and Definition – This is the first step of the Experimentation Methodology, where an experimenter approaches 6G-SANDBOX with their idea and requirements, and these will get converted into specifications for the 6G-SANDBOX Experiment, detailing, specifically for the design of the Trial Network. This step is discussed in **Section 2**
2. Trial Network Creation – Once an experiment has been ideated and all required parameters are understood; the next step is to create the relevant Trial Network. From the point of view of the experimenter, if all necessary information has been provided, this is a process entirely done by 6G-SANDBOX site admins, with the experimenter receiving a report upon completion of the process detailing information about the access and operation of their new Trial Network. This is discussed in **Section 3**
3. Experiment Execution – With the experiment understood and the Trial Network spun up, the experimenter can now begin to run their experiment(s). There are two main ways this can be done through the 6G-SANDBOX infrastructure, those being through the use of ELCM or the use of OpenTAP Campaign Manager. The use of both of these methods is detailed in **Section 4**.
4. Post Processing – Once execution of the experiment or a given part of an experiment is complete, the generated data must be post processed in order to present the desired KPIs to the experimenter. This post processing depends heavily on the pre-established definitions of the KPI as well as the format of the initially generated data. This step is discussed in **Section 5**
5. Data Collection and Storage – One of the goals for 6G-SANDBOX is the generation of data sets that can be used to inform research in the 6G space for years to come, and as such the way that generated data is stored, manipulated and presented for future use is extremely important. This is discussed in **Section 7**

With a general overview of these 5 steps now in mind, each can be explored in more detail.

2 EXPERIMENT REQUIREMENTS AND TRIAL NETWORK DESIGN

In 6G-SANDBOX the experimentation methodology begins with the identification of the specific goals and requirements of each experiment. These requirements often include objectives such as validating KPIs under specific network conditions, testing a new protocol, or benchmarking AI/ML models. In order to begin the process of testing within 6G-SANDBOX, these requirements need to be transformed into the design of a Trial Network.

When identifying requirements, they fall into two main categories:

- Identification of the required resources (e.g., RAN node, edge servers, etc.). The requirements extracted at this point will be used for the definition of which components need to be included within the Trial Network. These can either be physical nodes or virtual deployments.
- Identification of the test conditions. The requirements extracted at this phase will be used to configure the components included in the Trial Network. This includes details such as the type of traffic that the network is expected to operate with.

This information can be collected in a variety of ways, but most often involves the experimenter being assigned an admin at the 6G-SANDBOX platform that their experiment will be running upon. The admin may organize a meeting, provide a questionnaire, or some combination of both in order to collect the necessary information. They will then relay this information internally, passing it any technical personnel that are required in order to parse and act upon the requirements.

Both of these initial identification steps are extremely important for the definition of the experiment but specifically the Trial Network, as making modifications to certain core aspects after deployment of the Trial Network is possible, but involves intervention from a 6G-SANDBOX site admin, as is detailed in Section 3. More granular definitions and information may be required for a specific test case or aspect of an experiment, especially if it involves a custom component or device under test produced by the experimenter. This typically involves defining a set of necessary conditions and actions that need to be undertaken, as well as how the generated data will need to be processed and stored. The sources of this generated data will also need to be defined, in terms of which logs from which components of the trial network will need to be saved. This information about the required executions and conditions is required and passed to the methods used by 6G-SANDBOX to perform experiment execution, which are discussed in Section 4. Through this method of keeping the definition of their experiment up to date through constant contact with the 6G-SANDBOX platform, the experimenter ensures their requirements and conditions are always met in order to enable their experimentation within 6G-SANDBOX.

3 TRIAL NETWORK CREATION

Once the experimenter's requirements are fully understood and all necessary resource and condition identifications have been made, the process of Trial Network creation can be initiated. Deliverables D4.1 [1] and D4.2 [2] delved into the process of how a Descriptor is created, and how to apply it into the TNLCM [3] to start the deployment of the necessary infrastructure. This Section will go in depth into the deployment and operations of a Trial Network from the perspective of the experimenter, describing the various actions performed and generated resources they will receive in order to aid them in the usage of the Trial Network.

3.1 TRIAL NETWORK REPORT

One of the key steps of the deployment pipeline of each TN component is the generation of a report file in markdown format. This report is generated from a custom template for each component (present in the 6G-Library repository [4]) and updated with the runtime variables of the pipeline and Ansible playbook [5] execution. The report is uploaded to the S3 object storage (MinIO [6]) along with the rest of the given Trial Network's files.

Following the modular approach of 6G-SANDBOX, each singular report is self-contained, and defined independently, directly corresponding to the details of the component it describes.

The typical component will generate a markdown report that is like the ones shown in Figures 1 and 2. Figure 1 shows the simple, text-based format, whereas Figure 2 shows the fully rendered format that will eventually appear in the final overall report on the Trial Network.

```
# eucncdemo-vm_kvm-ubuntu22_04

The component `eucncdemo-vm_kvm-ubuntu22_04` has been succesfully created.

<p align="center">
  <a href="https://marketplace.opennebula.io/appliance/4562be1a-4c11-4e9e-b60a-85a045f1de05">
    
  </a>
</p>

It consists of an Ubuntu 22.04 LTS Virtual Machine

## Important information:

- **OpenNebula VM ID**:` 3476`
- **VM memory**:` 4096 MiB`
- **VM VCPUs**:` 2`
- **VM available storage**:` 25 GiB`
- **VM network interfaces**:`
  ``json
  {'1275': '192.168.199.2'}
  ``
---
```

Figure 1: Trial Network markdown report example

2.7 EUCNCDEMO-VM_KVM-UBUNTU22_04
 The component eucncdemo-vm_kvm-ubuntu22_04 has been succesfully created.



It consists of an Ubuntu 22.04 LTS Virtual Machine

2.7.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3476
- **VM memory:** 4096 MiB
- **VM VCPUs:** 2
- **VM available storage:** 25 GiB
- **VM network interfaces:**
 {'1275': '192.168.199.2'}

Figure 2: Trial Network markdown rendered reporting the successful deployment of component 'vm_kvm-ubuntu22_04' inside the Trial Network 'eucncdemo'

Each components developer determines the content and level of detail in their report; however, the recommended approach is to follow the structure outlined in Figure 3.

```
# {{ tn_id }}-{{ entity_name }}
```

```
The component `{{ tn_id }}-{{ entity_name }}` has been successfully created.

(Optional link to a image that can be imported from the TNLCM at pdf render time)

Some short description of what the component is and how does it fit inside the Trial
Network.
E.g. what technologies are now enabled, or what other components does this component
depend on
This description can include links to external documentation, but it is advisable to
move more generic information to the end of the report/annex.

## Important information:

Ambiguous section, with a resume of the core metadata of this component.
This can include values like inputs/outputs/generated data, or the time range during
which this component will be available.

Also, usage information

## Functionality 1

Information about one functionality of this component. This can include links to
webUIs, tokens/credentials, etc.

## Functionality 2

Another functionality of this component. The order between different functionalities
is according to its relevance.
```

Figure 3: Template report for 6G-Sandbox library components

Most URLs in the generated reports require name resolution, so they will only be accessible if the experimenter is connected to 6G-SANDBOX through a provided VPN, and the trial network's DNS server is configured to correctly resolve the domains in question. Both VPN and the DNS server are services within the Trial Network's Bastion VM, which will be elaborated on in the next section.

Once a Trial Network becomes active – a state reached when all of its components have finished deploying – the TNLCM is ready to generate a final Trial Network report in PDF format. This report includes all the generated markdown reports of the Trial Network components, fully rendered with working links, and with the addition of a cover, an index, a generic introduction for non-experienced users and possible annexes. This report will be used by the experimenter to guide them through the more technical aspects of using their now deployed trial network.

6G-SANDBOX

Trial Network Report: eucncdemo

Date: 02/06/2025

Trial Network Report — eucncdemo — 6G-SANDBOX — HORIZON-JU-SNS-2022

Figure 4: Cover of a Trial Network Report PDF file. The full report can be seen in Annex 1

3.2 TRIAL NETWORK OPERATIONS

An active Trial Network makes multiple useful files available to the experimenter. In addition to the PDF of the final report, there are also files such as SSH keys for virtual machine access, an SSH configuration file, VPN clients/configurations, and many other types of necessary configuration files. Any files containing text will have their content copied into the final trial network report in their corresponding chapter as well. For the purposes of an experimenter beginning their interactions with the trial network for the first time, the most important chapter of the final report is the one that describes the Trial Network Bastion. This contains:

1. The ssh keypair to SSH into the rest of the components as user 'tnuser'. The keys are also available as separate files to be downloaded
2. Instructions on how to import the SSH keys and a preconfigured sshconfig file into the experimenter's testing device (preferably Unix-like, Windows not tested).
3. The generated WireGuard [7] VPN client configurations. These files are also available as separate files to download.
4. URL, user and password for the Technitium [8] DNS Server included in the Bastion.

The Bastion itself is not reachable by the experimenter due to the security risk that Bastion access poses for the rest of the 6G-SANDBOX infrastructure. However, the experimenter does have limited freedom and flexibility to troubleshoot name resolution, and the clients connected to their Trial Network.

In certain situations, the experimenter may wish to connect external equipment to trial network components, be it new measurement equipment or new components to be put under test after the trial network has already been deployed. By default, this is not possible, as opening new firewall exceptions during runtime is not supported. If this is desired or necessary, the intended process is to either know the intended exceptions in advance and add them to the Trial Network's Bastion prior to deployment or ask for manual intervention from an admin at the 6G-SANDBOX platform in question.

With all of this in mind, the experimenter is now ready to begin executing tests in their 6G-SANDBOX trial network. The details of the methodologies for doing this are described in Section 4.

3.3 SAMPLE TRIAL NETWORK INTEGRATED WITH EXTERNAL EQUIPMENT

For illustrative purposes, included here is an example of a deployed trial network alongside external equipment. This trial network was deployed for the EUCNC conference [9]. It supported a demonstration where a virtualized 5G core on the trial network was attached to a radio outside of the trial network, with multiple mobile phones attached to the core through said radio.

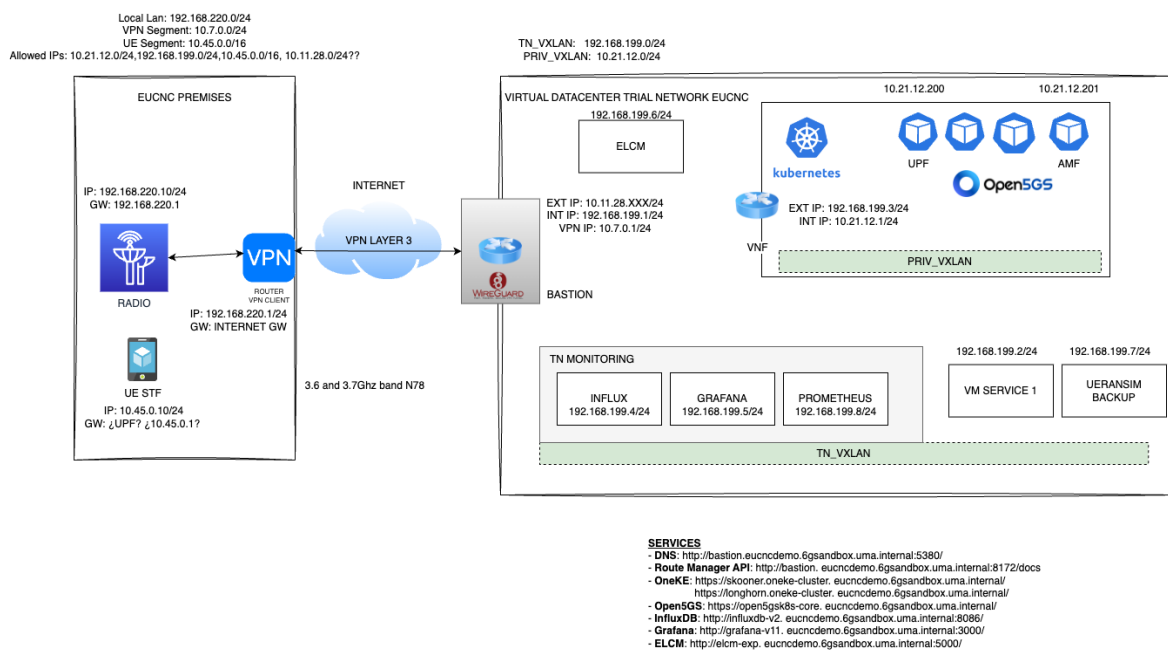


Figure 5: Diagram of the Trial Network design for EUCNC 2025 demo

The radio and UEs are configured beforehand with the specific parameters required from the core, such as the PLMN. To achieve the setup shown in Figure 5, the descriptor shown in Figure 6 is applied to the TNLCM and the trial network is activated.

```
trial_network:
  tn_vxlan:
    type: "tn_vxlan"
    dependencies: []
    input: {}
    # one_vxlan_netmask: 24 # Default value
    # one_vxlan_first_ip: "192.168.199.1" # Default value
    # one_vxlan_address_size: 254 # Default value

  tn_bastion:
    type: "tn_bastion"
    dependencies: ['tn_vxlan']
    input:
```

```

    one_bastion_vpn_clients: 5 # Default value is 1
    one_bastion_vpn_allowedips:
"192.168.199.0/24,10.7.0.0/24,10.21.12.0/24,10.45.0.0/16" # Default value is The
subnet of tn_vxlan, 192.168.199.0/24 in this case
    # one_bastion_routemanager_add: # Default value is []. Unnecessary as the WG
interface automatically sets this route up.
    #   - to: "192.168.220.0/24"
    #     dev: "wg0"
    one_bastion_fw_exceptions: # Default value is []
    - "192.168.220.0/24"
    one_bastion_nat_exceptions: # Default value is []
    - "192.168.220.0/24"
    - "10.21.12.200"
    - "10.21.12.201"

vnet-n2:
  type: "vnet"
  name: "n2"
  dependencies:
    - "tn_bastion"
  input: {}
  # one_vnet_first_ip: "10.21.12.1" # Default value
  # one_vnet_netmask: 24 # Default value
  # one_vnet_address_size: 100 # Default value

oneKE-cluster:
  type: "oneKE"
  name: "cluster"
  dependencies:
    - "vm_kvm-ubuntu22_04"
  input:
    one_oneKE_external_vnet: "tn_vxlan"
    one_oneKE_internal_vnet: "vnet-n2"
    one_oneKE_metallb_range: "10.21.12.200-10.21.12.240"

open5gs_k8s-core:
  type: "open5gs_k8s"
  name: "core"
  dependencies:
    - "oneKE-cluster"
  input:
    one_open5gs_k8s_target: "oneKE-cluster"
    # one_open5gs_k8s_amf_n2_ip: "10.21.12.200" # Default value
    # one_open5gs_k8s_upf_n3_ip: "10.21.12.201" # Default value
    # one_open5gs_k8s_ue_count: 20 # Default value
    one_open5gs_k8s_tac: 1 # Default value is 200
    one_open5gs_k8s_mcc: "999" # Default value is "001"
    one_open5gs_k8s_mnc: "70" # Default value is "01"
    one_open5gs_k8s_msin: "0000000100" # Default value is "0000000001"
    # one_open5gs_k8s_key: "465B5CE8B199B49FAA5F0A2EE238A6BC" # Default value
    # one_open5gs_k8s_opc: "E8ED289DEBA952E4283B54E88E6183CA" # Default value
    # one_open5gs_k8s_apn: "internet" # Default value
    # one_open5gs_k8s_s_nssai_sst: 1 # Default value
    # one_open5gs_k8s_s_nssai_sd: "000001" # Default value

influxdb-v2:
  type: "influxdb"
  name: "v2"
  dependencies: ["open5gs_k8s-core"]
  input: {}

```

```
grafana-v11:
  type: "grafana"
  name: "v11"
  dependencies: ["influxdb-v2"]
  input: {}

elcm-exp:
  type: "elcm"
  name: "exp"
  dependencies:
    - "influxdb-v2"
    - "grafana-v11"
  input:
    one_elcm_influxdb: "influxdb-v2"
    one_elcm_grafana: "grafana-v11"
```

Figure 6: EUCNC 2025 Trial Network Descriptor

After activation, the VPN node shown in Figure 5 would import and use the VPN configuration generated by the networks Bastion. In Figure 6, the list of whitelisted IPs for the Bastion is also present, with these relating to the physical equipment in use.

4 RUNNING EXPERIMENTS

With the trial network defined and deployed, the next step of the process is to actually run the tests desired by the experimenter. A key aspect of test execution within 6G-SANDBOX is that it should be done in an automated manner. This allows for the efficient execution of large-scale test campaigns with minimal input from the experimenter once execution has begun. 6G-SANDBOX supports two (2) concurrent methods for the automation of test execution, those being the use of the ELCM [10] and/or the OpenTAP Campaign Manager Tool [11].

4.1 RUNNING EXPERIMENTS WITH ELCM

The Experiment Lifecycle Manager (ELCM) [10] provides a new graphical interface for editing test cases and executing experiments. An experiment can consist of the execution of one or more test cases, which are managed and run by the ELCM sequentially.

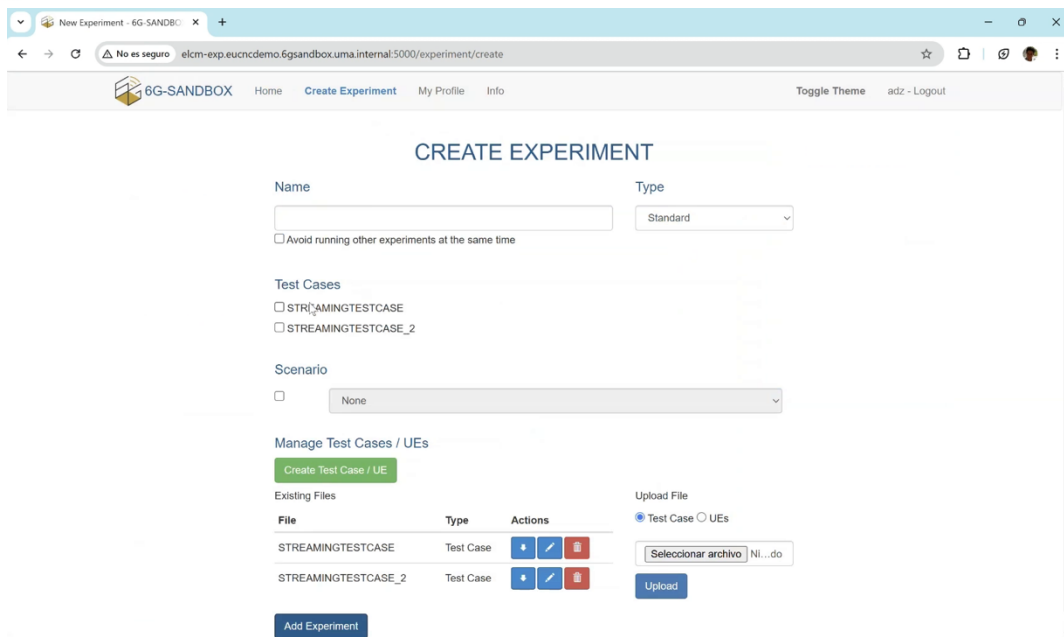


Figure 7: ELCM graphical interface for the creation of experiments

Once the experiment has been created, the experiment can be executed, and the results of all the executions can be also accessed, as seen in Figures 7 and 8.

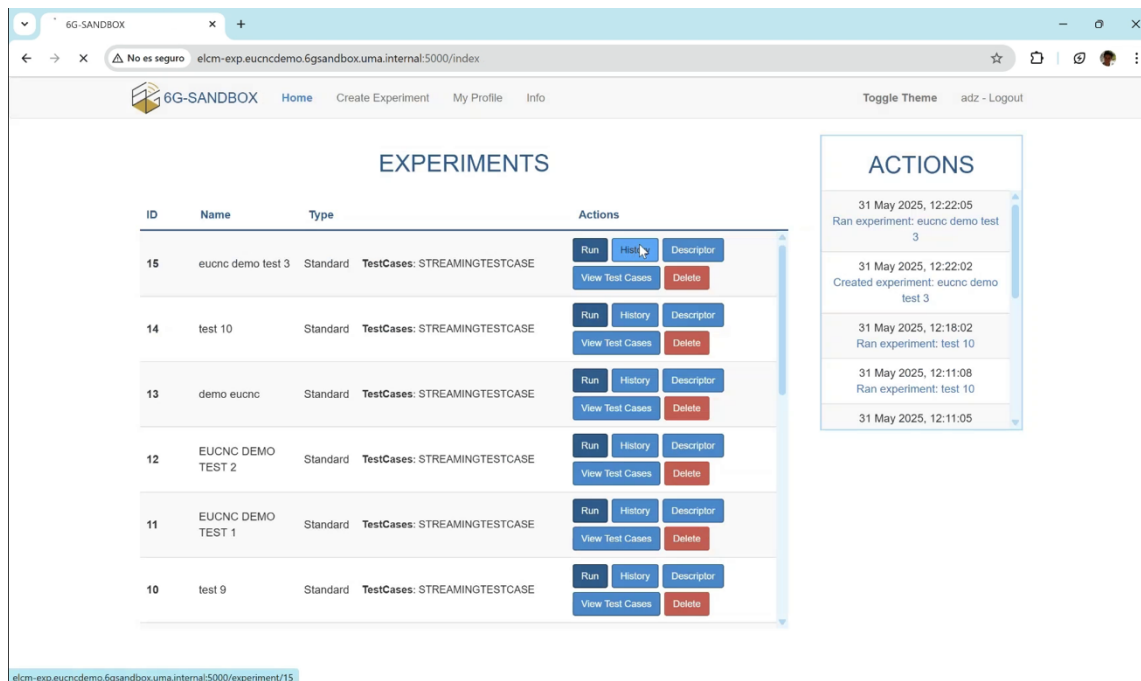
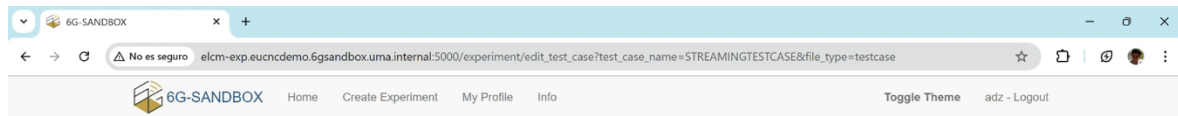


Figure 8: ELCM graphical interface for the execution of experiments



Edit Testcase: "STREAMINGTESTCASE"

```

3  Version: 2
4  Name: STREAMINGTESTCASE
5
6  Sequence:
7  - Order: 1
8  Task: Flow.Parallel
9  Children:
10 # Recolección de métricas Prometheus
11 Task: Run.PrometheusToInflux
12 Config:
13 ExecutionId: "@(ExecutionId)"
14 Uri: "192.168.199.8"
15 Port: "9090"
16 QueriesRange:
17 - rtsp_sessions_rtp_packets_jitter
18 - rtsp_conns
19 - rtsp_sessions_bytes_received
20 - rtsp_conns_bytes_sent
21 - rtsp_sessions
22 - rtsp_sessions_bytes_received
23 - rtsp_sessions_rtp_packets_received
24 - rtsp_sessions_rtp_packets_sent
25 - rtsp_sessions_rtp_packets_lost
26 - rtsp_sessions_rtp_packets_in_error
27 Measurement: PROMETHEUS
28 Step: "1s"
29 Stop: "stop_prometheus1"
30 Account: False
31 Encryption: False
32
33 - Task: Run.PrometheusToInflux
34 Config:
35 ExecutionId: "@(ExecutionId)"
36 Uri: "prometheus-sandbox.uma.internal:9090"

```

Figure 9: Editing the tasks within an ELCM test case

In this context, a test case is essentially a collection of tasks grouped under the same Execution ID. Based on the tasks defined in the test case, the ELCM generates a sequence of actions that make up the experiment—this process is referred to as Experiment Composition. These tasks will have links and specific instructions for components within the given trial network.

There are two types of tasks that can comprise a test case:

- **Top-Level Tasks:** These are defined directly within the first level of indentation of the test case. They are executed in the order specified by their Order field and always run sequentially.
- **Child Tasks:** These are defined under the Children field of a Top-Level or another Child Task. They do not have an Order field of their own; instead, their execution is controlled by the logic of their parent task.

The composition process only applies to Top-Level Tasks; with each being treated as an indivisible unit along with its tree of child tasks. During composition, all task definitions from the UEs and test cases included in the experiment descriptor are gathered into a single ordered list, sorted by the Order field of each Top-Level Task. Top-Level Tasks always execute sequentially, similarly to the behavior of conventional programming languages. However, Flow Tasks can be used to alter the control flow of child tasks. The repository for ELCM contains a full list of Flow Tasks, General-purpose tasks and guidance on how to implement new tasks [12] [13] [14].

If a Grafana [15] instance is available and configured, the ELCM can automatically create a Dashboard for displaying some of the most important raw results generated during an experiment execution. In order to use this functionality, the test case definition must include a collection of Grafana panel definitions. For each experiment execution, the panels defined by all the test cases selected will be aggregated in a single dashboard. The repository features instructions for the definition of a dashboard for a given test case [16]. It is also possible to download full reference test cases either for direct use or for use as a basis for defining new ones [17].

4.2 RUNNING EXPERIMENTS WITH OPENTAP CAMPAIGN MANAGER

After the setup of the experiment environment as described in Section 3, one of the methods through which test plans/campaigns can be executed in the environment is through the use of the OpenTAP Campaign Manager Tool [11]. This is a tool developed around the OpenTAP open-source project [18], with a significant amount of development coming from Keysight Technologies. OpenTAP Campaign Manager provides a simple interface and implementation that allows for the creation, running and monitoring of tests featuring multiple disparate tools and features.

The Campaign Manager has 3 main structural components. The first of these is the Runner, which is a piece of software that has to either be present in the experiment environment or have access to it over some sort of network connection. In the case of 6G-SANDBOX, the Runner can be deployed as a component in the Trial network or be permanently present within the infrastructure of a given platform. This Runner receives instructions on what needs to be performed in order to enable test execution. This can range from the activation of testing tools, modification of network parameters, exporting results to internal tools for further processing, and a vast variety of other options. A runner can run multiple different test campaigns at once, as long as the resources its accessing are available.

The runner receives its instructions from the second of the main components, the Campaign Manager interface. From this interface, an experimenter can create, run and monitor the execution of a Test Campaign. A test Campaign is comprised of multiple Test Plans, each corresponding to the generation of a specific KPI or set of KPIs. Each Test Plan in turn is made up of a set of Test Steps, which are the individual steps that need to be performed in order to successfully execute the test. Test Campaigns and Plans can also be saved for future use, further simplifying the process of interacting with the tool for those that are not extensively familiar with it. In the Campaign Manager, a Test Campaign can be assigned to a Runner that has been registered to a user group the experimenter has access to. Upon the initiation of the Test Campaign, all the instructions are sent to the Runner at once, which means the Runner can continue execution even if connection to the Campaign Manager is lost. The experimenter can monitor the progress of the Test Campaign from within the Campaign Manager Interface.

The final piece of the puzzle is the central OpenTAP repository [19]. This is not hosted within 6G-SANDBOX and is not even a part of the project but is instead a publicly hosted repository containing a variety of plugins for use with OpenTAP. The Campaign Manager and the Runner do not have inherent knowledge about any given testing tool – the commands, functions and Test Steps for interfacing with tools are stored in plugins in this repository and the Runner fetches whichever plugins are marked as necessary once a Test Campaign is sent to the Runner and run.

Figure 10 shows the main test execution interface of the OpenTAP Campaign Manager. On the left side of the screen, the list of Test Plans included in this Test Campaign is visible, each named after the result they are testing for. The first of these Test Plans has been expanded, showing the individual Test Steps that comprise the Test Plan. In the middle, the monitoring system for monitoring the progress of the Test Campaign is visible, which can show the progress per individual Test Step. On the right, the parameters for the Test Campaign, editable or otherwise, are displayed.

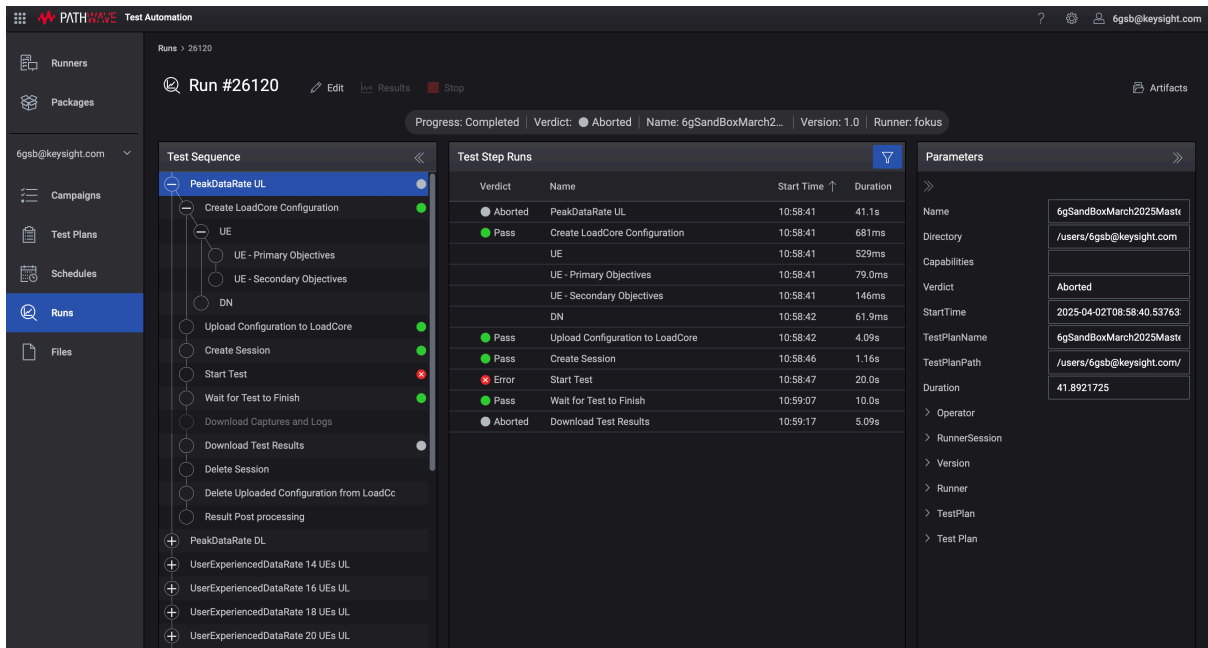


Figure 10: The main test execution interface of the OpenTAP Campaign manager

5 POST-PROCESSING

5.1 GENERAL POST PROCESSING METHODOLOGY

Running of experiments within 6G-SANDBOX generates vast amounts of data, but for the data to be translated into KPIs, stored, used and shared for the purposes of furthering research it must be post processed. Following the best practice of experiment execution within 6G-SANDBOX, this is also a process that is automated to the largest extent possible. As described in Section 4, experiment execution in 6G-SANDBOX, be it through ELCM or the OpenTAP Campaign manager, follows a format of test steps, combined into test plans, which come together to create a full experiment execution. Post processing follows the same principles, being comprised of test steps or combinations of test steps that can either be included within the original test campaign or run after the fact.

Data generated by test instruments within 6G-SANDBOX most often takes the form of large, keyed databases, which are most often produced as .csv files. These files usually represent the measurement of a given value at consistent time intervals over the course of the execution time of a given test plan. While this can obviously be graphed to provide information about the process of the measured value over time, more often than not further work is required to extract KPIs from this data. 6G-SANDBOX Deliverable 2.1 [20] contains details of the specific definitions of these KPIs, including the equations and tertiary data that is required to calculate them. To leverage this information, specific code exists for each KPI that takes the generated .csv file as an input and outputs the calculated KPI. Due to the simplicity of these operations this code can be quite flexible in terms of language and implementation. For example, one implementation involves writing said code in Python [21] and running it on the OpenTAP Runner used through the OpenTAP Campaign Manager experiment execution. The implementation of these post processing scripts is the responsibility of the automation developers within 6G-SANDBOX.

The time taken to post process data and generate KPIs varies heavily, depending on the complexity of operation required for KPI generation and the amount of data being analyzed. Some KPIs for example simply consist of the code finding the last value or the highest value in the sequence, while others involve the data being ordered based on some given parameters, and then the corresponding value being selected based on a predefined function. As mentioned previously, these post processing steps can be run as part of the test campaign execution, or afterwards on preexisting data. The results of these post processing steps can be presented to the

experimenter in the form of a spreadsheet or .csv file, usually divided into individual sheets by KPI, ready for them to use or perform whatever further processing they may require.

At time of writing, examples of post processing can be seen in Deliverable D5.3 [22], but will also feature in the final Deliverable 5.5 on test results within 6G-SANDBOX [2].

5.2 PLATFORM SPECIFIC TESTING POST PROCESSING METHODOLOGY

Some testing campaigns or operations are not available at the 6G-SANDBOX level and instead are specific to each platform. Such testing campaigns usually relate to platform specific capabilities that are not shared by the other platforms and also feature highly customized test steps, network setups, and result formats. As such, the post processing steps and methodology for these tests can vary significantly from the norm described previously, as they are up to the discretion of each platform for any given test that they run. At time of writing, no examples can be given, but it is worth noting that 6G-SANDBOX does provide the space and flexibility for these more “out of the ordinary” testing and processing schemes.

6 DATA COLLECTION AND STORAGE

The final aspect of the experiment methodology within 6G-SANDBOX is the collation of the generated and processed data for the purposes of holistic presentation to the experimenter and long-term storage for the benefit of future experimenters and the research community at large.

6.1 METHODOLOGY

The methodology for data collation has been described in Deliverable D2.3 [23]. An InfluxDB [24] is deployed as a component inside of a Trial Network, where every measurement is stored along with a minimum set of metadata. This approach allows for filtering, parsing and generally processing the experimental data for given research. In addition, should an experimenter decide to share their results with a wider community, all part of the data contained in a Trial Network can be coupled with a Data Descriptor, where the experimenter can declare additional metadata for the results, and then the data can be exported to the Platform Central Repository.

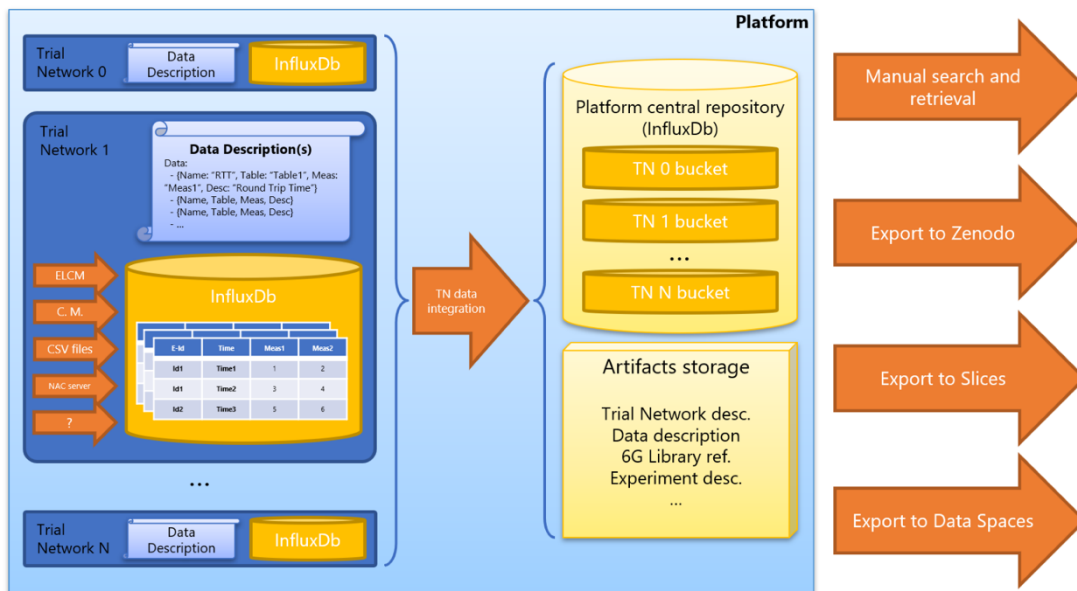


Figure 11: Data reusability in 6G-SANDBOX

The Platform Central Repository is itself another InfluxDB database, ensuring direct compatibility. This database stores a curated sub-set of all the measurements generated in an experimentation platform. The data contained

in the Platform Central Repository are structured in a specific pre-determined manner, include description on basic processing and handling. This allows them to be accessible through the use of simple APIs.

Additionally, other useful artefacts, such as the original Trial Network Descriptor or references to used components can be stored in the centralized Artefacts Storage. This central storage provides longer term storage that is not dependent on the existence of the original Trial Network and allows experimenters to continue processing the data without consuming resources in the platform that could be useful for other experimenters.

6.2 METADATA HANDLING

When committing data to long term storage, a common structure for metadata is required to make sure that all reported data is re-usable from future experimenters and well-documented. The *Test Data reusability sub-group* within the SNS-JU TMV has developed a common metadata template (CMT) with major contributions from the 6G-SANDBOX project. This template has been agreed among all the SNS-JU projects and has been used as a blueprint for the development of a central Metadata Registry System (MRS) for the SNS-JU projects. The 6G-SANDBOX project has offered its facilities to temporarily host (until the end of the project's lifetime) metadata from all the SNS-JU projects.

The proposed CMT structure is based mainly on what existing Metadata registry systems request during an addition of a new record (SLICES RI MRS [25], Zenodo [26], OpenAIRE [27], DataCite [28] etc.), and also what bibliographies include as vital information to best represent results coming from end-to-end experimentation processes (e.g., the ETSI TR 103 761 V1.1.1 [29] and 3GPP TS 28.552 [30])

In the proposed CMT, three main groups of information are requested, referring to:

- Dataset Identity (Basic fields regarding the ownership and the identity of the Dataset)
- Dataset Object Characteristics (Information on the Dataset model, size, availability etc.)
- Dataset Content Information (Dataset content description and its production process)

As depicted in the figure below, the CMT is used by the SNS-JU projects to store their metadata to a single reference Metadata Repository System, while the actual data can be stored at any repository (decided by the SNS-JU project) as long as the Metadata Repository System links to it.

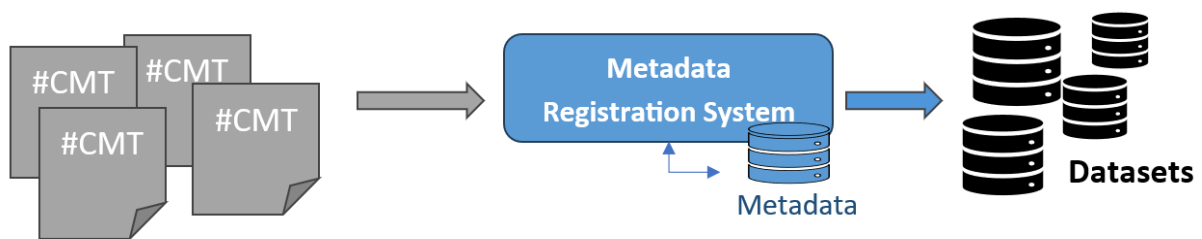


Figure 12: A Common metadata template is used to store metadata to a single Metadata Repository System for all SNS-JU projects

7 CONCLUSION

6G-SANDBOX is a large project with a lot of moving parts, and thus methodologies used within facilitating its key functionalities must be defined clearly. With the definition explored in this Deliverable, 6G-SANDBOX can provide experimenters with a consistent albeit flexible method of interacting with the infrastructure, facilitating them to perform research in the 6G space at a greater scale but at a significantly reduced effort on their part than would be expected otherwise. This Deliverable also serves as a guide for any experimenters looking to perform research within the 6G-SANDBOX framework. While the details of every experiment may differ, the information represented here will always be at the core of the experimentation process within 6G-SANDBOX.

8 REFERENCES

- [1] 6G-SANDBOX, "Deliverable D4.1 Integration and Management for 6G Trial Networks -Intermediate report," 2024. [Online]. Available: https://6g-sandbox.eu/wp-content/uploads/2024/02/6G-SANDBOX_D4.1_v1.0.pdf. [Accessed 17 6 2025].
- [2] 6G-SANDBOX, "6G-SANDBOX Deliverables," [Online]. Available: <https://6g-sandbox.eu/dissemination/deliverables/>. [Accessed 17 6 2025].
- [3] 6G-SANDBOX, "TNLCM Trial Network life Cycle Manager," 16 5 2025. [Online]. Available: <https://github.com/6G-SANDBOX/TNLCM>. [Accessed 17 6 2025].
- [4] 6G-SANDBOX, "6G-Library," [Online]. Available: <https://github.com/6G-SANDBOX/6G-Library>. [Accessed 17 6 2025].
- [5] Ansible Community Documentation, "Ansible Playbooks," [Online]. Available: https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_intro.html. [Accessed 26 6 25].
- [6] MinIO Inc, "MinIO | High Performance, Kubernetes Native Object Storage," [Online]. Available: <https://min.io/>. [Accessed 17 6 2025].
- [7] J.A.Donenfeld, "WireGuard: fast, modern, secure VPN Ttunnel," [Online]. Available: www.wireguard.com. [Accessed 17 6 2025].
- [8] Technitium, "Technitium - Push The Limits," 2024. [Online]. Available: <https://technitium.com>. [Accessed 17 6 2025].
- [9] EUCNC, "EUCNC and 6G Summit," [Online]. Available: <https://www.eucnc.eu>. [Accessed 26 6 25].
- [10] MORSE Research Group, "ELCM," 2025. [Online]. Available: <https://gitlab.com/morse-uma/elcm>. [Accessed 17 6 2025].
- [11] Keysight Technologies, "KS8400B Test Automation on PathWave," 13 5 2024. [Online]. Available: <https://www.keysight.com/gb/en/assets/7018-05485/technical-overviews/5992-1909.pdf>. [Accessed 17 6 2025].
- [12] MORSE Research Group, "FLOW_TASKS.md," 8 5 2025. [Online]. Available: https://gitlab.com/morse-uma/elcm/-/blob/main/docs/3-1b_FLOW_TASKS.md. [Accessed 17 6 2025].
- [13] MORSE Research Group, "GENERAL_TASKS.md," 22 5 2025. [Online]. Available: https://gitlab.com/morse-uma/elcm/-/blob/main/docs/3-2a_GENERAL_TASKS.md. [Accessed 17 6 2025].

- [14] MORSE Research Group, "TASK_IMPLEMENTATION.md," 3 10 2022. [Online]. Available: https://gitlab.com/morse-uma/elcm/-/blob/main/docs/3-5_TASK_IMPLEMENTATION.md. [Accessed 17 6 2025].
- [15] Grafana Labs, "Grafana Documentation," [Online]. Available: <https://grafana.com/docs/grafana/latest/>. [Accessed 26 6 25].
- [16] MORSE Research Group, "DASHBOARD_PDF.md," 3 10 2022. [Online]. Available: https://gitlab.com/morse-uma/elcm/-/blob/main/docs/2-3_DASHBOARD_PDF.md. [Accessed 17 6 2025].
- [17] MORSE Research Group, "Samples," 2025. [Online]. Available: https://gitlab.com/morse-uma/elcm/-/tree/main/Samples?ref_type=heads. [Accessed 17 6 2025].
- [18] OpenTAP, "OpenTAP Documentation," 2025. [Online]. Available: <https://doc.opentap.io>. [Accessed 17 6 2025].
- [19] OpenTAP, "Package Repository," 2025. [Online]. Available: <https://packages.opentap.io>. [Accessed 17 6 2025].
- [20] 6G-SANDBOX, "Deliverable D2.1 Ecosystem Analysis and 6G-SANDBOX Facility Design," 2023. [Online]. Available: https://6g-sandbox.eu/wp-content/uploads/2023/07/6G-SANDBOX_D2.1_v1.5.pdf. [Accessed 26 6 25].
- [21] Python Software Foundation, "Python," 18 11 2019. [Online]. Available: <https://www.python.org>. [Accessed 17 6 2025].
- [22] 6G-SANDBOX, "Deliverable D5.3 Intermediate Report on KPI and KVI Validation," 2024. [Online]. Available: https://6g-sandbox.eu/wp-content/uploads/2025/04/6G-SANDBOX_D5.3_V1.pdf. [Accessed 17 6 2025].
- [23] 6G-SANDBOX, "Deliverable D2.3 Final Definition of Trial Network Life Cycle," 2025. [Online]. Available: https://6g-sandbox.eu/wp-content/uploads/2025/03/6G-SANDBOX_D2.3_v1.0.pdf. [Accessed 17 6 2025].
- [24] Influxdata, "Time series starts with InfluxDB," 2025. [Online]. Available: <https://www.influxdata.com>. [Accessed 17 6 2025].
- [25] SLICES-RI, "MRS (Metadata Registry System) - Slices-RI Documentation," 2022. [Online]. Available: <https://doc.slices-ri.eu/BasicServices/MRS/MRS.html>. [Accessed 17 6 2025].
- [26] Zenodo.org, "Zenodo - Research. Shared," [Online]. Available: <https://zenodo.org>. [Accessed 17 6 2025].
- [27] K. Iatropoulou, "OpenAIRE," 2025. [Online]. Available: <https://www.openaire.eu>. [Accessed 17 6 2025].
- [28] P. Vierkant, "DataCite - Connecting Research, Advancing Knowledge," 2025. [Online]. Available: <https://datacite.org>. [Accessed 17 6 2025].
- [29] ETSI, "Core Network and Interoperability Testing (INT); Methodologies for E2E Testing & Validation of Vertical Applications over 5G & Beyond networks," 5 2022. [Online]. Available: https://www.etsi.org/deliver/etsi_tr/103700_103799/103761/01.01.01_60/tr_103761v010101p.pdf. [Accessed 17 6 2025].



[30] 3GPP, "Management and Orchestration; 5G performance measurements," 24 03 2025. [Online]. Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3413>. [Accessed 17 06 2025].

9 ANNEX 1: FULL DEPLOYMENT REPORT OF TRIAL NETWORK “EUCNCDEMO”

6G SNS



Funded by
the European Union



6G-SANDBOX

Trial Network Report: eucncdemo

Date: 02/06/2025

1 INTRODUCTION

A Trial Network with the identifier "EUCNCDEMO" was created as of June 2, 2025 on 6G-SANDBOX site UMA (University of Malaga). Below you will find a summary of the components deployed or accessible from the Trial Network, and all the relevant information to access the virtual infrastructure as well as the URLs to access the services and their credentials, after connecting to the private VPN that provides access to the Trial Network.

2 COMPONENTS

2.1 EUCNC-TN_BASTION

The component `eucncdemo-tn_bastion` has been successfully created. The `tn_bastion` is the main Virtual Machine of the Trial Network, serving as its default gateway, firewall, DNS server and VPN server.

2.1.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3475
- **VM network interfaces:**

```
{  
  "11": "10.11.28.217",  
  "1275": "192.168.199.1"  
}
```

2.1.2 SSH KEYPAIR

A pair of `ed25519` ssh keys is created as part of the `tn_bastion`'s deployment. Use the private key for ssh access to other component's with user `tnuser`.

To use them, please write the private key on a new file on `.ssh/misc-eucncdemo-id_ed25519` and give it permissions `0600`.

You might incur ssh silent warnings if the permissions are not set. Debug ssh connections with flag `-v`.

The `tn_bastion` VM is not meant to be accessed by experimenters, so it has no `tnuser` created

Public key:

```
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIJfeKS8XCmkdpv2D8Yf7LzH+ctlwcozz8DVR/CorCrBg  
tnuser@eucncdemo
```

Private key:

```
-----BEGIN OPENSSH PRIVATE KEY-----  
b3B1bnNzaC1rZXktdjEAAAABAG5vbmUAAAABbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW  
QyNTUxOQAAACX3ikvFwppHab9g/GH+y8x/nLZcHKM8/A1a/wqKwqwYAAAAJhGz1FxFs5R  
cQAAAAtzc2gtZWQyNTUxOQAAACX3ikvFwppHab9g/GH+y8x/nLZcHKM8/A1a/wqKwqwYA  
AAAEADakz9y5ap1m0pwsYPJ5DACb4dDIZ9ar3qgrLJ9F15AZ5feKS8XCmkdpv2D8Yf7LzH+  
ctlwcozz8DVR/CorCrBgAAAAEHRudXN1ckBlbWNUy2R1bW8BAgMEBQ==  
-----END OPENSSH PRIVATE KEY-----
```

2.1.3 WIREGUARD VPN CLIENT CONFIG

Access to the Trial Network can be performed using one of the following Wireguard VPN client configuration files:

wg_client0:

```
[Interface]
Address = 10.7.0.2/24
DNS = 192.168.199.1
PrivateKey = i02faSV90TmhhcOF1vruhE6e1IgdLzkaDIei+DKaYF0=

[Peer]
PublicKey = 0WVB26G0XhIUXp3X89Dju3GjghLUw8vGYRzTMa8VEA0=
PresharedKey = bBSUFBXi47ahpRxL4aMe4o7gOgUXvv6qUxPYarmuWLG=
AllowedIPs = 192.168.199.0/24,10.7.0.0/24,10.21.12.0/24,10.45.0.0/16
Endpoint = 150.214.47.156:39817
PersistentKeepalive = 25
```

wg_client1:

```
[Interface]
Address = 10.7.0.3/24
DNS = 192.168.199.1
PrivateKey = MGRoMfMtOPkhXDKw2pgMa4L+cPQUvAWG7akRJKckzFI=

[Peer]
PublicKey = 0WVB26G0XhIUXp3X89Dju3GjghLUw8vGYRzTMa8VEA0=
PresharedKey = ZT+s+StLJ7U9uNHjiulrPddNgGc6zGajMSAcmmGfJjs=
AllowedIPs = 192.168.199.0/24,10.7.0.0/24,10.21.12.0/24,10.45.0.0/16
Endpoint = 150.214.47.156:39817
PersistentKeepalive = 25
```

wg_client2:

```
[Interface]
Address = 10.7.0.4/24
DNS = 192.168.199.1
PrivateKey = 0Gg5MIstsvwQjzvVFTbvhhRY7opFRNsEPGkpDnRUEk=

[Peer]
PublicKey = 0WVB26G0XhIUXp3X89Dju3GjghLUw8vGYRzTMa8VEA0=
PresharedKey = T7mfm61er7Gglsl74KGk8J4pV6UvNzwe2AIMx5xDhkY=
AllowedIPs = 192.168.199.0/24,10.7.0.0/24,10.21.12.0/24,10.45.0.0/16
Endpoint = 150.214.47.156:39817
PersistentKeepalive = 25
```

wg_client3:

```
[Interface]
Address = 10.7.0.5/24
DNS = 192.168.199.1
PrivateKey = 0Ekzja/P1emyeJxwQK/0I9kL03FJbN5+8+75oe7pInU=

[Peer]
PublicKey = 0WVB26G0XhIUXp3X89Dju3GjghLUw8vGYRzTMa8VEA0=
PresharedKey = pccyEPOexEbhsxxyvPnuSx7c5EMjNmQe182MQ04uUxA=
AllowedIPs = 192.168.199.0/24,10.7.0.0/24,10.21.12.0/24,10.45.0.0/16
Endpoint = 150.214.47.156:39817
PersistentKeepalive = 25
```

wg_client4:

```
[Interface]
Address = 10.7.0.6/24
DNS = 192.168.199.1
PrivateKey = +GjzXQB8dgg32SHR0IP9LnLPVWe5GhLs1xhz/VG0QVk=

[Peer]
PublicKey = 0WVB26G0XhIUXp3X89Dju3GjghLUw8vGYRzTMa8VEA0=
PresharedKey = xR4TLJPkgw8qT7pMFGBS0K+Aq9EHZE0kl0oNgFcnjS0=
AllowedIPs = 192.168.199.0/24,10.7.0.0/24,10.21.12.0/24,10.45.0.0/16
Endpoint = 150.214.47.156:39817
PersistentKeepalive = 25
```

WARNING

Only one experimenter can use the same VPN client simultaneously.

2.1.4 TECHNITIUM DNS SERVER

You can access the Technitium DNS web portal

from <http://bastion.eucncdemo.6gsandbox.uma.internal:5380>

Credentials to login are:

- user: admin
- password: IOr1aPzAYjlzfdR1m3KMENVmmK2YZUDmi3urgRn1n5w=

2.2 EUCNCDEMO-TN_VXLAN

The component eucncdemo-tn_vxlan has been successfully created.

The **tn_vxlan** is the main subnetwork of the Trial Network, connecting all components, and reachable to the experimenter through the Wireguard VPN server in the **tn_bastion**.

2.2.1 IMPORTANT INFORMATION:

- **OpenNebula VNet ID:** 1275
- **VXLAN subnet:** 192.168.199.0/24
- **VXLAN first IP:** 192.168.199.1
- **VXLAN address size:** 254
- **VXLAN netmask:** 24
- **VXLAN gateway:** 192.168.199.1
- **VXLAN DNS:** 192.168.199.1
- **VXLAN MTU:** 1500
- **VXLAN guest MTU:** 1450

2.3 EUCNCDEMO-VNET-N2

The component `eucncdemo-vnet-n2` has been successfully created. It works as a secondary subnet inside your Trial Network components

2.3.1 IMPORTANT INFORMATION:

- **OpenNebula VNet ID:** 1276
- **VXLAN subnet:** 10.21.12.0/24
- **VNET first IP:** 10.21.12.1
- **VNET address size:** 100
- **VNET netmask:** 24
- **VNET gateway:** None
- **VNET DNS:** None
- **VNET MTU:** 1500
- **VNET guest MTU:** 1450

2.4 EUCNCDEMO-ONEKE-CLUSTER

The component `eucncdemo-oneke-cluster` has been successfully created.



The Kubernetes cluster has been deployed as a OneKE Service, with the ***Skooner Dashboard*** helm chart already deployed. OneKE is a minimal hyperconverged Kubernetes platform that comes with OpenNebula out of the box. It is based on RKE2 - Rancher's Next Generation Kubernetes Distribution with preinstalled components to handle persistence, ingress traffic, and on-prem load balancing.

OneKE Service has four different Roles:

- **VNF:** Load Balancer for Control-Plane and Ingress Traffic.
- **Master:** Control-Plane nodes
- **Worker:** Nodes to run application workloads
- **Storage:** Dedicated storage nodes for Persistent Volume replicas

pOZWtsNVRrUkNXazFDVfVQ2VYRkhDbE5OTkRsQlowVkhRME54UjFOTk5EbEjKMFZJ
UVRCSIFVsktjVXRxU2xkcFdFaDRiVzFKVG05YWMYWTNkQ3RQUzBaWfJrMUtSM1JaSz
A4cmRUSlJiVm9LZFRaTFNWbDVRVXBrTWpKWIFVaERibWxGU3psT1drY3hlakZvWm5
aVVRUaEhRMDVNT1RaRU56SIFhMm92TVN0cVVXcENRVTFCTkVqK1WVmtSSGRGU
WdvdMqXrKzRWGRKUTNCRVFWQkNaMDVXU0ZKTIFrRm1PRVZDVkVGRVFWRkIMM
DFDTUVkQk1WVmtSR2RSVjBKQ1V6QkNNbTVyTjFGM2IwMVINVEZCYzFCRkNrMVhjV
Gx3VjNabFFucEJTMEpuWjNGb2EycFBVRkZSUKVgblRraEJSRUgGUVdsQ1IwRk9MeIJW
WnpscmJYWXISRWxvZUZkV2RURnIXRkpEUm1oaVdEZ0tXRGxTYkUxVlpGZzVjV0p2WjJ
kSlowTIBXWGxaVkZZNE5rMUhZM2RwV1dVeFNHdHJkV1JOY1dGaVFqaDNWV015ZFU
xTE5rMHhMMnQzTURROUNpMHRMUzB0UIU1RUIFTkZVbFJKUmtsRFFWUkZMUzB0T
FMwSwogICAgc2VydmVyoibodHRwczovLzE5MmI4xNjguMTk5LjM6NjQ0MwogIG5hbW
U6IGRIZmF1bHQKY29udGV4dHM6Ci0gY29udGV4dDoKICAgIGNsdxN0ZXI6IGRIZmF1b
HQKICAgIHVzZXI6IGRIZmF1bHQKICBUYW10iBkZWZhdWx0cmN1cnJlbnQtY29udGV4
dDogZGVmYXVsdApraW5kOiBDb25maWcKcHJIZmVvZjZ5jZXM6IHt9CnVzZXJzOgotIG
5hbWU6IGRIZmF1bHQKICB1c2VyOgogICAgY2xpZW50LWNlcnRpZmljYXRILWRhdGE6I
ExTMHRMUzFDUIVksIRpQkRSVkpVU1VaSEwRIVSUzB0TFMwdENrMUUpTVUppyYWtOR
FFWUnBaMEYzU1VKQlowbEpUbU5DV0ZWQk1FTm5jVUYzUTJkWINVdHZXa2w2YWp
CRIFYZEpkMHBFUldsTlEwRkhrVEZwUUVGM2Qxb0tZMjEwYkUxcE1XcGISMnhzWW01
UmRGa3ISa0ZOVkdNd1QwUkpNazE2U1hsT1JFRmxSbmN3ZVU1VVFURk5hbGw0Vfd
wUmQwMXFVbUZHZHpCNVRtcEJNUXBOYWxsNFRXcFJkMDFxVW1GTIJFRjRSbnBCVm
tKblRsWkNRvzIVUkc1T05XTXpVbXhpVkhCMFdWaE9NRnBZU25wTIVsvjNSWGRaUkZ
aUIVVUkZkM2g2Q21WWVRqQmFWekEyV1ZkU2RHRlhOSGRYVkvGVVfTzGpjV2hyYW
s5UVVbENRbWRuY1docmFrOVFVTFDUW5kT1EwRkVbTV2VUhc1dHeE1kM3A0T
UVrS1pYcDBUV3Q2YTBKaVJtSIRWSFJVY0hsbGFFWkNjR1ZFY0ZOYVZWUmtTbTk0WkR
kWIRWRnFhbmN6ZW1aR2MzVndjRVJaVkvVWtk1Yb3ZhVmRYTVdzMmNBcDBTbk5OY
m1NcmNtOHdaM2RTYwTGUffTZE9Wa2hST0VKQlppqaEZRaoZOUTBKafFYZEzKMWxF
VmxJd2JFSkJKM2REWjFsSImzZFPa0pSVIVoQmQwbDNDa2gzV1VSV1VqQnFRa0puZD
BadlFWVkljMXBGUmpCVFFrTkhvVmhNUjJsaGRuWkpablEwZHpnNFIZDNRMMRaU1V
OdldrbDZhakJGUvhkSIJGTkJRWGNLVWxGSloxbGIWVTFtZVdwNk5TdFhNbkZITVRCW
WRERTFkbmQ0VnpCdWNFNvIXV0pyTIVZNVdHMURNM2hrUjBWRFNWRkRPSFk0WIZ
Oak5IWXJSVIVyVkfVeU1rUkdRbFZUYU1RFNDOHIVWGx0YjFSVWFraG1WelF6YTBOe
IRIYzIQUW90TFMwdExVvK9SQ0JEUZKVVNVWkpRMEZVUIMwdExTMHRDaTB0TFMw
dFFrVkhTVTRnUTBWU1ZFbEdTVU5CVkVdExTMHRMUXBOU1VsQ1pWUkRRMEZTSzJ
kQmQwbENRV2RKUWtGRVfVdENaMmR4YUd0cVQxQIJVVVJCYWtGclRWTkpkMGxCV
1VSV1VWRkVSRUpzZVdFeVZYbE1WMDV6Q21GWFZuVmtRekZxV1ZWQmVFNTZVVFJ
OYWxsNIRXcEpNRTFDtkZoRVZFa3hUVVJWZVU1cVJYbE9SRUY1VGtadldFUIVUVEZOU
kZWNVRrUkZIVTVFUVhrS1RrWnZkMHBFUldsTlEwRkhrVEZwUUVGM2QxcGpiWFJzVF
dreGFtSkhiR3hpYmxGMFDUSkdRVTFVWXpCUFJFa3IUWHBKZVU1RVFscE5RazFIUW5s
eFJ3cFRUVE1UVdkRIlwTkrjVWRUVFRRNVFYEZTRUV3U1VGQ1NVUXhXa1ZQZURJM
FNVSkipVSFzZWWt3NU5HeHZObHBtTW00MfIXRk1IbmMxY1dGWIJXWTNDbXh6Vm5
WdWvtZExkaZxVTFGaFrUnZXRE4xVEM5aFZWTTBSR0pOWkZBMWRYaHhXR1JvVvZ
WelITdHFTbGRYYWxGcVfrRk5RVFJIUVRGVIpFUjNSVUILTDNkUIJVRjNTVU53UkVgUVF
tZE9Wa2hTVFVKQlppqaEZRbFJCUkVGUINDOU5RakJIUVRGVIpFUm5VvMrdUWxGbGV
HdFJXRkpkUIVseFNtTnpZVXB4S3dvNGFDc3pha1I2ZDJKRvFVdENaMmR4YUd0cVQxQI
JVvVJcWjA1SIFVukNSa0ZwUWtVM1ZYTnZNRikxYkVJMU1XbDVUVGRyY0V4c1N6WIB
iM1pyYUVSTUNrb3dNbXQ0VIZOcWRXeExaVVSU1doQINvczBNM2xVU1ZkblRVOHIN
REJQTm5SeGNFUnFRWGXyZDJjemQwVIRZV3QyT0VsUmRsTmljRzFsU2dvdExTMHRM
VZPUkNCRFJWSIVTVVpKUTBGVVJTMHRMUzB0Q2c9PQogICAgY2xpZW50LWtleS1kY

```
XRhOiBMUzB0TFMxQ1JVZEpUaUJGUXICUVVrbFdRVIJGSUV0RldTMHRMUzB0Q2sxSV
kwTkJVVVZGU1UxblRuUTBUVzR5ZGpVNFpYm5ZekJYUhcCmQzUkRaMjk1TWxabm
QzZHpOM0ZYTWxOc2NXeGtUVXB2UVc5SFEwTnhSMU5OTkRrS1FYZEZTRzIwVVVSUI
owRkZXalpFT0VwV05WTTTRUVGhrUTBoek4xUktUVFZCVjNoWE1HczNWVFPqYm05U1
VXRllaelpWYlZaRk0xTmhUVmhsTWtSRINRbZBPRTQ0TTNoaVRIRmhVVEpGZUVST1I5O
DBiR3gwV2s5eFlsTmISRW96VUhgM1BUMEtMUzB0TFMxRIRrUWdSVU1nVUZKSIZrRI
VSU0JMUIZrdExTMHRMUW89Cg==
```

- Node IPs:

```
{
  "master_0": "10.21.12.2",
  "storage_0": "10.21.12.4",
  "storage_1": "10.21.12.5",
  "storage_2": "10.21.12.6",
  "vnf_0": "192.168.199.3",
  "worker_0": "10.21.12.3"
}
```

If you need detailed information about OneKE you can visit the official documentation:
https://github.com/OpenNebula/one-apps/wiki/oneke_ops#operating-oneke

The validity of the Skooner Dashboard token is ten years from now. If you need to create another one, use this command:

```
kubectl create token skooner-sa -n kube-system --duration=8760h
```

More info here: <https://github.com/skooner-k8s/skooner>

2.4.2 HOW TO DEBUG ONEKE

In case access to the Kubernetes cluster via the kubeconfig file is not enough to troubleshoot issues, you might need to access OneKE's nodes via ssh. To do so, please paste the content of file `misc-eucncdemo-ssh_config` into your current ssh configuration.

```
ssh eucncdemo-oneKE-cluster-vnf_0
ssh eucncdemo-oneKE-cluster-master_0
ssh eucncdemo-oneKE-cluster-worker_0
ssh eucncdemo-oneKE-cluster-storage_0
ssh eucncdemo-oneKE-cluster-storage_1
ssh eucncdemo-oneKE-cluster-storage_2
```

2.5 EUCNCDEMO-OPEN5GS_K8S-CORE

The component eucncdemo-open5gs_k8s-core has been successfully created.



Open5GS is a C-language Open Source implementation for 5G Core and EPC, i.e. the core network of LTE/NR network (Release-17)

Deployed Open5GS includes the following 5G Core components:

- NRF - NF Repository Function
- SCP - Service Communication Proxy
- AMF - Access and Mobility Management Function
- SMF - Session Management Function
- UPF - User Plane Function
- AUSF - Authentication Server Function
- UDM - Unified Data Management
- UDR - Unified Data Repository
- PCF - Policy and Charging Function
- NSSF - Network Slice Selection Function
- BSF - Binding Support Function

If you need detailed information about Open5GS you can visit the official documentation: <https://open5gs.org/open5gs/docs/guide/01-quickstart/>

Current versions:

- Open5GS: v2.7.2
- Helm Chart used: v2.2.6

2.6 IMPORTANT INFORMATION

- **Kubernetes Cluster:** oneKE-cluster
- **PLMN ID (MCC):** 999
- **PLMN ID (MNC):** 70
- **TAC:** 1
- **S-NSSAI (SST):** 1
- **S-NSSAI (SD):** 000001
- **AMF N2 IP:** 10.21.12.200
- **UPF N3 IP:** 10.21.12.201
- **Open5GS Dashboard URL:** <https://open5gsk8s-core.eucncdemo.6gsandbox.uma.internal>
- **Web portal credentials:**
 - admin
 - 1423

2.7 EUCNCDEMO-VM_KVM-UBUNTU22_04

The component `eucncdemo-vm_kvm-ubuntu22_04` has been successfully created.



It consists of an Ubuntu 22.04 LTS Virtual Machine

2.7.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3476
- **VM memory:** 4096 MiB
- **VM VCPUs:** 2
- **VM available storage:** 25 GiB
- **VM network interfaces:**
{'1275': '192.168.199.2'}

2.8 EUCNCDEMO-UERANSIM-BOTH

The component `eucncdemo-ueransim-both` has been successfully created.



UERANSIM is a simulator for both an UE and/or a RAN (gNodeB). Both utilities are separated into two different services, stopped and disabled by default:

- **ueransim_gnb.service:** gNB simulator service defined at `/etc/systemd/system/ueransim_gnb.service`. Its behaviour is defined through the `/etc/UERANSIM/open5gs-gnb.yaml` file
- **ueransim_ue.service:** User Equipment simulator service defined at `/etc/systemd/system/ueransim_ue.service`. Its behaviour is defined through the `/etc/UERANSIM/open5gs-ue.yaml` file

If you need detailed information about what each parameter in the input files mean, check the [official documentation](#).

Component configuration logs can be found at `/var/log/post-ueransim.log`

2.8.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3486
- **VM memory:** 2048 MiB
- **VM VCPUs:** 2
- **VM available storage:** 10 GiB
- **VM network interfaces:**
`{'1275': '192.168.199.7'}`

2.8.2 GNB

Service `ueransim_gnb.service` is currently **running** and **enabled** with the following configuration:

- **Linked Open5GS:** `open5gs_k8s-core`
- **AMF IPv4 address:** `10.21.12.200`
- **TAC:** `999`
- **MCC:** `999`
- **MNC:** `70`
- **Slice SST:** `1`
- **Slice SD:** `000001`

2.8.3 UE

Service `ueransim_ue.service` is currently **running** and **enabled** with the following configuration:

- **Linked gNBs:** `localhost`
- **gNB Search List:** `localhost`
- **MCC:** `999`
- **MNC:** `70`
- **MSIN:** `0000000100`
- **Resulting SUPI:** `imsi-999700000000100`
- **Permanent Subscription Key:** `465B5CE8B199B49FAA5FOA2EE238A6BC`
- **Operator Code:** `E8ED289DEBA952E4283B54E88E6183CA`
- **Session APN:** `internet`
- **Session SST:** `1`
- **Session SD:** `000001`

2.9 EUCNCDEMO-ELCM-EXP

The component eucncdemo-elcm-exp has been successfully created.

2.9.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3485
- **VM memory:** 2048 MiB
- **VM VCPUs:** 2
- **VM available storage:** 7 GiB
- **VM network interfaces:**
{'1275': '192.168.199.6'}

2.9.2 ELCM BACKEND

The backend dashboard is available on port 5001.

<http://elcm-exp.eucncdemo.6gsandbox.uma.internal:5001>

2.9.3 ELCM FRONTEND

The frontend dashboard is available on port 5000.

<http://elcm-exp.eucncdemo.6gsandbox.uma.internal:5000>

Information to access to the frontend dashboard:

Registration is required to access the frontend dashboard in the following link:

<http://elcm-exp.eucncdemo.6gsandbox.uma.internal:5000/auth/register>

Fill the form with your information.

Login with your credentials in the following link:

<http://elcm-exp.eucncdemo.6gsandbox.uma.internal:5000/auth/login>

2.10 EUCNCDEMO-PROMETHEUS-METRICSERVER

The component eucncdemo-prometheus-metricserver has been successfully created.

2.10.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3487
- **VM memory:** 2048 MiB
- **VM VCPUs:** 2
- **VM available storage:** 7 GiB
- **VM network interfaces:**
{'1275': '192.168.199.8'}

2.10.2 PROMETHEUS V2.53.4

Prometheus is available on port 9090.

<http://prometheus-metricsserver.eucncdemo.6gsandbox.uma.internal:9090>

2.11 EUCNCDEMO-INFLUXDB-V2

The component eucncdemo-influxdb-v2 has been successfully created.

2.11.1 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3483
- **VM memory:** 2048 MiB
- **VM VCPUs:** 2
- **VM available storage:** 7 GiB
- **VM network interfaces:**
{'1275': '192.168.199.4'}

2.11.2 INFLUXDB V2.7.11 INFORMATION:

InfluxDB is available on port 8086.

<http://influxdb-v2.eucncdemo.6gsandbox.uma.internal:8086>

- **Username:** admin
- **Password:** adminadmin
- **Organization:** testing
- **Bucket:** testing
- **Token:** default-token-testing

2.12 EUCNCDEMO-GRAFANA-V11

The component eucncdemo-grafana-v11 has been successfully created.

2.13 IMPORTANT INFORMATION:

- **OpenNebula VM ID:** 3484
- **VM memory:** 2048 MiB
- **VM VCPUs:** 2
- **VM available storage:** 7 GiB
- **VM network interfaces:**
{'1275': '192.168.199.5'}

2.14 GRAFANA V11.6.0

Grafana is available on port 3000.

<http://grafana-v11.eucncdemo.6gsandbox.uma.internal:3000>

Information to access to Grafana dashboard:

- **Username:** admin
- **Password:** adminadmin