



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

[H2020 - Grant Agreement No.101016608]

Deliverable D4.6

# Network Apps for Security Guarantees and Risk Analysis

**Editor** George Kontopoulos (8BELLS)

**Contributors** Vasilis Pasios (8BELLS), Katerina Giannopoulou (FOGUS),  
Stavros-Anastasios Charismiadis (FOGUS), Ioannis  
Stylianou (IQBT)

**Version** 1.0

**Date** August 31<sup>st</sup>, 2023

**Distribution** PUBLIC (PU)



## 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 a 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
1.0	August 31, 2023	George Kontopoulos	

## LIST OF AUTHORS

<b>Partner ACRONYM</b>	<b>Partner FULL NAME</b>	<b>Name &amp; Surname</b>
<i>8BELLS</i>	<i>EIGHT BELLS</i>	<i>George Kontopoulos</i>
<i>8BELLS</i>	<i>EIGHT BELLS</i>	<i>Vasilis Pasios</i>
<i>FOG</i>	<i>FOGUS Innovations &amp; Services</i>	<i>Katerina Giannopoulou</i>
<i>FOG</i>	<i>FOGUS Innovations &amp; Services</i>	<i>Stavros-Anastasios Charismiadis</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Ioannis Stylianou</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Eleni Argyriou</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Constantinos Patsakis</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Dimitrios Dres</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Dimitrios Drakoulis</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Alexandra Dritsa</i>
<i>IQBT</i>	<i>InQbit Innovations</i>	<i>Ioannis Makropodis</i>

## GLOSSARY

<b>Abbreviations/Acronym</b>	<b>Description</b>
5GC	5G Core
5GS	5G System
CAPIF	Common API Framework
FoF	Factories of the future
IIoT	Industrial Internet of Things
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
MNO	Mobile Network Operator
NEF	Network Exposure Function
NPN	Non-Public Networks
NetApp	Network Application
OT	Operational Technology
PEI	Permanent Equipment Identifier
SDK	Software Development Kit
SME	Small and Medium Enterprises
SUPI	Subscription Permanent Identifier
vAPP	Vertical Application
UE	User Equipment

## EXECUTIVE SUMMARY

The objective of this deliverable is to present in detail the final prototypes and the two cycles of integration activities that have been followed for each of the three Network Applications by the three SMEs participating in the task.

Initially, the deliverable describes in detail the final prototypes of the Network Apps developed within the Security Guarantees and Risk Analysis pillar in the EVOLVED-5G context, driven by Task 4.4:

- **A Traffic Management Network App (8Bells):** Programmable Next-Generation Firewall that is capable of considering not only IPs, ports, and MAC addresses to identify user devices, but also any parameters that can be exposed by the 5G network, including subscriber location, thus allowing enhanced security policies.
- **A Security Information and Event Manager** based on blockchain technologies (FOGUS): security protection by performing real-time monitoring and analysis of events as well as tracking and logging of security data for compliance or auditing purposes.
- **Software for entity authentication, registration, and authorization (InQBit):** Enhancement to the existing CAPIF protocol by incorporating an OpenID Connect layer on top of the OAuth2.0 protocol proposed by CAPIF, also providing single sign-on functionality between CAPIF instances.

Next, the two development cycles (1st and 2nd iterations of integration activities) and use case testing that have been followed for each of the three Network Applications, are presented.

The first round of integrations has been carried out with the aim of ensuring seamless and reliable communication between various components within the system, including Network Apps, Vertical Apps (vApp), NEF, CAPIF and 5G network connectivity, on top of the cloud infrastructure provided by the Athens platform. The connectivity of 5G with the cloud infrastructure has been verified, specifically the connection between vApps and Demokritos' 5G network.

The purpose of the second integration round was to validate the use-cases utilizing the final components of EVOLVED-5G. On the one hand, NEF, CAPIF and the SDK had been enriched with additional features. On the other hand, SMEs finalized their Network Apps by enhancing the 3.0 version and using the last versions of NEF, CAPIF and SDK. This version 4.1 of the Network Apps also exploited the validation pipeline before the integration test. Finally, the Networks Apps were deployed in Kubernetes clusters in Athens premises instead of using Docker containers running locally.

With the second round of integration tests, the Networks Apps of the SEC pillar have reached their final stage, interacting with the last versions of NEF and CAPIF through the SDK and communicating with their respective vApp(s). The three SME use-cases have also been validated and such result highlight the fact that the Network Apps reached a mature enough state to be used by other SMEs through the Evolved-5G Marketplace.

For all three developed Network Apps, we have demonstrated that the integration with the 5G control plane enhances the functionality and usefulness of the applications in a NPN context, due to the additional parameters that can be exposed by the 5G network.

In addition, during the two rounds of integration activities that took place, it was also proven that the overall deployment is flexible, quick, easy to test, and can easily be upgraded, due to the dockerization of the Network Application and the Kubernetes deployment environment.

As a final point, in the context of EVOLVED-5G, it is essential to highlight that a terminology update has been implemented. Specifically, the term "Network App" is now being used instead of "NetApp," as initially selected in the first period of the project. This update reflects the shortened form of "Network Application" and has been applied consistently across all project's documents and materials.

## TABLE OF CONTENTS

1	INTRODUCTION .....	1
1.1	Purpose of the document .....	1
1.2	Structure of the document .....	1
1.3	Target Audience .....	2
2	PILLAR OVERALL FRAMEWORK .....	2
3	FINAL PROTOTYPE OF NETWORK APPLICATIONS .....	6
3.1	Traffic Management Network Application .....	6
3.1.1	Use case description .....	6
3.1.2	Detailed Architecture .....	6
3.1.3	Additional dependencies .....	8
3.2	5G SIEM Network Application .....	9
3.2.1	Use case description .....	9
3.2.2	Detailed Architecture .....	9
3.3	Identity and Access Management Network Application .....	15
3.3.1	Use case description .....	15
3.3.2	Detailed Architecture .....	16
3.3.3	Additional dependencies .....	18
4	INTEGRATION ACTIVITIES AND USE CASE TESTING .....	18
4.1	Purpose Of The Integration Tests (1 <sup>st</sup> Round) .....	18
4.2	Topology and Setup .....	19
4.2.1	Network App1: Traffic Management Network Application .....	19
4.2.2	5G SIEM Network Application .....	19
4.2.3	Network App3: Identity Management Network Application .....	22
4.3	Results and Takeaways .....	24
4.3.1	Traffic Management Network Application .....	24
4.3.2	5G SIEM Network Application .....	24
4.3.3	Identity and Access Management Network Application .....	26
4.4	Purpose Of The Integration Tests (2 <sup>nd</sup> Round) .....	27
4.5	Topology and Setup .....	27
4.5.1	Traffic Management Network App .....	27
4.5.2	5G SIEM Network Application .....	30
4.5.3	Identity and Access Management Network Application .....	33
4.6	Results and Takeaways .....	37
4.6.1	Traffic Management Network Application .....	37
4.6.2	5G SIEM Network Application .....	39
4.6.3	Identity and Access Management Network Application .....	41





5	Conclusion and next steps .....	47
6	Bibliography .....	48

# 1 INTRODUCTION

---

## 1.1 PURPOSE OF THE DOCUMENT

The current report compliments the final prototype of the three Network Applications, that have been developed within the Security Guarantees and Risk Analysis (SEC) pillar to support the security management and threat detection in a smart factory environment and is driven by Task 4.4. The report provides details on the development of the final prototype (version 4.1) in terms of technical architecture, features and dependencies, while also utilising the final version of the tools (SDK, NEF, CAPIF) developed within the EVOLVED-5G framework. Moreover, the report contributes to the testing and evaluation of the use cases, described in the previous deliverable of WP4 (D4.2), through the integration activities that took place both in Athens infrastructure focusing on the iterative validation of 5G connectivity and communication between components (5G network <--> Network Applications <-> Vertical Applications).

The two rounds of integration activities are described in depth and the use cases testing has been followed for each of the three Network Applications by the three SMEs participating in the task:

- Development of a Traffic Management Network App (Next-Generation Firewall, Virtual & Containerized) (8Bells)
- A security information and event manager based on blockchain technologies (FOGUS)
- Software for entity authentication, registration, and authorization (InQBit)

In summary, this document consolidates the progress made in developing, integrating and testing the Network Apps of the Security pillar within the EVOLVED-5G project.

## 1.2 STRUCTURE OF THE DOCUMENT

- Section 2 **“PILLAR OVERALL FRAMEWORK”** echoes D4.2 and presents a summary of the SEC pillar goals, challenges and specificities.
- Section 3 **“FINAL PROTOTYPE OF NETWORK APPLICATIONS”** describes the finalized version of the SEC Network Apps (version 4.1). After a reminder of the Network App use-case(s), it describes the technical architecture, features and dependencies of each Network App.
- Section 4 **“INTEGRATION ACTIVITIES AND USE-CASE TESTING”** presents the two integration rounds for use-case testing performed during the project. The first reported test was performed with intermediate versions of Network Apps and components (NEF, CAPIF and SDK) while the second test was performed with final versions of Network Apps and all EVOLVED-5G components.
- Finally, section 5 discusses the conclusion and next steps.

### 1.3 TARGET AUDIENCE

The release of the deliverable is public, intending to expose the overall EVOLVED-5G ecosystem and Network Apps progress to a wide variety of research individuals and communities.

From specific to broader, different target audiences for D4.4 are identified as detailed below:

- **Project Consortium:** To validate the fact that all SEC pillar Network Apps have reached their final state. One of the main goals is to document the technical evolution of these Network Apps with respect to the initial vision and use-case.
- **Industry 4.0 and FoF (factories of the future) vertical groups:** To crystallise a common understanding of technologies, and tools that were used for the development of the Network Apps. Besides, it also demonstrates the final architecture and features a Network App can reach. A non-exhaustive list of Industry 4.0-related groups is as follows:
  - Manufacturing industries (including both large and SMEs) and IIoT (Industrial Internet of Things) technology providers.
  - European, national, and regional manufacturing initiatives, including funding programs, 5G-related research projects, public bodies and policy makers.
  - Technology transfer organizations and market-uptake experts, researchers, and individuals.
  - Standardisation Bodies and Open-Source Communities.
  - Industry 4.0 professionals and researchers with technical knowledge and expertise, who have an industrial professional background and work on industry 4.0-related areas.
  - Industry 4.0 Investors and business angels.
- **Other vertical industries and groups:** To seek impact on other 5G-enabled vertical industries and groups in the long run. Indeed, all the architectural components of the facility are designed to secure interoperability beyond vendor specific implementation and across multiple domains. The same categorization as the above but beyond Industry 4.0 can be of application.
- **The scientific audience, general public and the funding EC Organisation:** To document the work performed and justify the effort reported for the relevant activities. The scientific audience can also get an insight of finalized Network Apps' processes, tools and features.

## 2 PILLAR OVERALL FRAMEWORK

---

Security Guarantees and Risk Analysis (SEC) is one of the four pillars in the EVOLVED-5G context.

Cybersecurity and Cyber Threats are identified as one of the top roadblocks to Industry 4.0 [1] As industries increasingly digitalize, a strong emphasis on cybersecurity has become critical for companies. Cyberattacks have caused operational hazards for companies making use of connected operational technologies, and security concerns have delayed many companies' move to the cloud. Companies must work to enact best practices with regards to networking

infrastructure and the deployment of operational technology (OT) and IoT cybersecurity features. Companies are increasingly turning to private networks, segregated and segmented networks, and zero trust models. However, many existing cybersecurity solutions focus on the security challenges facing IT, leaving a gap in the market for OT security. Furthermore, the security challenges and priorities when it comes to OT are different than those of IT, as can be seen in the table provided below. As industries increasingly digitalize, a strong emphasis on cybersecurity has become critical for companies. Cyberattacks have caused operational hazards for companies making use of connected operational technologies, and security concerns have delayed many companies' move to the cloud. Companies must work to enact best practices with regards to networking infrastructure and the deployment of operational technology (OT) and IoT cybersecurity features. Companies are increasingly turning to private networks, segregated and segmented networks, and zero trust models. However, many existing cybersecurity solutions focus on the security challenges facing IT, leaving a gap in the market for OT security. Furthermore, the security challenges and priorities when it comes to OT are different than those of IT, as can be seen in the table provided below.

*Table 1. Aspects of OT Security vs. IT Security*

	OT	IT
<b>Availability</b>	Failure not tolerable	Short failure tolerable
<b>Restart</b>	Difficult	Possible
<b>Patch management</b>	Big challenge	Automated is possible
<b>HW Lifetime</b>	7 to 20 years	3 to 5 years
<b>Know How</b>	Distributed	Centralized

It is critical to align IT and OT at every stage of the digital transformation for enterprises to experience success.

Therefore, it is becoming increasingly important to make use of best practices when designing network infrastructure. Such best practices include, making use of software-defined networking principles where available, and new approaches to cybersecurity, including zero trust models, assume that threats may materialize at multiple parts within the IT and OT infrastructure – including within the network itself.

Factories of the future rely heavily on interconnectivity between multiple devices, sensors, machines, and workstations. A factory can be visualized as a very busy hub of knowledge and data exchange, which requires a robust, secure, and reliable environment in order to operate properly. 5G private networks are emerging as the technology of choice to support Industry 4.0 transformation. Advantages over legacy connectivity technologies include user device density (e.g., 5G can support connectivity of 1 million connected devices per square kilometer), overcoming congestion challenges, reducing costs and power requirements due to the need of fewer 5G radios vs legacy.

A Zero Trust security approach is recommended in highly inhomogeneous environments, where the traditional perimeter security model might fall short.

By definition, Zero Trust is a strategic approach to cybersecurity that secures an organization by eliminating implicit trust and continuously validating every stage of a digital interaction. Zero Trust for 5G removes implicit trust regardless of what the situation is, who the user is, where the user is or what application they are trying to access. The impact of Zero Trust on network

security specifically protects the security of sensitive data and critical applications by leveraging network segmentation, preventing lateral movement, providing Layer 7 threat prevention and simplifying granular user-access controls. Where traditional security models operate under the assumption that everything inside an organization's perimeter can be trusted, the Zero Trust model recognizes that trust is a vulnerability.

To create a zero-trust policy, it is critical to know the asserted **identity of every flow** on the network. Source/destination IP addresses and ports are traditionally used to implement security policy controls by legacy Firewalls. An IP address, however, does not provide identity, as in mobile networks these are assigned by the network and are not constant. Instead, **subscriber identifiers** such as IMSI in 4G and SUPI in 5G are critical. In addition, relevant **equipment IDs**, include the IMEI in 4G and PEI in 5G. If network slicing is enabled, then **Slice ID** identifies the logical network in the 5G environment.

In order to create security policies, you need to base them on things that do not change, such as the above-described permanent identifiers, and not IPs.

Furthermore, the subscriber **location** might also be necessary in defining security policies for Industry 4.0 use cases. 5G networks have enhanced localization functionalities targeting high levels of location accuracy even in indoor environments (introduced in Rel. 15 for NSA operation [2], continued in Rel. 16 with SA operation, with further enhancements in Rel. 17 [3]).

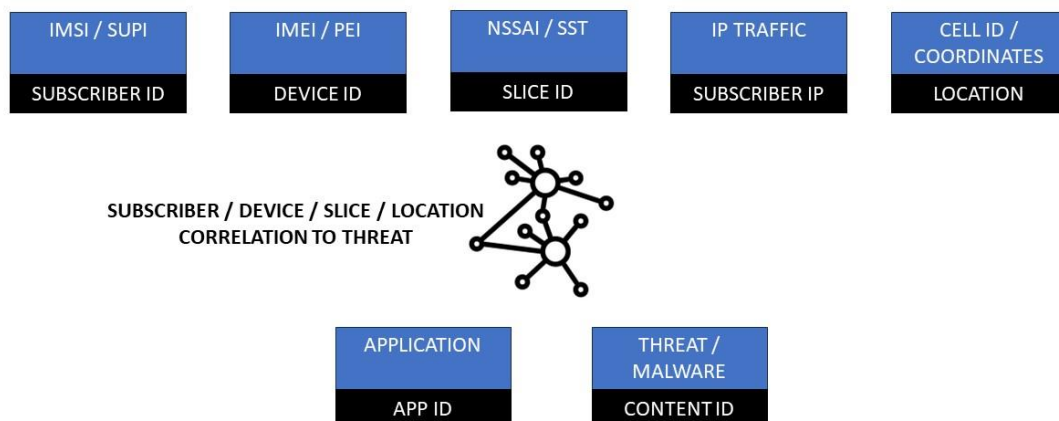


Figure 1. Subscriber / Device / Slice / Location identifiers for Granular Policies and Threat Correlation

Beyond the faster and more robust connectivity that is being introduced by 5G, a more disruptive concept is the network programmability through the 5GC. Programmability may have different interpretations, but in the context explored by this project, it includes the ability to abstract, encapsulate and expose internal capabilities and accept a set of instructions via APIs, and adapt behaviors at runtime accordingly.

For example, in the context of security policies for Industry 4.0 described above, 5GC APIs can be leveraged by external applications to obtain 5G network context information that might be necessary to enforce enhanced security policies. Subscriber and device permanent IDs, user location, subscriber status (e.g., disconnected), network status (e.g., cell congestion) that are

normally “hidden” within 5G, can be exposed and used to enforce enhanced security strategies that otherwise would be impossible to design.

As discussed, security-aware tools that enable accessing, storing, manipulating, and steering data streams and devices form the core basis for guaranteeing reliability in a factory of the future. To address this challenge, the pillar has devoted resources towards:

- Development of a Traffic Management Network App (*Next-Generation Firewall, Virtual & Containerized*) (8Bells).
- A security information and event manager based on blockchain technologies (FOGUS).
- Software for entity authentication, registration, and authorization (InQBit).

## 3 FINAL PROTOTYPE OF NETWORK APPLICATIONS

---

### 3.1 TRAFFIC MANAGEMENT NETWORK APPLICATION

The Traffic Management Network App is essentially a programmable Next-Generation Firewall that is capable of taking into account not only IPs, ports, and MAC addresses to identify user devices, but also any parameters that can be exposed by the 5G network. Such parameters could be permanent subscriber and device identifiers in the mobile network context (IMSI, SUPI, IMEI, PEI). Other parameters that could also be leveraged include subscriber location, network slice ID, etc.

The Traffic Management Network App offers flexible deployment options, such as, in a dedicated server, in a Virtual Machine, or combination (Vertical app in a VM, Network App in a Container), which is the approach that has been selected based on the implementation principles set by the EVOLVED-5G project.

The code of the Network App can be found in the following Github [repository](#).

#### 3.1.1 Use case description

The Traffic Management Network App, which is being developed within the EVOLVED-5G project and falls under the security guarantees and risk analysis pillar, offers the following services through two different use cases:

- Use Case 1: Firewall - IP whitelisting.
- Use Case 2: Throttling: Lessening of the burden on a reportedly congested device in the network, applicable to many different aspects of a FoF.

The above-mentioned use cases and services act as the baseline for future improved implementations in the same direction. As 5G network exposure APIs mature and more network parameters become available for consumption, additional use cases could be developed. These use cases that are envisioned for the future, could enable implementation of advanced security policies that otherwise would be impossible to design (in a traditional firewall).

#### 3.1.2 Detailed Architecture

The overall functionality at a high level is achieved through the interaction of three main blocks, each serving a specific role:

- a. A Network App, which implements the intelligence of the solution, and the interfaces towards the 5G network (CAPIF, NEF emulator, etc.)
- b. A Vertical App (vApp), which is a programmable software switch installed in the user traffic plane
- c. An interface between the Network App and vApp for performing the configuration of the later by the former

The main operation and mechanism (for detecting congestion and limits or redirects) being performed by Eight Bells proposed and developed system, is the following:

- L7 Switch requests for congestion statistics from Network App which is subscribed to MonitoringEvent API

- The processed congestion info (of Network App) supplies the list with devices/destination IPs with high congestion stats
- L7 Switch requests for new congestion statistics from Network App
- The processed congestion info resupplies the list with devices/destination IPs with high congestion stats
- L7 Switch requests for further processing of traffic Filters which are subscribed to MonitoringEvent API
- The processed congestion info resupplies the list with devices/destination IPs with high congestion stats
- No further processing takes place

The vertical application consists of a virtual Switch, implemented using Open vSwitch (OVS) software and the Traffic Control (TC) command-line utility. By combining the capabilities of OVS and TC, control over network traffic, firewalling, optimized network performance and better QoS can be achieved.

OVS (Open vSwitch) is a production quality, multilayer virtual switch licensed under the open-source Apache 2.0 license. It is a software-based virtual switch that operates at the data connection layer in the networking subsystem of the Linux kernel. It enables network virtualization and flexible connectivity in virtualized environments. OVS facilitates communication between virtual machines (VMs) and physical network infrastructure by allowing the creation and control of virtual switches.

OVS provides a wide range of configuration and management interfaces. To improve traffic shaping, quality of service, network monitoring, and analysis, OVS can be integrated with other networking tools such as TC (Traffic Control).

It is widely used in data centers, service providers, and virtualized infrastructure environments, where it provides a flexible and scalable solution for network virtualization and management.

Overall, OVS is an effective software-defined networking technology with several capabilities and flexibility for managing and regulating network traffic in virtualized environments. Because of its open-source nature, which encourages cooperation and innovation, it is a popular choice for developers and network administrators looking for effective network virtualization solutions.



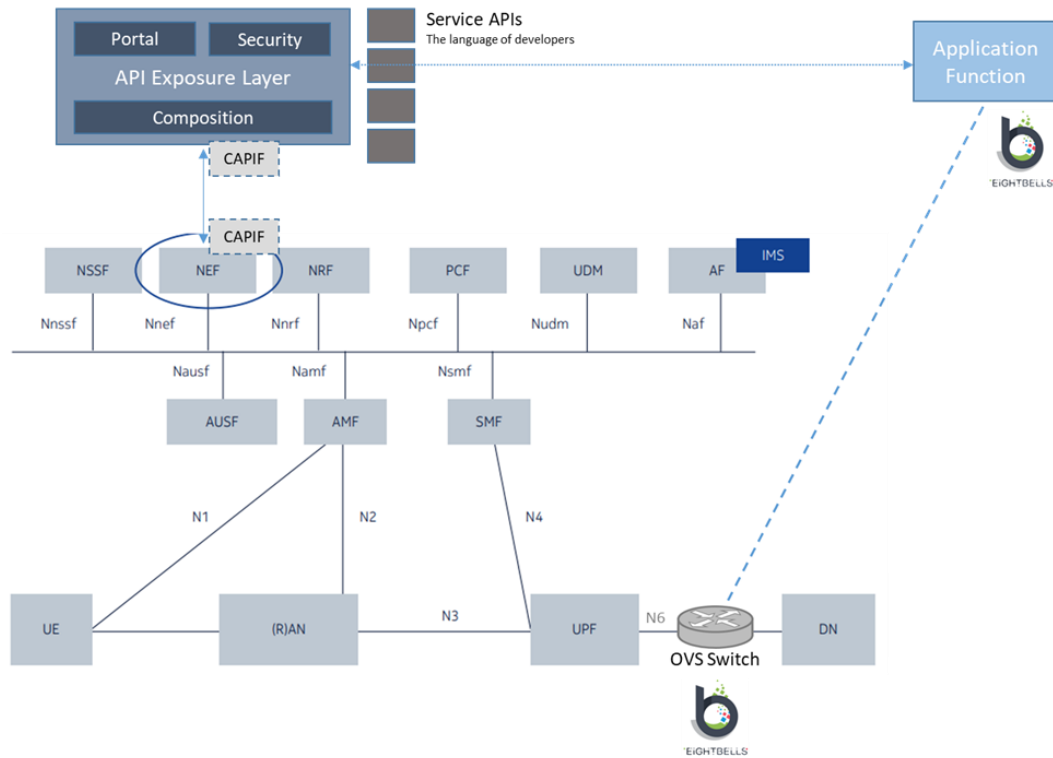


Figure 2. Network App – vApp (L7 Switch) schematic

### 3.1.3 Additional dependencies

The full-stack of the Network Application incorporates a diverse array of technologies and tools to create a solution that leverages the capabilities of the 5G network.

The backend of the application was built with the Python-Flask framework, which is well-known for its simplicity and flexibility in developing web applications. Flask's extensive library support and ease of integration make it an ideal choice for building a robust backend that can handle the complex operations required to process and manage data efficiently.

To ensure efficient and reliable data storage, we selected the popular open-source relational database management system PostgreSQL. PostgreSQL offers advanced features such as data integrity, concurrency control, and robust transaction management, making it an ideal choice for handling the data generated by the application. To manage the database effectively, we also used Adminer, a lightweight and user-friendly database management tool, which will facilitate easy administration and monitoring of the database.

For the frontend, HTML and CSS have been employed to create a visually appealing and intuitive user interface. The focus was on the user's experience that allowed for seamless interaction with the backend services and efficient visualization of the data provided by the 5G network.

To align with all the EVOLVED-5G principles regarding the implementation aspects, the application has been containerized using Docker, in order to achieve consistency and speed in the deployment process across different environments. Docker containers encapsulate the application and its dependencies, providing a portable and reproducible deployment environment. This approach enabled for easily packaging the entire application, including the backend, database, and frontend components, into a single container, ensuring the consistent behavior that was needed for the integration activities and the overall testing of the Network App.

### 3.2 5G SIEM NETWORK APPLICATION

### 3.2.1 Use case description

As the concept of Industry 4.0 evolves, apart from the IP network, industries will progressively establish small or large-scale 5G Non-Public Networks (NPN) to their premises to exploit the advanced capabilities of 5G technology (low-latency, high throughput etc.) for a set of their equipment.

As such, the implementation of enhanced mechanisms to manage and ensure security in this industrial ecosystem is mandatory. For IP networks, SIEM (Security information and event management) systems offer security protection by performing real-time monitoring and analysis of events as well as tracking and logging of security data for compliance or auditing purposes. However, in 5G networks, security management is handled by the 5G Core Network. Therefore, today, in a unified industrial network (IP and 5G NPN), security information systems have no monitoring and control capabilities for industrial devices that use 5G access.

Extending a SIEM system with 5G capabilities enhances the platform by offering access to 5G security information, such as real-time monitoring and updates on the security status of the 5G NPN devices. As a result, the security administrator of an Industry 4.0 environment can have a clearer and more complete picture of the underlying industrial network. FOGUS, with the development of its Network App, aims to bridge the communication gap between SIEM and 5G NPN devices.

### 3.2.2 Detailed Architecture

FOGUS 5G SIEM Network application is a containerized application, following the stand-alone model defined in the EVOLVED-5G project, and resides between the vertical application (AlienVault OSSIM) and 5G Network (CAPIF and NEF Emulator). All the components (OSSIM – FOGUS Network App – NEF Emulator and CAPIF) are deployed locally on our premises, each one on a separate host, but under the same network. The code is uploaded on the project [GitHub repository](#) and a detailed architecture of the Network App is depicted in Figure 3.

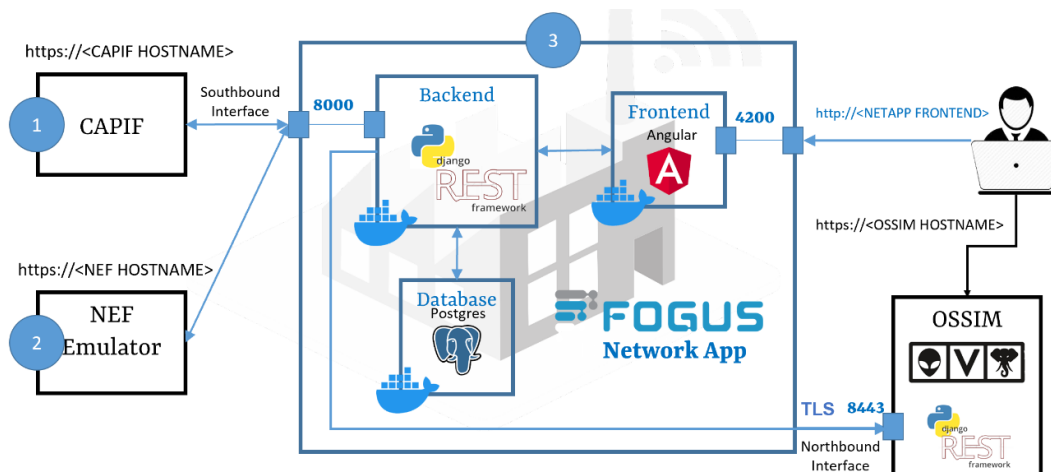
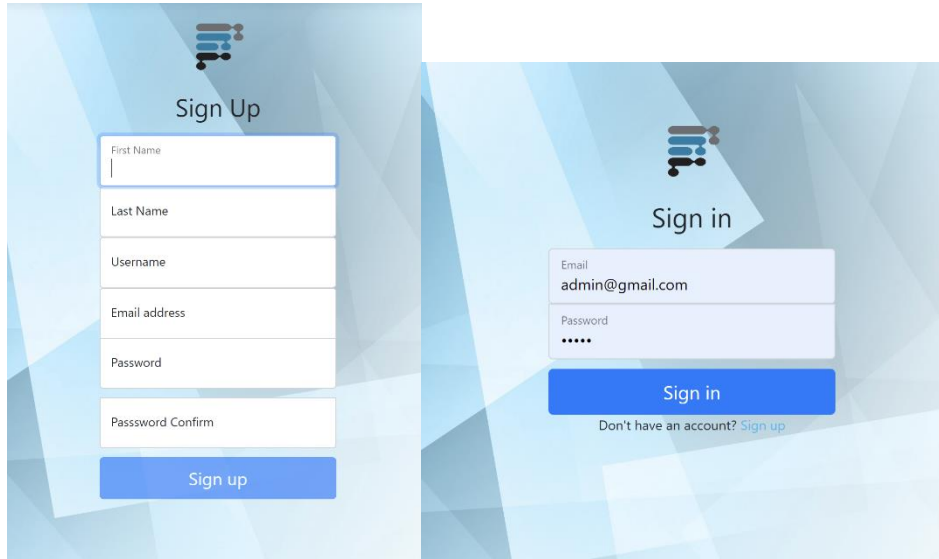


Figure 3. FOGUS Network App architecture

The Network App consists of three services, each one deployed as a separate Docker container:

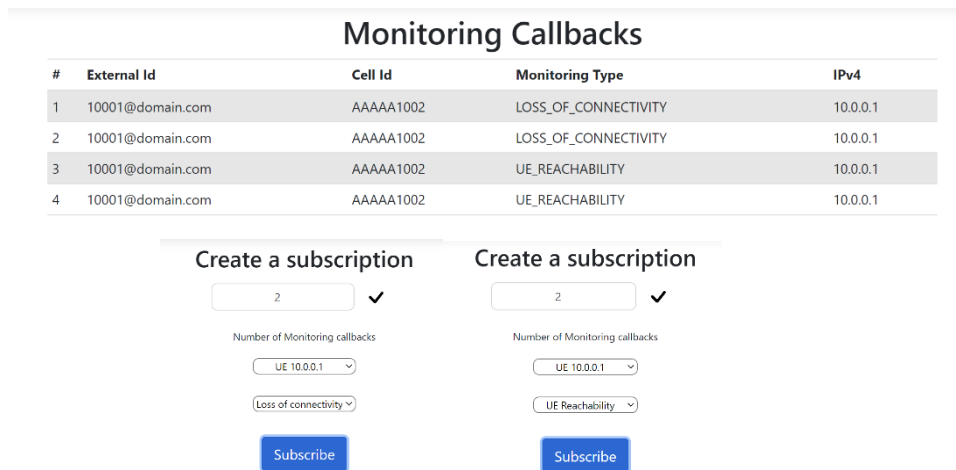
- Frontend (container name: netappfe): A webpage, built on Angular framework, that enables the user to sign up and sign in, as illustrated in Figure 4, and then to create subscriptions

for the NEF APIs, as well as monitor NEF callbacks. The former capability is depicted in Figure 5 and Figure 6.



The image shows two side-by-side web forms. The left form is titled 'Sign Up' and contains input fields for First Name, Last Name, Username, Email address, Password, and Password Confirm, followed by a 'Sign up' button. The right form is titled 'Sign in' and contains input fields for Email (pre-filled with 'admin@gmail.com') and Password (masked with dots), followed by a 'Sign in' button and a link 'Don't have an account? Sign up'.

Figure 4. Sign up and Sign in



The image displays a table titled 'Monitoring Callbacks' with four columns: #, External Id, Cell Id, Monitoring Type, and IPv4. Below the table are two identical 'Create a subscription' forms. Each form includes a text input for the number of callbacks (set to 2), a dropdown for UE IP address (set to UE 10.0.0.1), and a dropdown for monitoring type (set to Loss of connectivity or UE Reachability), followed by a 'Subscribe' button.

#	External Id	Cell Id	Monitoring Type	IPv4
1	10001@domain.com	AAAAA1002	LOSS_OF_CONNECTIVITY	10.0.0.1
2	10001@domain.com	AAAAA1002	LOSS_OF_CONNECTIVITY	10.0.0.1
3	10001@domain.com	AAAAA1002	UE_REACHABILITY	10.0.0.1
4	10001@domain.com	AAAAA1002	UE_REACHABILITY	10.0.0.1

Figure 5. Subscription for Loss of connectivity, UE reachability and monitoring callbacks

## Create a subscription

✓

Number of Monitoring callbacks

External Id	Cell Id	Monitoring Type	IPv4	gNBId
10001@domain.com	AAAAA1002	LOCATION_REPORTING	10.0.0.1	AAAAA1

Figure 6. Subscription for Location reporting

- Backend (container name: netappdjango): A Python application, built on Django framework, implementing communication with the external components (CAPIF, NEF Emulator and OSSIM). It receives data from the 5G network, converts it properly to a readable format and then it sends it over HTTPS protocol to SIEM. The functionality of the framework through the dedicated UI is depicted in Figure 7 below.

## Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups + Add    ✎ Change

NETAPP\_ENDPOINT

Analytics event notifications + Add    ✎ Change

Cells + Add    ✎ Change

Monitoring callbacks + Add    ✎ Change

Users + Add    ✎ Change

Figure 7. UI of backend provided by Django framework

- Database (container name: netapppostgres): A Postgres database, that stores all data exchanged with NEF Emulator and OSSIM. Instances from the database and the logs reflecting the exchange of the data are presented in Figure 8 and Figure 9 respectively.

▼		<b>fogusnetapp</b>		Running (3/3)	22 seconds ago	■	⋮	🗑	
		<b>netapppostgres</b> 81c7a95365ae 	<a href="#">postgres:10</a>	Running	24 seconds ago	■	⋮	🗑	
		<b>netappdjango</b> a693d8c1f9d3 	<a href="#">netappdjango</a>	Running	<a href="#">8000:8000</a> 	23 seconds ago	■	⋮	🗑
		<b>netappfe</b> 9abdd1ec8a10 	<a href="#">netappfe</a>	Running	<a href="#">4200:4200</a> 	22 seconds ago	■	⋮	🗑

Figure 8. The containers of FOGUS Network application

```
[+] Building 3.5s (13/13) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 1.12kB 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/node:16.15.0 2.0s
=> [internal] load build context 0.0s
=> => transferring context: 1.22MB 0.0s
=> [1/8] FROM docker.io/library/node:16.15.0@sha256:59eb4e9d6a344ae1161e7d6d8af831 0.0s
=> CACHED [2/8] RUN apt-get update && apt-get install -yq && apt-get install -yq g 0.0s
=> CACHED [3/8] RUN mkdir /usr/src/app 0.0s
=> CACHED [4/8] WORKDIR /usr/src/app 0.0s
=> CACHED [5/8] COPY package.json /usr/src/app/package.json 0.0s
=> CACHED [6/8] RUN npm install 0.0s
=> CACHED [7/8] RUN npm install -g @angular/cli@13.3.4 0.0s
=> [8/8] COPY . /usr/src/app 1.1s
=> exporting to image 0.4s
=> => exporting layers 0.4s
=> => writing image sha256:a503ec78106e8eb20df13e11bfff7eadb737f740486a53677402da09 0.0s
=> => naming to docker.io/library/netappfe 0.0s
[+] Running 4/4
✓ Network fogusnetapp_default Created 0.1s
✓ Container netapppostgres Started 12.5s
✓ Container netappdjango Started 13.3s
✓ Container netappfe Started 13.9s
```

Figure 9. Logs of the building process of the containers

Once all the containers are up and running, the initial step of the authentication and authorization of Network app by CAPIF is completed. Then using the EVOLVED-5G SDK tools the Network App can communicate with NEF emulator and create subscriptions for the NEF APIs such as Location reporting (UE changes location), Loss of connectivity (UE is out of the cell's range and there is no connection to the 5G network), illustrated in Figure 10, and UE reachability as presented in Figure 11(UE returns to the cell's range and the network connection is restored). The Network App collects the above data and transfers it in a secure way to the OSSIM SIEM environment. Finally, the OSSIM system represents them and provides the necessary alerts and statistics to the user as shown in Figure 12.

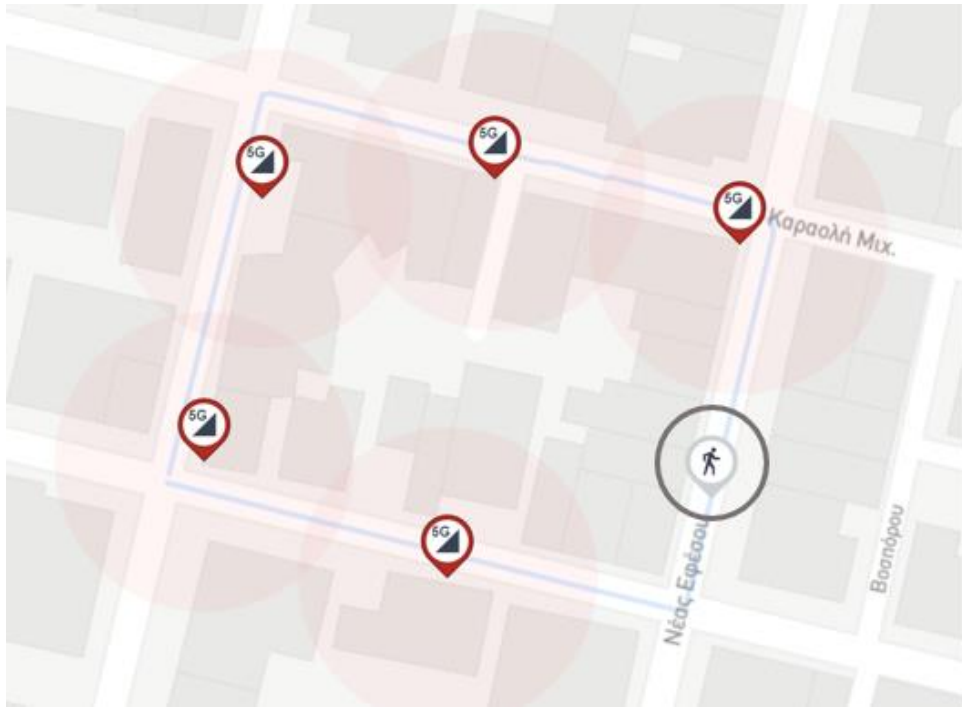


Figure 10. UE out of cell's range s

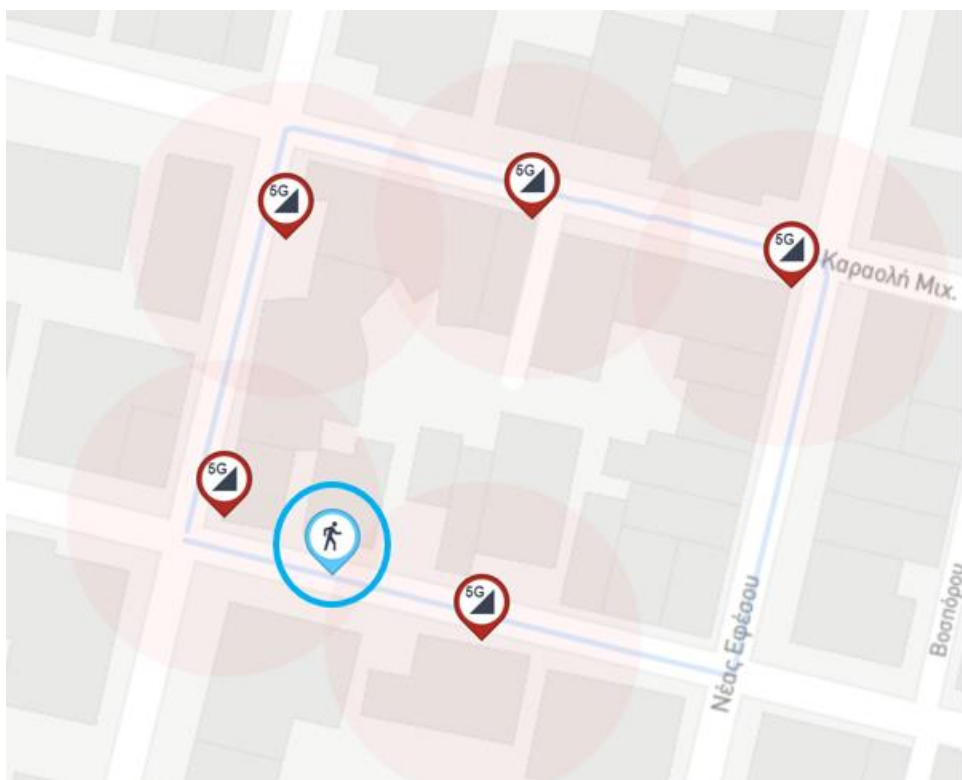


Figure 11. UE in cell's range

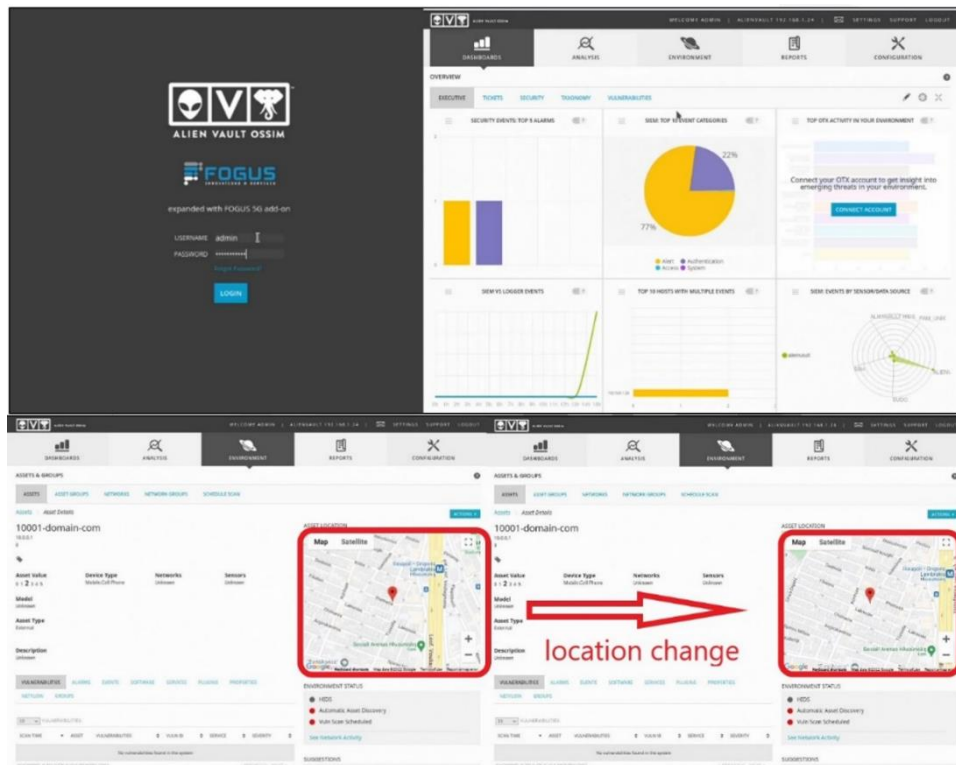


Figure 12. OSSIM alerts and statistics

### 3.2.2 Additional dependencies

FOGUS Network App is a totally containerized application, using Docker containers, and is developed in Angular, Python and Postgres. To deploy locally the application, a [Makefile](#) has been created to simplify the running process. The Network App communicates with NEF Emulator and CAPIF using the EVOLVED-5G SDK tools, and with OSSIM through a set of exposed REST APIs implemented on the Vertical App side, using Python language, and more specifically Django framework.



Figure 13. Tools and technologies of FOGUS Network app



### 3.3 IDENTITY AND ACCESS MANAGEMENT NETWORK APPLICATION

#### 3.3.1 Use case description

With the advancement of mobile networks to the 5th generation and beyond, timely deployment is of essence to capitalize on the benefits of new technologies as fast as possible.

As such, the implementation of security measures to handle identity and access management to ensure the robustness of the ecosystem is critical. The CAPIF framework proposes making use of the OAuth2.0 protocol to handle authorization.

OpenID Connect (OIDC) is an identity layer on top of OAuth2.0 which extends the security capabilities. The identity layer creates a solid foundation for future machine learning implementations that scale well with the incredible number of logs that can be generated for each identity in the context of mobile networks. Furthermore, OIDC enables single sign-on (SSO) between providers to further evolve the authentication process. An additional security measure has been implemented that revokes access by forcefully de-authenticating invokers that either attempt to access inexistent endpoints or attempt to access existing endpoints with disallowed methods.

A Use Case diagram of the Identity and Access Management Network Application is depicted in Figure 14.

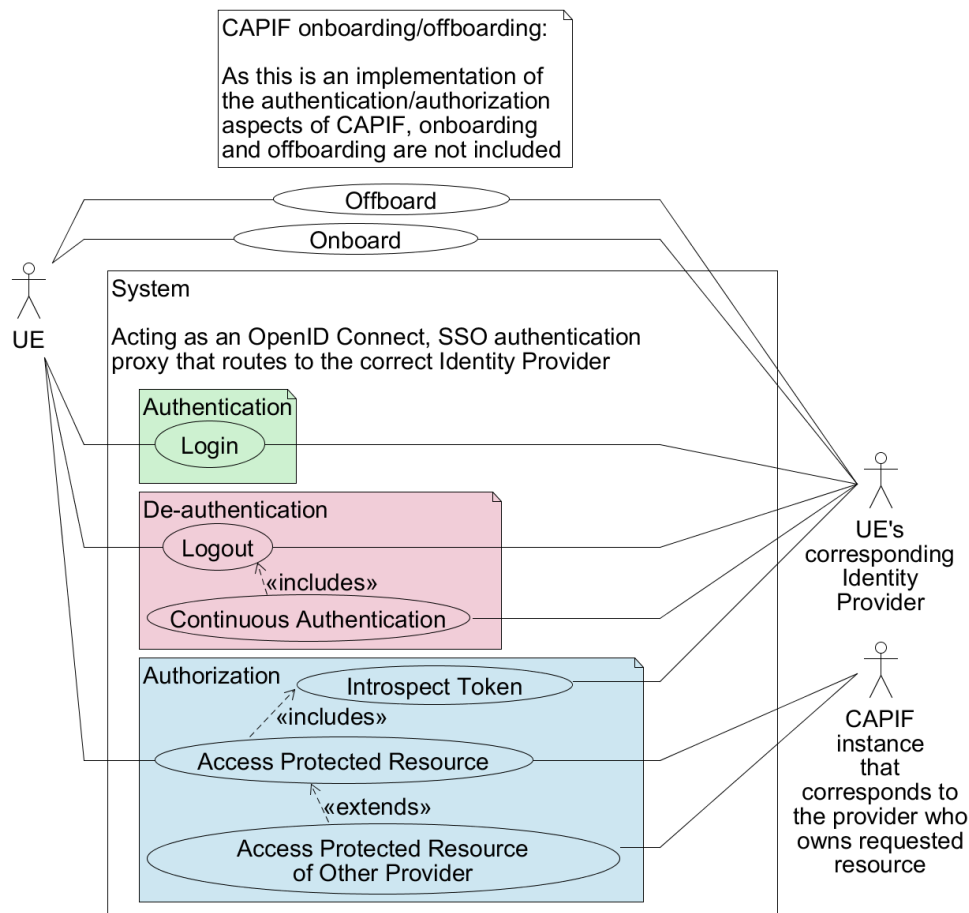


Figure 14. IQB Network Application: Use Case



### 3.3.2 Detailed Architecture

The Identity and Access Management Network Application is a containerized application, following the standalone model defined in the EVOLVED-5G project. The Network Application can handle the authentication and authorization aspects of CAPIF, or act as an intermediary between network applications and CAPIF instances to provide single sign-on capabilities between providers. The code is uploaded on the project's [GitHub repository](#), and the detailed architecture of the Network Application is depicted in Figure 15.

#### NetApp implementation

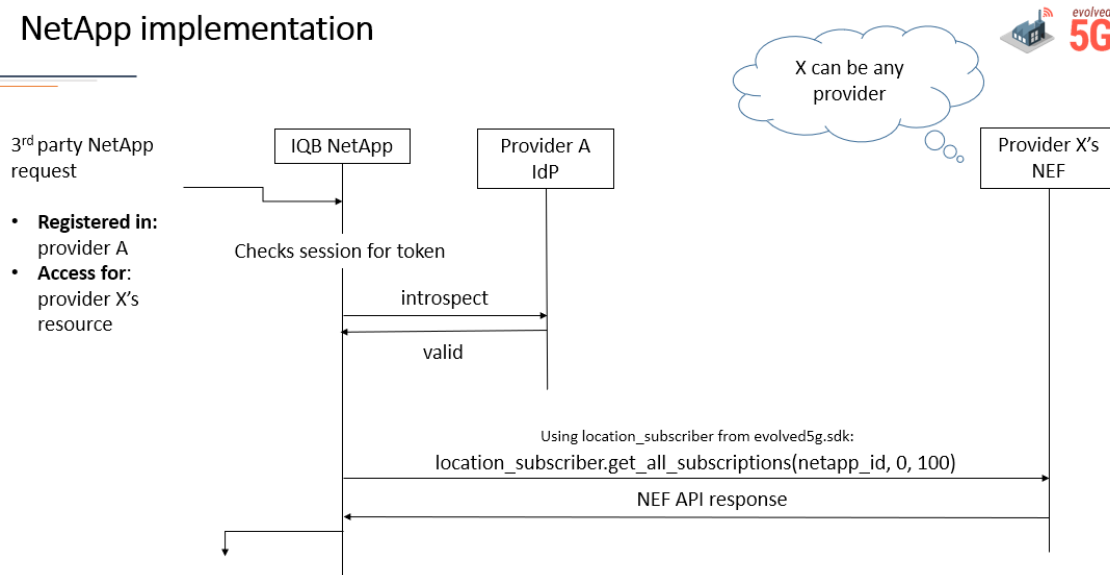


Figure 15. IQB Network Application: Implementation

In order to realize the functionality of the use case, the following containers consist the implementation:

- The Network Application container

The Network application resides in this container. It includes a flask server with endpoints that can be consumed by clients in order to authenticate themselves and access NEF endpoints of their own or another provider through SSO.

- The Keycloak IdP server container

The keycloak server contains a realm and three test clients. Client secrets are transferred to the Network application during deployment to create connectors for future communication between the components. A user has been already onboarded. The keycloak container provides OIDC capabilities such as producing and introspecting tokens.

- The callbacks server container

The callbacks server's purpose is to collect notifications from any subscriptions created on NEF by clients. This container does not directly contribute to the security aspects of the identity and access management Network application, but rather serves as proof of proper communication and functionality between the client, the IQB Network app and the NEF.

The full communication between the components can be summarized as the activity diagram in Figure 16.

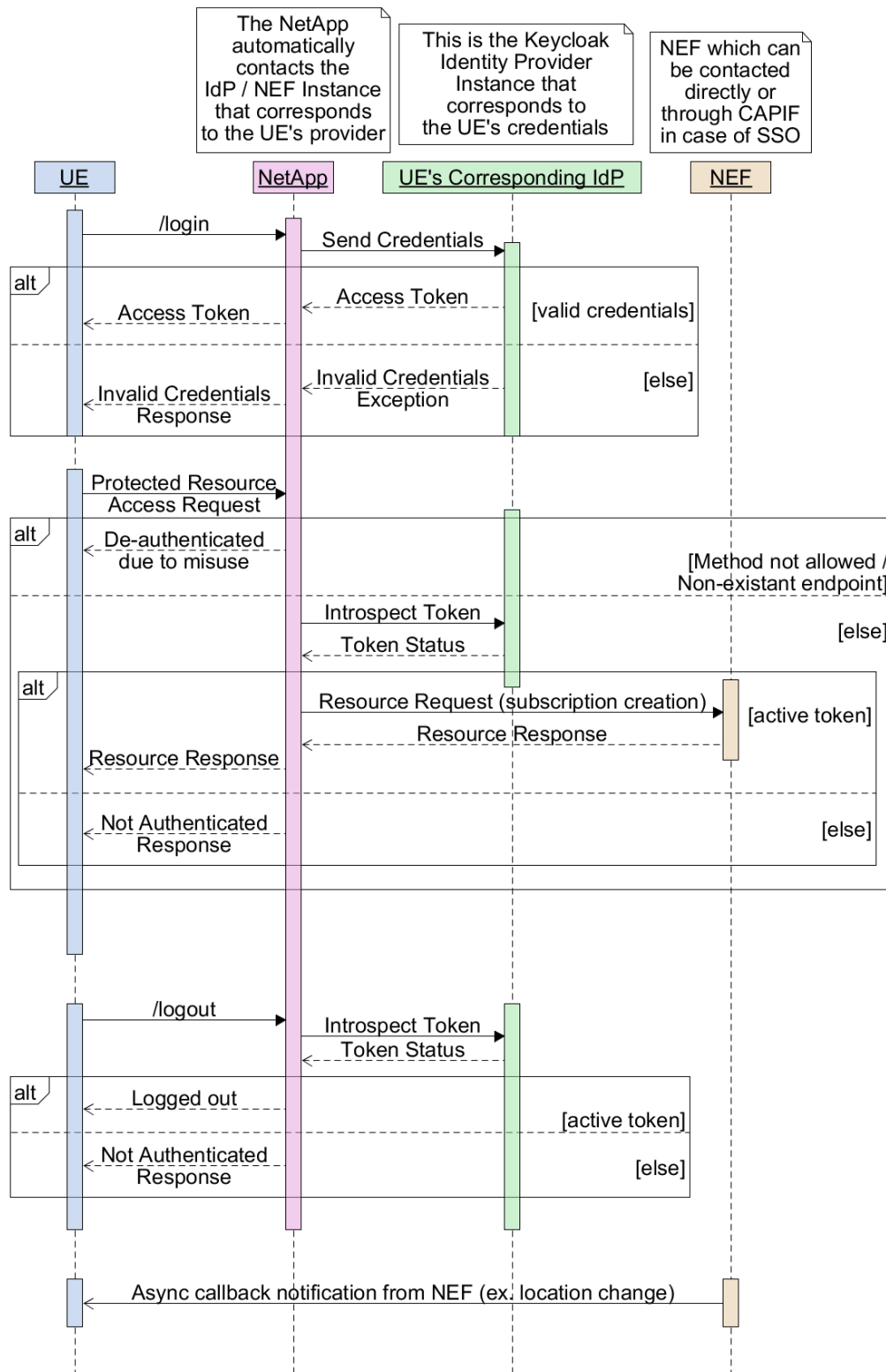
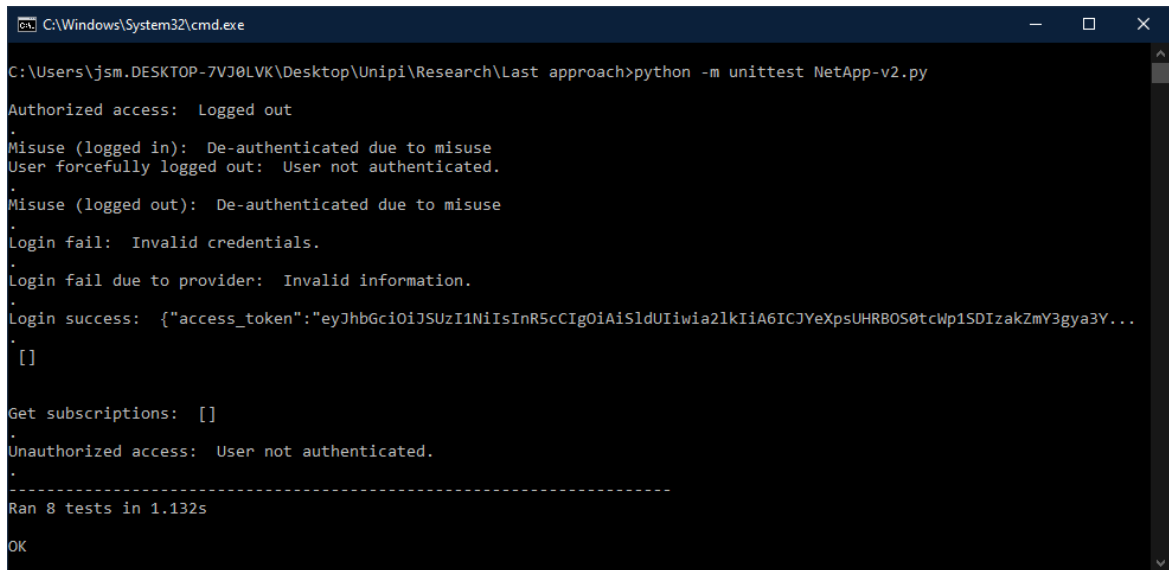


Figure 16. IQB Network Application: Activity Diagram for the components' communication

Once the containers have been deployed, the functionality of the IQB Network app can be tested by running the unit tests as shown in Figure 17.



```
C:\Windows\System32\cmd.exe
C:\Users\jsm.DESKTOP-7VJ0LVK\Desktop\Unipi\Research\Last approach>python -m unittest NetApp-v2.py
Authorized access: Logged out
.
Misuse (logged in): De-authenticated due to misuse
User forcefully logged out: User not authenticated.
.
Misuse (logged out): De-authenticated due to misuse
.
Login fail: Invalid credentials.
.
Login fail due to provider: Invalid information.
.
Login success: {"access_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXLTJ0eXpsUHRBOS0tcWp1SDIzakZmY3gya3Y...
.
[]
.
Get subscriptions: []
.
Unauthorized access: User not authenticated.
.
-----
Ran 8 tests in 1.132s
OK
```

Figure 17. IQB Network Application: Unit Tests Pass

### 3.3.3 Additional dependencies

The Identity and Access Management Network App is written in Python and is a fully containerized implementation that depends on Docker. Keycloak v15.0.2 is required to provide OIDC functionality. The Network app, the Keycloak server and the callbacks server can be deployed all at once using the command "docker compose up". Scripts that are included in the repository will set everything up and establish end to end communication between the components. A pre-defined realm will be automatically imported to Keycloak and secret keys are obtained by the Network app. The latter communicates with the NEF emulator and CAPIF using the EVOLVED-5G SDK-CLI. The repository includes a Postman collection in .json format that can be used to test all endpoints of the network application. Additionally, unit tests have been implemented in python that can validate the functionality of the Network app. Instructions to run the tests are included in the Readme.md file of the repository.

## 4 INTEGRATION ACTIVITIES AND USE CASE TESTING

### 4.1 PURPOSE OF THE INTEGRATION TESTS (1<sup>ST</sup> ROUND)

The first round of integrations has been carried out from April 2022 to April 2023 with the aim of ensuring seamless and reliable communication between various components within the system, including Network Apps, Vertical Apps (vApp), NEF, CAPIF and 5G network connectivity, on top of the cloud infrastructure provided by the Athens platform.

In this initial phase of integration, the following components have been utilized by all the Network Apps:

- Network Applications v3
- NEF releases up to v1.6.2

- CAPIF releases up to v2
- SDK releases up to v0.8.7

The connectivity of 5G with the cloud infrastructure has been verified, specifically the connection between vApps and Demokritos' 5G network.

## 4.2 TOPOLOGY AND SETUP

### 4.2.1 Network App1: Traffic Management Network Application

The first round of integration activity included two deployments.

The first deployment started on February 10<sup>th</sup>, where Telefonica's cloud infrastructure (Openshift) has been used to deploy all the components for compatibility purposes.

The second deployment started on February 15<sup>th</sup>, where the deployment of 8Bells Network Application took place in the cloud infrastructure of NCSRD (Openstack). To test the application, 4 VMs were created to host the NEF Emulator, CAPIF, vertical Application (vApp) and the Network Application respectively. In order to deploy the vertical application, a VPN was used to set up the components remotely at the NCSRD infrastructure. The overall setup in a high-level view can be depicted in the following Figure.

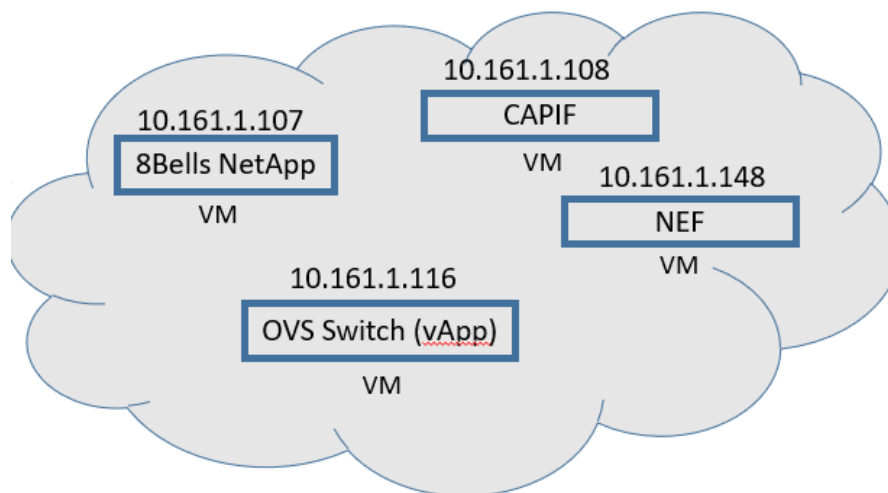


Figure 18. Demokritos Openstack Cloud Infrastructure

In each of the deployments, we have used the following versions:

- SDK tool release v0.8.9
- NEF Emulator v1.6.2
- CAPIF Service v3.0

### 4.2.2 5G SIEM Network Application

The first round of integration activities took place at 23rd of February 2023 in NCSRD premises. In the cloud infrastructure of NCSRD (Openstack), 3 VMs were created to host CAPIF, NEF emulator and FOGUS Network Application respectively. The following resources were given to the VM hosting FOGUS Network Application:

- 2 vCPUs

- 4 GB RAM
- 20 GB Disk

During that period FOGUS application was using version 0.8.9 of SDK tools and was integrated along version 3.0 of CAPIF and 1.6.2 of NEF Emulator.

The vertical application (OSSIM platform) was deployed on FOGUS premises and was accessible via public IP. In figure 19, a topology of the integration activities is presented.

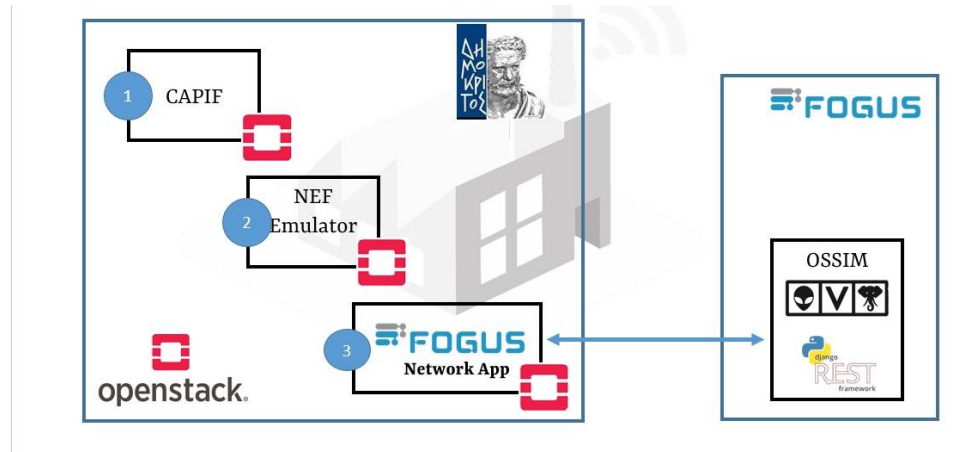


Figure 19. 1<sup>st</sup> Integration Activities Topology

The integration activities in sequence of steps included:

- 1) Deployment of CAPIF Services in Openstack VM as depicted in the Figure below.

```
ubuntu@openSUSE-U:/~$ cat /etc/os-release
NAME="openSUSE Tumbleweed"
VERSION="20240826"
ID="opensuse-tumbleweed"
ID_LIKE="suse"
VERSION_ID="20240826"
PRETTY_NAME="openSUSE Tumbleweed"
ANSI_COLOR="0;32"
LOGO="https://logo.opensuse.org/tumbleweed/"
VENDOR="SUSE"

# All Conif services are running
```

Figure 20. Successful deployment of CAPIF services

- ## 2) Deployment of NEF Services in Openstack VM

```

ubuntu@open5gs-u1: ~/NEF_emulator
ubuntu@open5gs-u1: ~/NEF_emulator
ubuntu@open5gs-u1: ~/NEF_emulator$ docker compose --profile dev up
[+] Running 25/25
  db Pulled
  bb263688fed1 Pull complete
  75a54e59e091 Pull complete
  3ce7f8df2b36 Pull complete
  f36287ef02b9 Pull complete
  dc1f0e8024d8 Pull complete
  7f0a68628bce Pull complete
  32b11818cae3 Pull complete
  48111fe612c1 Pull complete
  f80b1d6d5234 Pull complete
  f19fad3d1049 Pull complete
  bf9102184052 Pull complete
  a3b314ffacae Pull complete
  2ee35d8e1779 Pull complete
  mongo_nef Pulled
  7b1a6ab2e44d Pull complete
  90eb44ebc60b Pull complete
  5005b59f2efb Pull complete
  c7499923d022 Pull complete
  019496b6c44a Pull complete
  b52e5b3baa61 Pull complete
  4737ba38aa64 Pull complete
  94c515c55d41 Pull complete
  8afc96649890 Pull complete
  1cdf67751347 Pull complete
[+] Running 1/4
  network_nef_emulator_services_default Created
  container_nef_emulator-backend-1 Creating
  container_nef_emulator-db-1 Creating
  container_nef_emulator-mongo_nef-1 Creating

```

Figure 21. Successful deployment of NEF services

```

172.21.0.16 - - [23/Feb/2023 11:09:21] "GET /ca-root HTTP/1.1" 201 -
10.161.1.148 - - [23/Feb/2023 11:09:21 +0000] "GET /ca-root HTTP/1.1" 201 1203 "-" "python-requests/2.28.2"
172.21.0.16 - - [23/Feb/2023 11:09:22] "POST /register HTTP/1.1" 201 -
10.161.1.148 - - [23/Feb/2023 11:09:22 +0000] "POST /register HTTP/1.1" 201 222 "-" "python-requests/2.28.2"
[2023-02-23 11:09:22.507] INFO in app: sign-car for user test_nef01
Using configuration from /root/.easyrsa-3.0.4/openssl-easyrsa.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName      :ASN.1 12:'Telefonica I-D'
organizationName :ASN.1 12:'Telefonica I-D'
countryName      :P,IN,INDIA:ES
Certificate is to be certified until Feb 20 11:09:22 2033 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated
172.21.0.15 - - [23/Feb/2023 11:09:22] "POST /sign-car HTTP/1.1" 201 -
172.21.0.16 - - [23/Feb/2023 11:09:22] "POST /getauth HTTP/1.1" 201 -
10.161.1.148 - - [23/Feb/2023 11:09:22 +0000] "POST /getauth HTTP/1.1" 201 6434 "-" "python-requests/2.28.2"
172.21.0.16 - - [23/Feb/2023 11:09:23] "POST /api-provider-management/v1/registries HTTP/1.1" 201 -
10.161.1.148 - - [23/Feb/2023 11:09:23 +0000] "POST /api-provider-management/v1/registrations HTTP/1.1" 201 9508 "-" "python-requests/2.28.2"
[2023-02-23 11:09:23.397] INFO in __main__: Receive Event
[2023-02-23 11:09:23.397] INFO in __main__: Receive Event
172.21.0.16 - - [23/Feb/2023 11:09:23] "POST /published-apis/v1/6d0c378a350e0e/service-apis HTTP/1.1" 201 -
10.161.1.148 - - [23/Feb/2023 11:09:23 +0000] "POST /published-apis/v1/6d0c378a350e0e/service-apis HTTP/1.1" 201 1312 "-" "python-requests/2.28.2"
172.21.0.16 - - [23/Feb/2023 11:09:23] "POST /published-apis/v1/6d0c378a350e0e/service-apis HTTP/1.1" 201 -
10.161.1.148 - - [23/Feb/2023 11:09:23 +0000] "POST /published-apis/v1/6d0c378a350e0e/service-apis HTTP/1.1" 201 1301 "-" "python-requests/2.28.2"
[2023-02-23 11:09:23.449] INFO in __main__: Receive Event
[2023-02-23 11:09:23.449] INFO in __main__: Receive Event

```

Figure 22. NEF is registered and onboarded to CAPIF

3) Inside the VM of FOGUS Network Application a series of actions occurred:

- Download the code of network application
- Map CAPIF IP address to the name "capifcore" by adding a record to /etc/hosts file of FOGUS Network Application VM.
- Edit "env\_to\_copy.dev" to match the correct IP addresses of CAPIF and NEF

4) Deployment of FOGUS Network Application in Openstack VM

```
ubuntu@fogus-netapp:~/FogusNetApps$ make
[+] Running 4/15
  dbnetapp Pulling
    bff3e048017e Pull complete
    e3e180bf7c2b Pull complete
    62eff3cc0cff Pull complete
    3d90a128d4ff Pull complete
    ba4ce0c5ab29 Extracting [=====] 4.129MB/6.106MB
    a8f4b87076a9 Download complete
    4b437d281a7e Download complete
    f1841d9dcb17 Download complete
    b05674a6c170 Download complete
    d59b5be914c6 Download complete
    901d5d9b0beb Download complete
    4a7aa9540b2c Download complete
    0a0d389be22f Download complete
    fb7bd7cfbcd2 Download complete
```

Figure 23. Deployment of FOGUS Network Application

#### 4.2.3 Network App3: Identity Management Network Application

The first round of integration activities took place on the 31st of January 2023 in Telefonica premises. In the cloud infrastructure of Telefonica (openshift), 3 VMs were created to host CAPIF, NEF emulator and IQB Network Application. The versions that were used were SDKv0.8.9, CAPIFv3 and NEFv1.6.2.

On the 9<sup>th</sup> of March 2023, the IQB Network Application was tested on NCSRD premises as well. In the cloud infrastructure of NCSRD (Openstack), 3 VMs were created to host CAPIF, NEF emulator and FOGUS Network Application. The versions tested were SDKv1.0.2, CAPIFv3, and NEFv2. During the integration activities, the following steps were performed inside the VM of IQB Network App:

1) Downloading the code and installing prerequisites:

```
Select iqbit@iqbitserver: ~/IQB-NetApp
iqbit@iqbitserver:~$ git clone https://github.com/EVOLVED-5G/IQB-NetApp/tree/evolved5g
Cloning into 'evolved5g'...
fatal: repository 'https://github.com/EVOLVED-5G/IQB-NetApp/tree/evolved5g/' not found
iqbit@iqbitserver:~$ git clone https://github.com/EVOLVED-5G/IQB-NetApp/
Cloning into 'IQB-NetApp'...
remote: Enumerating objects: 236, done.
remote: Counting objects: 100% (68/68), done.
remote: Compressing objects: 100% (29/29), done.
remote: Total 236 (delta 57), reused 39 (delta 39), pack-reused 168
Receiving objects: 100% (236/236), 80.66 KiB | 842.00 KiB/s, done.
Resolving deltas: 100% (124/124), done.
iqbit@iqbitserver:~$ ls
IQB-NetApp
iqbit@iqbitserver:~$ cd IQB-NetApp
iqbit@iqbitserver:~/IQB-NetApp$ docker-compose up
Command 'docker-compose' not found, but can be installed with:
sudo apt install docker-compose
iqbit@iqbitserver:~/IQB-NetApp$ sudo apt install docker-compose
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz python3-docker python3-dockerpty python3-doccopt
  python3-dotenv python3-texttable python3-websocket runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker-compose docker.io pigz python3-docker python3-dockerpty
  python3-doccopt python3-dotenv python3-texttable python3-websocket runc ubuntu-fan
0 upgraded, 15 newly installed, 0 to remove and 0 not upgraded.
```

Figure 24. Prerequisites Setup (1/2)



```
iqbit@iqbitserver: ~/IQB-NetApp
File "/usr/lib/python3/dist-packages/compose/cli/command.py", line 152, in get_project
  client = get_client()
File "/usr/lib/python3/dist-packages/compose/cli/docker_client.py", line 41, in get_client
  client = docker_client()
File "/usr/lib/python3/dist-packages/compose/cli/docker_client.py", line 170, in docker_client
  client = APIClient(use_ssh_client=not use_paramiko_ssh, **kwargs)
File "/usr/lib/python3/dist-packages/docker/api/client.py", line 197, in __init__
  self.version = self._retrieve_server_version()
File "/usr/lib/python3/dist-packages/docker/api/client.py", line 221, in _retrieve_server_version
  raise DockerException(
docker.errors.DockerException: Error while fetching server API version: ('Connection aborted.', PermissionError(13, 'Permission denied'))
iqbit@iqbitserver:~/IQB-NetApp$ newgrp docker
iqbit@iqbitserver:~/IQB-NetApp$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
```

Figure 25. Prerequisites Setup (2/2)

### 2) Successfully deploying the containers

```
iqbit@iqbitserver:~/IQB-NetApp$ docker-compose up
WARNING: The netapp_name variable is not set. Defaulting to a blank string.
WARNING: The netapp_ip variable is not set. Defaulting to a blank string.
WARNING: The netapp_server_vapp variable is not set. Defaulting to a blank string.
WARNING: The netapp_port_5g variable is not set. Defaulting to a blank string.
WARNING: The netapp_port_web variable is not set. Defaulting to a blank string.
WARNING: The netapp_port_vapp variable is not set. Defaulting to a blank string.
WARNING: The nef_callback_url variable is not set. Defaulting to a blank string.
Building keycloak
Sending build context to Docker daemon 66.05kB
Step 1/8 : FROM quay.io/keycloak/keycloak:15.0.2
15.0.2: Pulling from keycloak/keycloak
2a99c93da168: Pull complete
```

Figure 26. Start Deploying the Containers

```
Select iqbit@iqbitserver: ~/IQB-NetApp
iqb_netapp | IQB NetApp v 2.4
iqb_netapp |
iqb_netapp | -----EMU is not accessible-----
iqb_netapp |
iqb_netapp | * Serving Flask app 'NetApp-v3'
iqb_netapp | * Debug mode: on
iqb_netapp | WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI s
erver instead.
iqb_netapp | * Running on all addresses (0.0.0.0)
iqb_netapp | * Running on http://127.0.0.1:5000
iqb_netapp | * Running on http://172.18.0.4:5000
iqb_netapp | Press CTRL+C to quit
iqb_netapp | * Restarting with watchdog (inotify)
iqb_netapp | /usr/local/lib/python3.9/site-packages/requests/_init_.py:102: RequestsDependencyWarning: urllib3 (1.2
6.14) or chardet (5.1.0)/charset_normalizer (2.0.12) doesn't match a supported version!
iqb_netapp | warnings.warn("urllib3 ({}), or chardet ({}), charset_normalizer ({}), doesn't match a supported "
keycloak | 13:21:19,835 DEBUG [org.keycloak.transaction.JtaTransactionWrapper] (default task-1) new JtaTransactionW
rapper
keycloak | 13:21:19,835 DEBUG [org.keycloak.transaction.JtaTransactionWrapper] (default task-1) was existing? false
keycloak | 13:21:19,840 DEBUG [org.keycloak.authentication.AuthenticationProcessor] (default task-1) AUTHENTICATE C
LIENT
keycloak | 13:21:19,841 DEBUG [org.keycloak.authentication.ClientAuthenticationFlow] (default task-1) client authen
ticator: client-secret
keycloak | 13:21:19,841 DEBUG [org.keycloak.authentication.ClientAuthenticationFlow] (default task-1) client authen
ticator SUCCESS: client-secret
```

Figure 27. All Containers are Up and Running

### 3) Executing the unit tests

The unit tests were executed using the command:

`docker exec iqb_netapp python -m unittest NetApp-v3``

4) The final step was to test manually the endpoints using the Postman collection

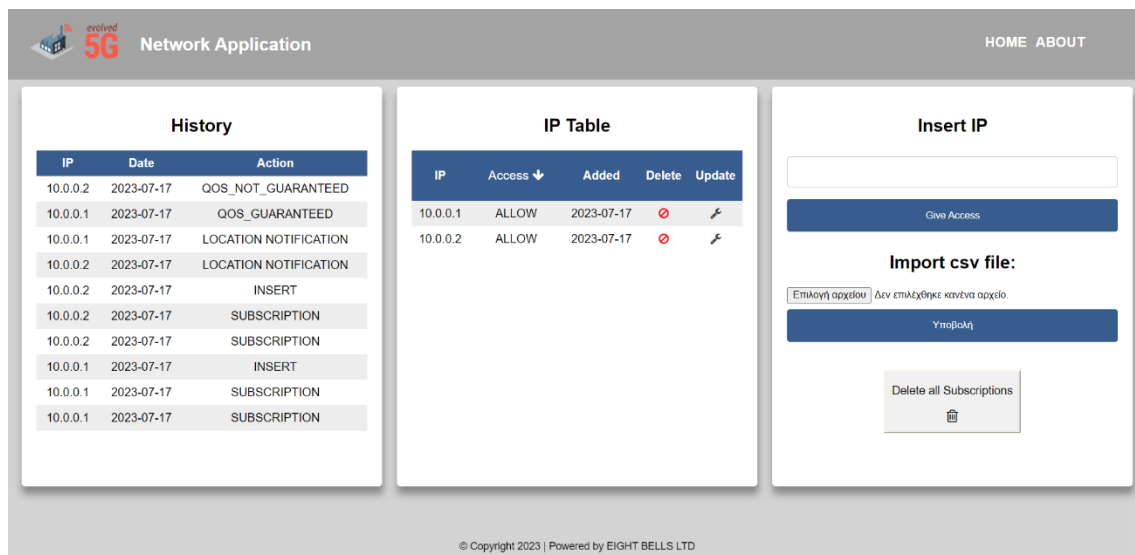


## 4.3 RESULTS AND TAKEWAYS

### 4.3.1 Traffic Management Network Application

The integration activities at Telefonica's Openshift and NSCRD Openstack have both been successful.

Initially, some environmental values needed to be modified to facilitate the deployment of the application. Following this configuration, the testing was performed to ensure successful intercommunication among all components.



The screenshot displays the 'Network Application' interface with a header bar containing 'HOME' and 'ABOUT' links. The main content area is divided into three panels:

- History:** A table with columns 'IP', 'Date', and 'Action'. It lists several actions like 'QOS\_NOT\_GUARANTEED', 'QOS\_GUARANTEED', 'LOCATION NOTIFICATION', 'INSERT', and 'SUBSCRIPTION' for various IP addresses (10.0.0.1 and 10.0.0.2) on 2023-07-17.
- IP Table:** A table with columns 'IP', 'Access', 'Added', 'Delete', and 'Update'. It shows two entries for IP addresses 10.0.0.1 and 10.0.0.2, both with 'ALLOW' access, added on 2023-07-17.
- Insert IP:** A form with an input field for IP, a 'Give Access' button, and an 'Import csv file:' section with a file upload button and a 'Delete all Subscriptions' button.

At the bottom, a footer indicates '© Copyright 2023 | Powered by EIGHT BELLS LTD'.

Figure 28. Successful Execution of the 8Bells Network Application

### 4.3.2 5G SIEM Network Application

The breakdown of the activities and the modifications occurred for the 5G SIEM integration tests are as follows:

After the deployment of the necessary components, the functionality of the network application was tested, by accessing the frontend of FOGUS network application and making Monitoring Location requests.

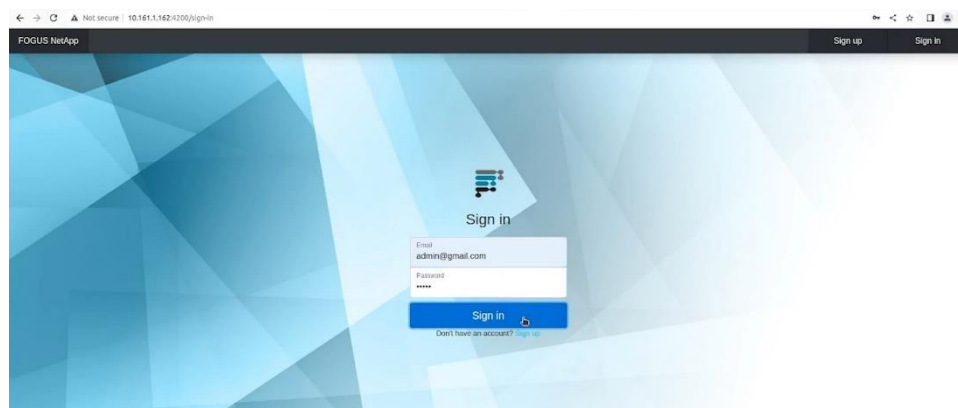
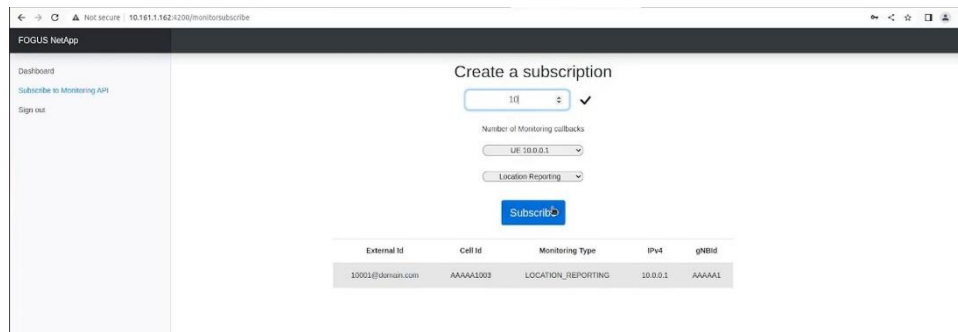
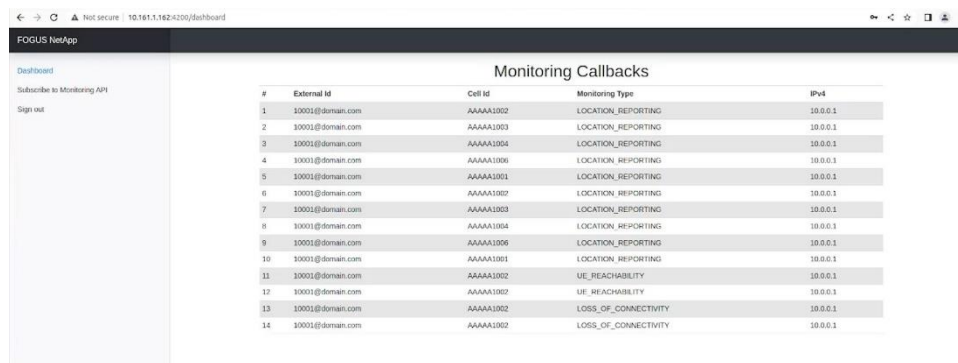


Figure 29. FOGUS frontend portal



External Id	Cell Id	Monitoring Type	IPv4	gNBId
10001@domain.com	AAAAA1003	LOCATION_REPORTING	10.0.0.1	AAAAA1

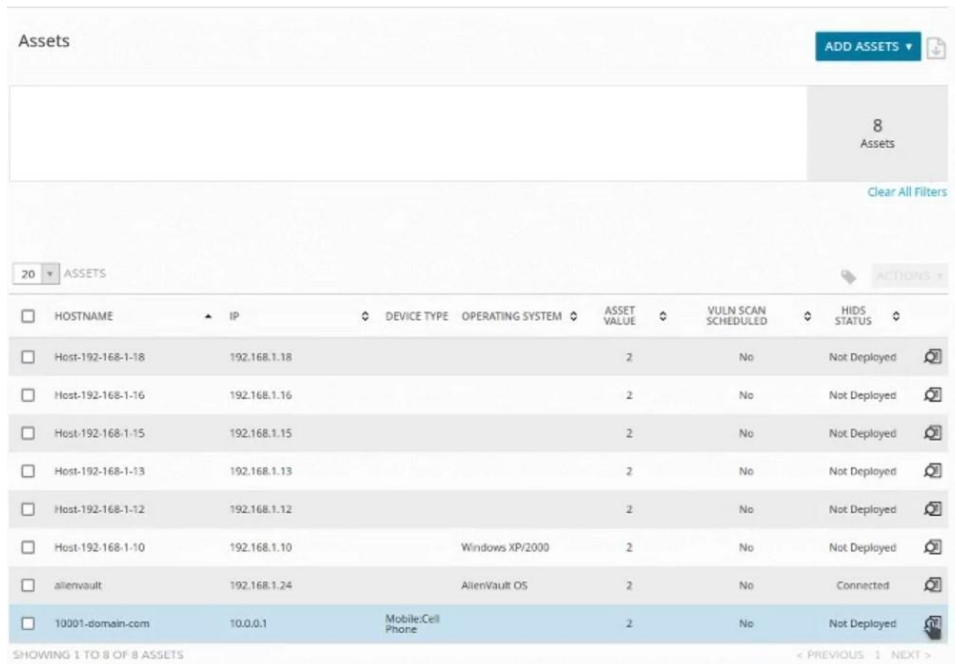
Figure 30. Successful location reporting request



#	External Id	Cell Id	Monitoring Type	IPv4
1	10001@domain.com	AAAAA1002	LOCATION_REPORTING	10.0.0.1
2	10001@domain.com	AAAAA1003	LOCATION_REPORTING	10.0.0.1
3	10001@domain.com	AAAAA1004	LOCATION_REPORTING	10.0.0.1
4	10001@domain.com	AAAAA1006	LOCATION_REPORTING	10.0.0.1
5	10001@domain.com	AAAAA1001	LOCATION_REPORTING	10.0.0.1
6	10001@domain.com	AAAAA1007	LOCATION_REPORTING	10.0.0.1
7	10001@domain.com	AAAAA1008	LOCATION_REPORTING	10.0.0.1
8	10001@domain.com	AAAAA1004	LOCATION_REPORTING	10.0.0.1
9	10001@domain.com	AAAAA1006	LOCATION_REPORTING	10.0.0.1
10	10001@domain.com	AAAAA1001	LOCATION_REPORTING	10.0.0.1
11	10001@domain.com	AAAAA1002	UE_REACHABILITY	10.0.0.1
12	10001@domain.com	AAAAA1002	UE_REACHABILITY	10.0.0.1
13	10001@domain.com	AAAAA1002	LOSS_OF_CONNECTIVITY	10.0.0.1
14	10001@domain.com	AAAAA1002	LOSS_OF_CONNECTIVITY	10.0.0.1

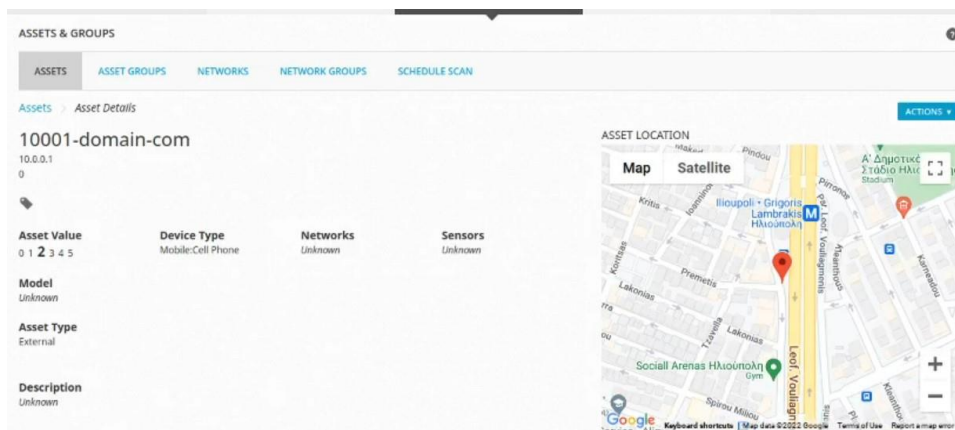
Figure 31. List of successful callbacks from NEF

To validate the correct communication between network and vertical applications access to the OSSIM portal was required in order to check that an asset (device monitored by the vertical application) has been created.



HOSTNAME	IP	DEVICE TYPE	OPERATING SYSTEM	ASSET VALUE	VULN SCAN SCHEDULED	HIDS STATUS
Host-192-168-1-18	192.168.1.18			2	No	Not Deployed
Host-192-168-1-16	192.168.1.16			2	No	Not Deployed
Host-192-168-1-15	192.168.1.15			2	No	Not Deployed
Host-192-168-1-13	192.168.1.13			2	No	Not Deployed
Host-192-168-1-12	192.168.1.12			2	No	Not Deployed
Host-192-168-1-10	192.168.1.10		Windows XP/2000	2	No	Not Deployed
alienvault	192.168.1.24		AlienVault OS	2	No	Connected
10001-domain-com	10.0.0.1	Mobile/Cell Phone		2	No	Not Deployed

Figure 32. List of assets in OSSIM dashboard



**ASSETS & GROUPS**

ASSETS | ASSET GROUPS | NETWORKS | NETWORK GROUPS | SCHEDULE SCAN

Assets > Asset Details

**10001-domain-com**

10.0.0.1

0

**Asset Value**  
0 1 2 3 4 5

**Device Type**  
Mobile/Cell Phone

**Networks**  
Unknown

**Sensors**  
Unknown

**Model**  
Unknown

**Asset Type**  
External

**Description**  
Unknown

**ASSET LOCATION**

Map | Satellite

Map showing the location of the asset in Athens, Greece, near the Sociali Arenas Hlyssimouli and the Leo's Vasilogi.

Figure 33. Description of the asset created after a NEF request

In conclusion, the 1<sup>st</sup> round of integration of FOGUS Network App in NCSRd's cloud infrastructure was successful. After some initial network configuration, the application was correctly deployed, registered/onboarded to CAPIF and used NEF APIs.

## 4.3.3 Identity and Access Management Network Application

On the first round of the integration activity in Telefonica premises, the deployment was thoroughly tested with successful results.

On the second round of the integration activity in NSCRD premises, the identity and access management aspects of the Network App were fully functional as dictated by the unit tests. The unit test related to NEF connectivity failed, providing insights on how the environmental variables need to be properly configured for proper end-to-end communication between components. This was a minor issue and was promptly fixed.

```

iqbit@iqbitserver: ~
-----
Ran 8 tests in 0.901s
FAILED (failures=1)
Authorized access: Logged out
Misuse (logged in): De-authenticated due to misuse
User forcefully logged out: User not authenticated.
Misuse (logged out): De-authenticated due to misuse
Login fail: Invalid credentials.
Login fail due to provider: Invalid information.
Login success: {"access_token":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpzZW50L3N1bWU6Iiwia2lkIiA6IjBjMzY0Yk1SNzd4R0hhNmYwQTB3czF...
Get subscriptions: <!doctype html>
<html lang=en>
<title>500 Internal Server Error</title>
<h1>Internal Server Error</h1>
<p>The server encountered an internal error and was unable to complete your request. Either the server is overloaded or
there is an error in the application.</p>
Unauthorized access: User not authenticated.
iqbit@iqbitserver:~$

```

Figure 34. NCSRD Premises Unit Test First Execution Results

## 4.4 PURPOSE OF THE INTEGRATION TESTS (2<sup>ND</sup> ROUND)

The second round of integrations has been carried out from May 2023 to July 2023.

The purpose of this second integration round was to validate the use-cases utilizing the final components of EVOLVED-5G. On the one hand, NEF, CAPIF and the SDK had been enriched with additional features. On the other hand, SMEs finalized their Network Apps by enhancing the 3.0 version and using the last versions of NEF, CAPIF and SDK. This version 4.1 of the Network Apps also exploited the validation pipeline before the integration test. Finally, the Networks Apps were deployed in Kubernetes clusters in Athens premises instead of using Docker containers running locally.

It's worth noting that until the end of WP3, it was deemed necessary for the SDK to undergo some minor improvements, primarily aimed at enhancing functionality and addressing specific bugs. During this second integration round, the final version of components has been utilized:

- Network Applications v4.1 v4
- NEF v2.2.2
- CAPIF v3.1.2
- SDK v1.0.8
- TSN 1.2.1

## 4.5 TOPOLOGY AND SETUP

### 4.5.1 Traffic Management Network App

The second round of integration activities took place on 29<sup>th</sup> and 30<sup>th</sup> of June 2023 remotely, as physical presence was not necessary. This second round of activities aimed at the deployment of 8BELLS Network App in the NCSRD Kubernetes platform, along CAPIF and NEF, and the evaluation of the end-to-end communication between Network and Vertical App.

The 8BELLS application, has been integrated with the latest versions of all necessary components:

- Version 1.0.8 of EVOLVED5G CLI & SDK package
- Version 3.1.2 of CAPIF and
- Version 2.2.2 of NEF Emulator.

All the necessary components, including the virtual switch/vertical app, have been deployed in the NCSRD K8s cluster. In collaboration with NCSR Demokritos the 2<sup>nd</sup> round of integration activities included the following steps:

Step 1: To Upload the latest docker images of 8Bells Network Application in Dockerhub (<https://hub.docker.com/repository/docker/vasilis8/8bellsnetapp/general>)

Step 2: Created the manifest .yaml files required for the deployment

- o File 1: environment.yaml, including all environmental variables that the Network Application uses

```
1  apiVersion: v1
2  kind: ConfigMap
3  metadata:
4    name: 8bells-configmap
5  data:
6    #NETAPP
7    netapp_name: "myNetapp"
8    netapp_ip: http://172.17.0.1:5000
9    netapp_port: "5000"
10
11   #VAPP
12   vapp_ip: "10.161.1.116"
13   vapp_user: "root"
14   vapp_pass: "8bellsadmin"
15
16   #NEF
17   nef_ip: https://nefemu:4443
18   nef_user: "admin@my-email.com"
19   nef_password: "pass"
20   nef_port: "4443"
21
22   callback_address: "http://bellsnetapp-svc:5000/monitoring/callback"
23   capifhost: "capifcore"
24
25   #DB VARIABLES
26   postgres_db: "postgres"
27   postgres_username: "postgres"
28   postgres_password: "postgres"
29   postgres_port: "5432"
30
31   #ADMINER VARIABLES
32   adminer_password: "1234"
33   adminer_port_one: "8008"
34   adminer_port_two: "8080"
```

Figure 35. environment.yaml file

- File 2: deployment.yaml, including all necessary configurations to create a pod for each container

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: 8bellsnetapp
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: 8bellsnetapp
10   template:
11     metadata:
12       labels:
13         app: 8bellsnetapp
14     spec:
15       containers:
16         - name: 8bellsnetapp
17           image: vasilis8/8bellsnetapp
18           imagePullPolicy: Always
19           envFrom:
20             - configMapRef:
21               name: 8bells-configmap
22           env:
23             - name: NETAPP_NAME
24               value: $(netapp_name)
25             - name: NETAPP_IP
26               value: $(netapp_ip)
27             - name: NEF_IP
28               value: $(nef_ip)
29             - name: NEF_USER
30               value: $(nef_user)
31             - name: NEF_PASS
32               value: $(nef_password)
33             - name: NEF_PORT
34               value: $(nef_port)
35             - name: CALLBACK_ADR
36               value: $(callback_address)
37             - name: DB_NAME
38               value: $(postgres_db)
39             - name: DB_USERNAME
40               value: $(postgres_username)
41             - name: DB_PASS
42               value: $(postgres_password)
43             - name: DB_PORT
44               value: $(postgres_port)
45             - name: CAPIF_HOSTNAME
46               value: $(capifhost)
```

Figure 36. Part of 'deployment.yaml' file

- File 3: service.yaml, including network ports that need to be exposed in each pod.

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: bellsnetapp-svc
5  spec:
6    selector:
7      app: 8bellsnetapp
8    ports:
9      - protocol: TCP
10        port: 5000
11        targetPort: 5000
12  ---
13  apiVersion: v1
14  kind: Service
15  metadata:
16    name: postgres-svc
17  spec:
18    selector:
19      app: postgres
20    ports:
21      - protocol: TCP
22        port: 5432
23        targetPort: 5432
24  ---
25  apiVersion: v1
26  kind: Service
27  metadata:
28    name: adminer-svc
29  spec:
30    selector:
31      app: adminer
32    ports:
33      - protocol: TCP
34        port: 8008
35        targetPort: 8080
```

Figure 37. 'service.yaml' file

Step 3: Perform Minor fixes to the Ingress Controller to make a successful deployment

Step 4: Testing the deployment with vApp and all the components and the integration among them.

#### 4.5.2 5G SIEM Network Application

The second round of integration activities took place at 28<sup>th</sup> and 29<sup>th</sup> of June 2023 remotely, as FOGUS use case did not require any physical presence. This second round of activities aimed at the deployment of FOGUS Network App in the NCSRD Kubernetes platform, along CAPIF and NEF, and the evaluation of the end-to-end communication between Network and Vertical App.

FOGUS application, during the time of the integration activities, was using the latest versions of all necessary components:

- Version 1.0.8 of Evolved5G CLI & SDK package
- Version 3.1.2 of CAPIF and
- Version 2.2.2 of NEF Emulator.

The vertical application (OSSIM platform) was deployed on FOGUS premises and was accessible via public IP. In figure 38, a topology that has been utilised during the integration activities is presented.

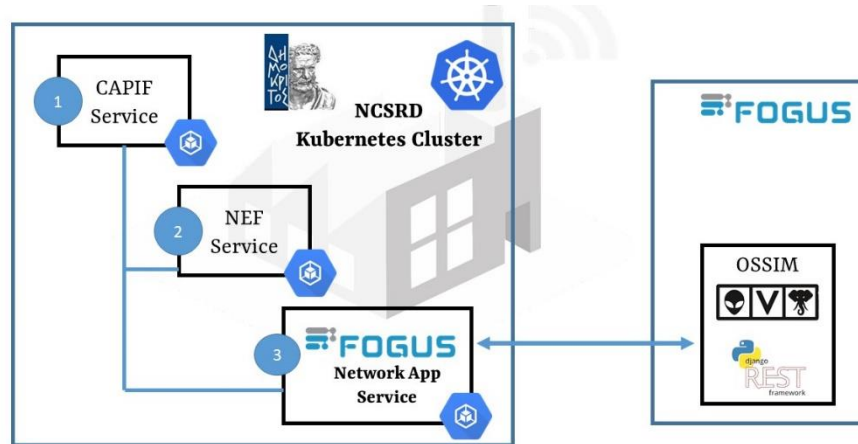


Figure 38. 2nd Integration Activities Topology

The 2<sup>nd</sup> round of integration activities included the following steps:

- 1) Upload the images of FOGUS network application in Dockerhub
- 2) Creation of 3 yaml files required for FOGUS Network Application to run on Kubernetes platform.
  - a. "environment.yaml", including all environmental variables that are imported to the Network Application during the deployment phase

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: fogus-configmap
data:
  NEF_ADDRESS: nefemu:4443
  NEF_USER: admin@my-email.com
  NEF_PASSWORD: pass
  PATH_TO_CERTS: /code/capif_onboarding
  CAPIF_HOSTNAME: capifcore
  CAPIF_PORT_HTTP: "8080"
  CAPIF_PORT_HTTPS: "443"
  CALLBACK_ADDRESS: http://fogusnetworkapp-svc-be:8000
  BACKEND_ADDRESS: https://fogusnetapp-backend.com
  FRONTEND_ADDRESS: https://fogusnetapp-frontend.com
  POSTGRES_SERVER: fogusnetworkapp-svc-db
  POSTGRES_URI: fogusnetworkapp-svc-db
  POSTGRES_PORT: "5432"
  POSTGRES_DB: evolvedb
  POSTGRES_USER: evolveclient
  POSTGRES_PASSWORD: evolvepass
  VAPP_ADDRESS: 195.134.66.79:8443
```

Figure 39. FOGUS environment.yaml

- b. "deployment.yaml", which contains all necessary configurations to create a pod for each container of the network application



```

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fogusnetworkapp-dep-be
spec:
  replicas: 1
  selector:
    matchLabels:
      app: fogusnetworkapp-pod-be
  template:
    metadata:
      labels:
        app: fogusnetworkapp-pod-be
    spec:
      containers:
        - name: netappdjango
          image: vtsolkas/fogus-network-app:netappdjango-1.1.1
          imagePullPolicy: Always
          command: ["/wait_db.sh", "${POSTGRES_SERVER}:${POSTGRES_PORT}", "--", "sh", "docker_start_up.sh"]
          envFrom:
            - configMapRef:
                name: fogus-configmap
          ports:
            - containerPort: 8000
---

```

Figure 40. FOGUS deployment.yaml

- c. “service.yaml”, which defines all network ports that need to be exposed in each pod

```

---
apiVersion: v1
kind: Service
metadata:
  name: fogusnetworkapp-svc-be
spec:
  selector:
    app: fogusnetworkapp-pod-be
  ports:
    - protocol: TCP
      port: 8000
      targetPort: 8000
---
apiVersion: v1
kind: Service
metadata:
  name: fogusnetworkapp-svc-fe
spec:
  selector:
    app: fogusnetworkapp-pod-fe
  ports:
    - protocol: TCP
      port: 4200
      targetPort: 4200

```

Figure 41. FOGUS services.yaml

- 3) Deployment of FOGUS Network Application using the above-mentioned yaml files and making some adjustment to the Ingress Controller of Kubernetes Infrastructure. This step was performed remotely (via internet call).

### 4.5.3 Identity and Access Management Network Application

The second round of integration activities were performed on June 27<sup>th</sup> 2023, remotely at NCSRD premises. The network application was deployed alongside CAPIF and NEF on Kubernetes, and the end-to-end communication was tested.

This version of the network application has been integrated with SDKv1.0.7, CAPIFv3.1.2, NEFv2.2.2.

The components were deployed on Kubernetes through the following process:

Step 1: Each component was uploaded on Docker Hub:

- [https://hub.docker.com/repository/docker/johnst99/iqb\\_netapp\\_img](https://hub.docker.com/repository/docker/johnst99/iqb_netapp_img)
- [https://hub.docker.com/repository/docker/johnst99/keycloak\\_img](https://hub.docker.com/repository/docker/johnst99/keycloak_img)
- <https://hub.docker.com/repository/docker/johnst99/callbacks>

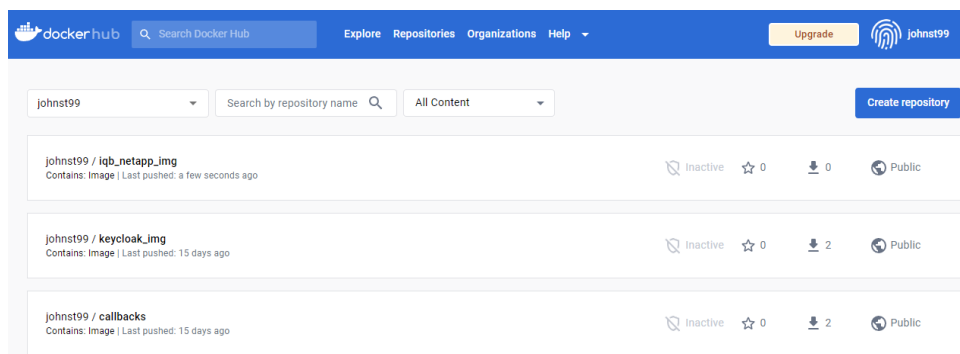


Figure 42. IQB Dockerhub images

Step 2: Manifest .yaml files were created in order to perform deployment

environment.yaml: includes the environmental variables required by the containers

```

1  apiVersion: v1
2  kind: ConfigMap
3  metadata:
4    name: iqbit-configmap
5  data:
6    NETAPP_ID: "myNetapp"
7    KEYCLOAK_ADDRESS: "keycloak:8980/auth"
8    KEYCLOAK_REALM: "EVOLVED-5G"
9    KEYCLOAK_ADMIN: "admin"
10   KEYCLOAK_ADMIN_PASSWORD: "admin"
11   NEF_ADDRESS: "nefemu:4443"
12   NEF_USER: "admin@my-email.com"
13   NEF_PASSWORD: "pass"
14   VAPP_ADDRESS: "NA"
15   PATH_TO_CERTS: "/app/capif_onboarding"
16   CAPIF_HOSTNAME: "capifcore"
17   CAPIF_PORT_HTTP: "8080"
18   CAPIF_PORT_HTTPS: "443"
19   CALLBACK_ADDRESS: "callbacks:5002"
20   FRONTEND_ADDRESS: "NA"

```

Figure 43. IQB environment.yaml

- deployment.yaml: includes pod configuration such as ports, environmental variable mapping and labeling, it is the docker-compose.yml equivalent of Kybernetes

```
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: iqbnetapp
5  spec:
6    replicas: 1
7    selector:
8      matchLabels:
9        app: iqbnetapp
10   template:
11     metadata:
12       labels:
13         app: iqbnetapp
14     spec:
15       containers:
16         - name: iqbnetapp
17           image: johnst99/iqb_netapp_img
18           imagePullPolicy: Always
19           env:
20             - name: NETAPP_ID
21               valueFrom:
22                 configMapKeyRef:
23                   name: iqbit-configmap
24                   key: NETAPP_ID
25             - name: KEYCLOAK_ADDRESS
26               valueFrom:
27                 configMapKeyRef:
28                   name: iqbit-configmap
29                   key: KEYCLOAK_ADDRESS
30             - name: KEYCLOAK_REALM
31               valueFrom:
32                 configMapKeyRef:
33                   name: iqbit-configmap
34                   key: KEYCLOAK_REALM
35             - name: KEYCLOAK_ADMIN
36               valueFrom:
37                 configMapKeyRef:
38                   name: iqbit-configmap
39                   key: KEYCLOAK_ADMIN
40             - name: KEYCLOAK_ADMIN_PASSWORD
41               valueFrom:
42                 configMapKeyRef:
```

Figure 44. IQB deployment.yaml

- service.yaml: includes the port mappings and protocol that need to be exposed on the services

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: iqbnetapp
5  spec:
6    selector:
7      app: iqbnetapp
8    ports:
9      - protocol: TCP
10      port: 5000
11      targetPort: 5000
12  ---
13  apiVersion: v1
14  kind: Service
15  metadata:
16    name: keycloak
17  spec:
18    selector:
19      app: keycloak
20    ports:
21      - protocol: TCP
22      port: 8980
23      targetPort: 8080
24  ---
25  apiVersion: v1
26  kind: Service
27  metadata:
28    name: callbacks
29  spec:
30    selector:
31      app: callbacks
32    ports:
33      - protocol: TCP
34      port: 5002
35      targetPort: 5002
```

Figure 45. IQB service.yaml

Step 3: Setting the proper Ingress rules

```
rlz@rlz:~/github/evolved/k8s-validation-templates/iqbitnetapp$ kubectl describe ingresses
Name: nef-ingress
Labels: app=nginx-ingress
Namespace: default
Address: 10.220.2.201
Ingress Class: nginx
Default backend: <default>
TLS:
  test-tls terminates validation-athens.com
Rules:
  Host            Path  Backends
  ----            -
  validation-athens.com /    nef-backend:80 (10.244.1.126:80)
  mongocapif.com    /    capif-mongo-express:8082 (10.244.1.65:8081)
  mongonef.com       /    mongo-express:8081 (10.244.1.248:8081)
  iqbitnetapp.com    /    iqbitnetapp:5000 (10.244.1.75:5000)
Annotations: <none>
Events:
  Type    Reason    Age          From          Message
  ----    -
  Normal  Sync      2m23s (x8 over 10d)  nginx-ingress-controller  Scheduled for sync
rlz@rlz:~/github/evolved/k8s-validation-templates/iqbitnetapp$
```

Figure 46. IQB Ingress Rules

Step 4: Deployment of each service using the manifest files

```
rlz@rlz:~/github/evolved/k8s-validation-templates/iqbitnetapp$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP              NODE           NOMINATED NODE   READINESS GATES
api-Invocation-logs-6b96f499c-pxnlv 1/1     Running   0           52m   10.244.1.250    nef-worker     <none>            <none>
api-Invoker-management-5888597bb-k2np5 1/1     Running   2 (52m ago)  52m   10.244.1.249    nef-worker     <none>            <none>
api-provider-management-55f99cff9f-x7x9d 1/1     Running   2 (52m ago)  52m   10.244.1.5      nef-worker     <none>            <none>
backend-55498549c6-nvnnns           1/1     Running   0           7m15s 10.244.1.126    nef-worker     <none>            <none>
callbacks-bccc66cfd-zpwpv           1/1     Running   0           52s   10.244.1.58     nef-worker     <none>            <none>
capif-events-8476b4875d-twfsq        1/1     Running   0           52m   10.244.1.48     nef-worker     <none>            <none>
capif-mongo-7d89e9f6f8-k2s2b         1/1     Running   0           52m   10.244.1.177    nef-worker     <none>            <none>
capif-mongo-express-7d547f8679-j6tzt 1/1     Running   2 (51m ago)  52m   10.244.1.65     nef-worker     <none>            <none>
capif-routing-info-5d8fd667b-v9rwn   1/1     Running   0           52m   10.244.1.209    nef-worker     <none>            <none>
capif-security-758c9bcd8b-4l4rf      1/1     Running   2 (52m ago)  52m   10.244.1.18     nef-worker     <none>            <none>
db-65465448bd-w67kd                 1/1     Running   0           7m16s 10.244.1.136    nef-worker     <none>            <none>
easy-rsa-88f8cd5b5-5mgmr             1/1     Running   0           52m   10.244.1.194    nef-worker     <none>            <none>
iqbitnetapp-fd8c86bdb-hp9tb          1/1     Running   0           52s   10.244.1.75     nef-worker     <none>            <none>
jwtauth-5cdfb84b5c-2mkck            1/1     Running   1 (52m ago)  52m   10.244.1.252    nef-worker     <none>            <none>
keycloak-6966fd4c4b-2kzpb           1/1     Running   0           52s   10.244.1.1      nef-worker     <none>            <none>
logs-5d499587f4-6xsnc               1/1     Running   0           52m   10.244.1.166    nef-worker     <none>            <none>
mongo-express-6c9cc9f746-ygxj7       1/1     Running   1 (7m13s ago) 7m16s 10.244.1.248    nef-worker     <none>            <none>
nef-mongo-9b85fcd44-t92bk            1/1     Running   0           7m16s 10.244.1.49     nef-worker     <none>            <none>
nginx-578d7d64f8-d42bc              1/1     Running   0           52m   10.244.1.71     nef-worker     <none>            <none>
published-apis-5ff8f5f8dd-f7jqg      1/1     Running   0           52m   10.244.1.176    nef-worker     <none>            <none>
redis-7c8976bb95-xtpg8              1/1     Running   0           52m   10.244.1.247    nef-worker     <none>            <none>
reverse-proxy-84b8865bf6-7ls5j       1/1     Running   0           7m15s 10.244.1.113    nef-worker     <none>            <none>
service-apis-d5645c484-d2946         1/1     Running   0           52m   10.244.1.218    nef-worker     <none>            <none>
```

Figure 47. IQB Pods Deployed

Step 5: Performing unit tests and additional functionality tests.

## 4.6 RESULTS AND TAKEAWAYS

### 4.6.1 Traffic Management Network Application

After the successful deployment and testing of the Network Application in the NCSR Kubernetes platform, we proceeded to test the end-to-end functionality. Following are some screenshots running the Network application after deployment.

```
10.244.1.48 - - [30/Jun/2023:12:17:41 +0000] "GET /ca-root HTTP/1.1" 201 1203 "-" "python-requests/2.31.0"
10.244.1.48 - - [30/Jun/2023:12:17:41 +0000] "POST /register HTTP/1.1" 201 216 "-" "python-requests/2.31.0"
10.244.1.48 - - [30/Jun/2023:12:17:41 +0000] "POST /getauth HTTP/1.1" 200 657 "-" "python-requests/2.31.0"
10.244.1.48 - - [30/Jun/2023:12:17:44 +0000] "POST /api-provider-management/v1/registrations HTTP/1.1" 201 18778 "-" "python-requests/2.31.0"
10.244.1.48 - - [30/Jun/2023:12:17:44 +0000] "POST /published-apls/v1/cf1b15e30be358ea46982db872bd05/service-apls HTTP/1.1" 201 1310 "-" "python-requests/2.31.0"
10.244.1.48 - - [30/Jun/2023:12:17:44 +0000] "POST /published-apls/v1/cf1b15e30be358ea46982db872bd05/service-apls HTTP/1.1" 201 1299 "-" "python-requests/2.31.0"

10.244.1.172 - - [30/Jun/2023:12:18:01 +0000] "POST /register HTTP/1.1" 201 223 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:02 +0000] "POST /getauth HTTP/1.1" 200 1868 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:03 +0000] "POST /api-invoker-management/v1/onboardedInvokers HTTP/1.1" 201 6359 "-" "python-requests/2.26.0"

10.244.1.172 - - [30/Jun/2023:12:18:43 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:43 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:43 +0000] "PUT /capif-security/v1/trustedInvokers/856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 201 371 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:44 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/token HTTP/1.1" 200 915 "-" "python-requests/2.26.0"
10.244.1.48 - - [30/Jun/2023:12:18:44 +0000] "POST /api-invocation-logs/v1/98d47b4096ba790f669fd1e6859c6/logs HTTP/1.1" 201 1621 "-" "python-requests/2.31.0"
10.244.1.172 - - [30/Jun/2023:12:18:44 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:44 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:44 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:44 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/update HTTP/1.1" 200 523 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:45 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/token HTTP/1.1" 200 908 "-" "python-requests/2.26.0"
10.244.1.48 - - [30/Jun/2023:12:18:45 +0000] "POST /api-invocation-logs/v1/98d47b4096ba790f669fd1e6859c6/logs HTTP/1.1" 201 1168 "-" "python-requests/2.31.0"
10.244.1.172 - - [30/Jun/2023:12:18:53 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:53 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:53 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/token HTTP/1.1" 200 915 "-" "python-requests/2.26.0"
10.244.1.48 - - [30/Jun/2023:12:18:54 +0000] "POST /api-invocation-logs/v1/98d47b4096ba790f669fd1e6859c6/logs HTTP/1.1" 201 1621 "-" "python-requests/2.31.0"
10.244.1.172 - - [30/Jun/2023:12:18:54 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:54 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:54 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:55 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/token HTTP/1.1" 200 908 "-" "python-requests/2.26.0"
10.244.1.48 - - [30/Jun/2023:12:18:55 +0000] "POST /api-invocation-logs/v1/98d47b4096ba790f669fd1e6859c6/logs HTTP/1.1" 201 1168 "-" "python-requests/2.31.0"
10.244.1.172 - - [30/Jun/2023:12:18:59 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:59 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:59 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:18:59 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/token HTTP/1.1" 200 915 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:19:00 +0000] "POST /api-invocation-logs/v1/98d47b4096ba790f669fd1e6859c6/logs HTTP/1.1" 201 1621 "-" "python-requests/2.31.0"
10.244.1.172 - - [30/Jun/2023:12:19:00 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:19:00 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:19:00 +0000] "GET /service-apls/v1/allServiceAPIs?api-invoker-id=856ebefdc3d56cfac2647260f37c9 HTTP/1.1" 200 2641 "-" "python-requests/2.26.0"
10.244.1.172 - - [30/Jun/2023:12:19:01 +0000] "POST /capif-security/v1/securlities/856ebefdc3d56cfac2647260f37c9/token HTTP/1.1" 200 908 "-" "python-requests/2.26.0"
10.244.1.48 - - [30/Jun/2023:12:19:01 +0000] "POST /api-invocation-logs/v1/98d47b4096ba790f669fd1e6859c6/logs HTTP/1.1" 201 1168 "-" "python-requests/2.31.0"
```

Figure 48. Capif logs during the runtime of the Network Application

```
fl@fl:~/github/evolved/k8s-validation-templates/8bellsnetapp$ kubectl logs -f 8bellsnetapp-7f87c5f9cd-jgmsk
Your netApp has been successfully registered and onboarded to the CAPIF server.You can now start using the evolved5G SDK!
```

```

Initializing Database..
Netapp running..
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://10.244.1.172:5000
Press CTRL+C to quit
* Restarting with watchdog (inotify)

Initializing Database..
Netapp running..
* Debugger is active!
* Debugger PIN: 903-608-157

10.244.1.224 - - [30/Jun/2023 12:18:33] "GET /netapp HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:18:33] "GET /static/css/main.css HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:18:33] "GET /static/EVOLVED5G.png HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:18:33] "GET /static/8bells_research.png HTTP/1.1" 200 -
Working with ip: 10.0.0.1
Trying New QoS subscription with ip: 10.0.0.1
--- Subscribed to Qos successfully with id 649ec824afda33a15f30c2d9---
Trying New Location subscription with ip: 10.0.0.1
--- Subscribed to Location successfully with id 649ec825afda33a15f30c2da---
10.244.1.224 - - [30/Jun/2023 12:18:45] "POST /addip HTTP/1.1" 302 -
10.244.1.224 - - [30/Jun/2023 12:18:45] "GET /netapp HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:18:45] "GET /static/css/main.css HTTP/1.1" 304 -
10.244.1.224 - - [30/Jun/2023 12:18:45] "GET /static/EVOLVED5G.png HTTP/1.1" 304 -
Working with ip: 10.0.0.2
Trying New QoS subscription with ip: 10.0.0.2
--- Subscribed to Qos successfully with id 649ec82eafda33a15f30c2db---
Trying New Location subscription with ip: 10.0.0.2
--- Subscribed to Location successfully with id 649ec82fafda33a15f30c2dc---
10.244.1.224 - - [30/Jun/2023 12:18:55] "POST /addip HTTP/1.1" 302 -
10.244.1.224 - - [30/Jun/2023 12:18:55] "GET /netapp HTTP/1.1" 200 -
```

Figure 49. 8Bells Network Application runtime output log 1



```
Working with ip: 10.0.0.3
Trying New QoS subscription with ip: 10.0.0.3
--- Subscribed to QoS successfully with id 649ec834afda33a15f30c2dd----
Trying New location subscription with ip: 10.0.0.3
--- Subscribed to Location successfully with id 649ec835afda33a15f30c2de----
10.244.1.224 - - [30/Jun/2023 12:19:01] "POST /addip HTTP/1.1" 302 -
10.244.1.224 - - [30/Jun/2023 12:19:01] "GET /netapp HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:19:01] "GET /static/css/main.css HTTP/1.1" 304 -
10.244.1.224 - - [30/Jun/2023 12:19:01] "GET /static/EVOLVED5G.png HTTP/1.1" 304 -
New event notification retrieved:
New event notification retrieved:
10.244.1.48 - - [30/Jun/2023 12:19:10] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
10.244.1.48 - - [30/Jun/2023 12:19:10] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
sudo ovs-ofctl -O OpenFlow13 add-flow Firewall dl_type=0x0800,ip_src=10.0.0.2,priority=100,hard_timeout=360,actions=goto_table:100
sudo ovs-ofctl -O OpenFlow13 add-flow Firewall dl_type=0x0800,ip_src=10.0.0.1,priority=100,hard_timeout=360,actions=goto_table:100
QOS_GUARANTEED
10.244.1.48 - - [30/Jun/2023 12:19:12] "POST /monitoring/callback HTTP/1.1" 200 -
QOS_GUARANTEED
10.244.1.48 - - [30/Jun/2023 12:19:12] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
10.244.1.48 - - [30/Jun/2023 12:19:16] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
sudo ovs-ofctl -O OpenFlow13 add-flow Firewall dl_type=0x0800,ip_src=10.0.0.3,priority=100,hard_timeout=360,actions=goto_table:101
QOS_NOT_GUARANTEED
10.244.1.48 - - [30/Jun/2023 12:19:17] "POST /monitoring/callback HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:19:21] "GET /netapp HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:19:21] "GET /static/css/main.css HTTP/1.1" 304 -
10.244.1.224 - - [30/Jun/2023 12:19:21] "GET /static/EVOLVED5G.png HTTP/1.1" 304 -
10.244.1.224 - - [30/Jun/2023 12:19:21] "GET /static/8bells_research.png HTTP/1.1" 200 -
New event notification retrieved:
10.244.1.48 - - [30/Jun/2023 12:19:34] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
sudo ovs-ofctl -O OpenFlow13 add-flow Firewall dl_type=0x0800,ip_src=10.0.0.3,priority=100,hard_timeout=360,actions=goto_table:100
QOS_GUARANTEED
10.244.1.48 - - [30/Jun/2023 12:19:35] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
10.244.1.48 - - [30/Jun/2023 12:19:41] "POST /monitoring/callback HTTP/1.1" 200 -
New event notification retrieved:
10.244.1.224 - - [30/Jun/2023 12:19:41] "GET /netapp HTTP/1.1" 200 -
10.244.1.224 - - [30/Jun/2023 12:19:41] "GET /static/css/main.css HTTP/1.1" 304 -
10.244.1.224 - - [30/Jun/2023 12:19:41] "GET /static/EVOLVED5G.png HTTP/1.1" 304 -
10.244.1.224 - - [30/Jun/2023 12:19:41] "GET /static/8bells_research.png HTTP/1.1" 200 -
sudo ovs-ofctl -O OpenFlow13 add-flow Firewall dl_type=0x0800,ip_src=10.0.0.3,priority=100,hard_timeout=360,actions=goto_table:100
QOS_GUARANTEED
10.244.1.48 - - [30/Jun/2023 12:19:42] "POST /monitoring/callback HTTP/1.1" 200 -
```

Figure 50. 8Bells Network Application runtime output log 2

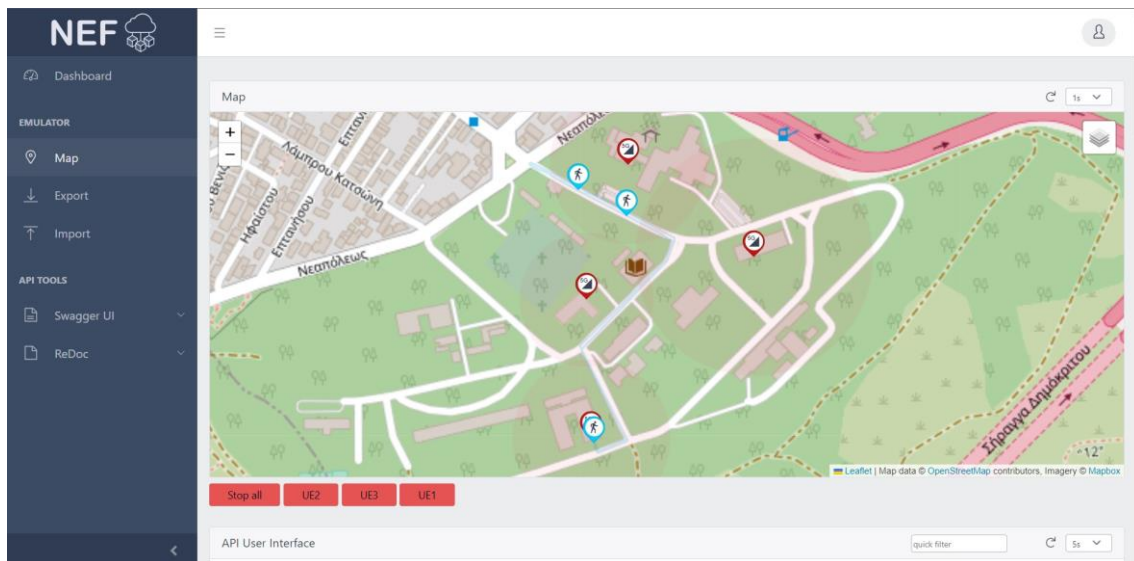


Figure 51. NEF deployment topology scenario for 8BELLS Network Application

The scenario consists of 4 cells and 3 UEs located in the campus. The UEs move on a predetermined path with different speeds to simulate different outcomes.

The following Figure depicts a live representation of the 8BELLS Network App, where the 'History' table captures the UE subscriptions, and NEF notifications that show QoS status & location. The 'IP Table' shows the UEs that are registered in Network App, while those that have 'allowed' status are also configured in the vApp.

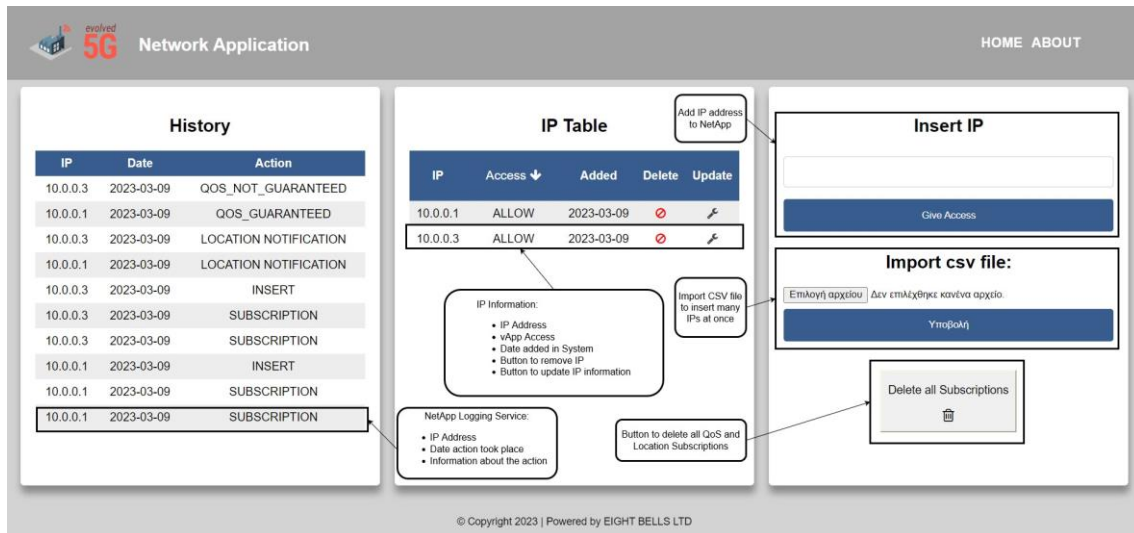


Figure 52. 8BELLS Network Application high level overview of front-end interworking with all components (NEF & CAPIF)

#### 4.6.2 5G SIEM Network Application

After the successful deployment of CAPIF, NEF and FOGUS network application in NCSRDKubernetes platform, the functionality of the network application and the end-to-end communication with the Vertical Application (OSSIM) has been tested. The initial step was to access the frontend of FOGUS network application at the following url: “https://fogusnetapp-frontend.com”. Due to the fact that the Kubernetes platform exposed the ports inside NCSRDKlocal network, the use of a VPN was required to enter on it remotely. Having access to FOGUS frontend application, FOGUS scenario has been imported to NEF emulator and then some Monitoring Location requests from the Network Application were performed, that are depicted on the figures below.

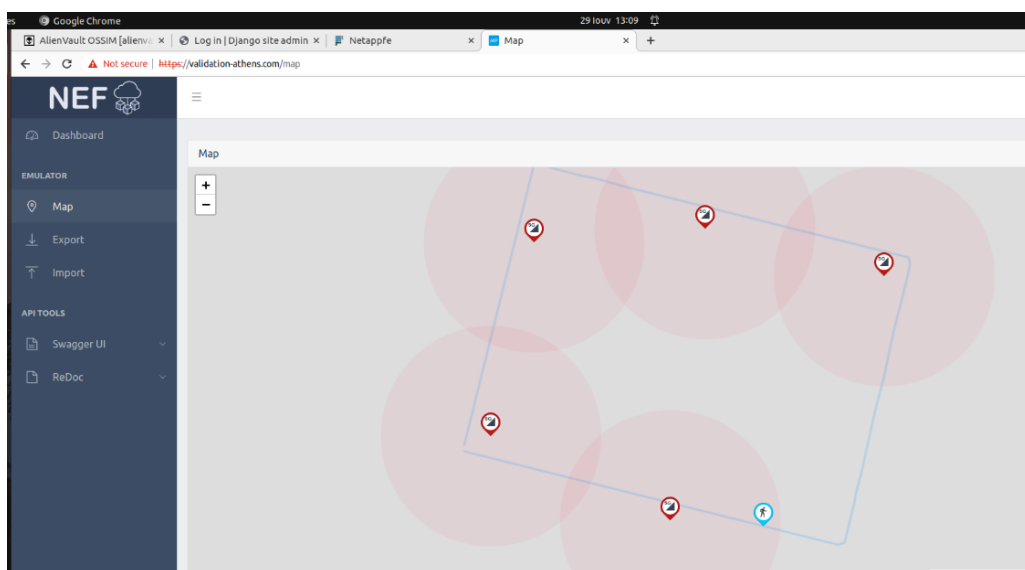
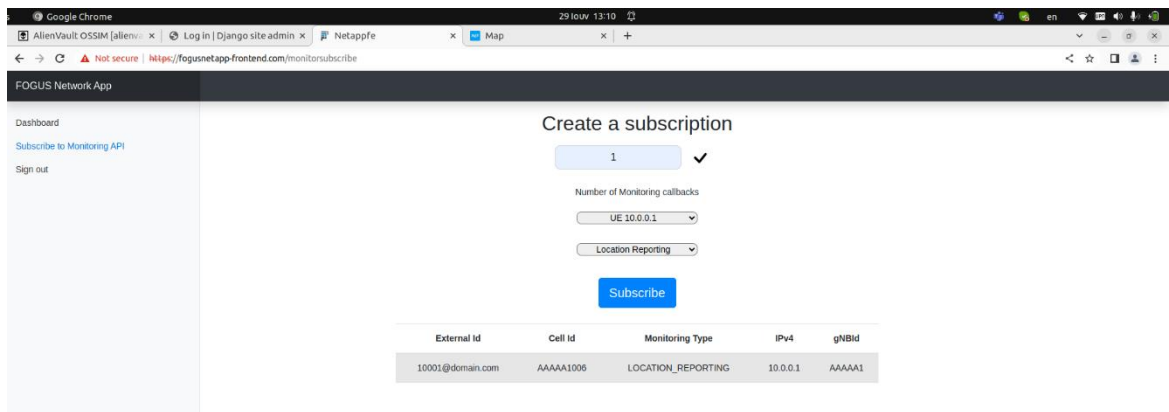


Figure 53. Import FOGUS scenario in NEF Emulator





External Id	Cell Id	Monitoring Type	IPv4	gNBId
10001@domain.com	AAAAA1006	LOCATION_REPORTING	10.0.0.1	AAAAA1

Figure 54. Succesful one time Location Monitoring subscription

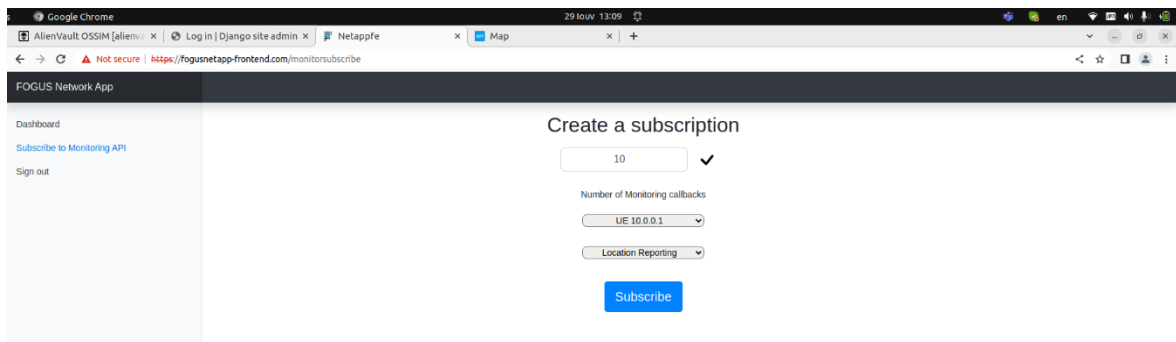
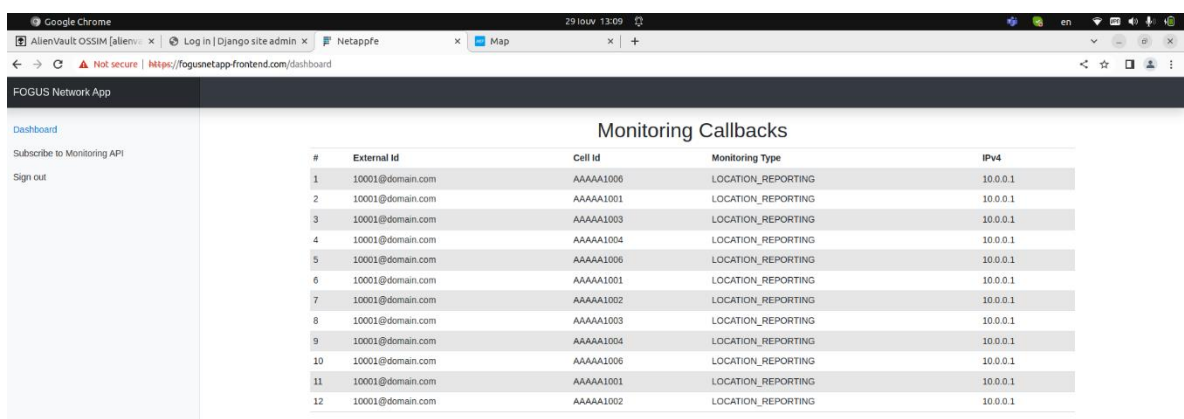


Figure 55. Succesful multiple times Location Monitoring subscription



#	External Id	Cell Id	Monitoring Type	IPv4
1	10001@domain.com	AAAAA1006	LOCATION_REPORTING	10.0.0.1
2	10001@domain.com	AAAAA1001	LOCATION_REPORTING	10.0.0.1
3	10001@domain.com	AAAAA1003	LOCATION_REPORTING	10.0.0.1
4	10001@domain.com	AAAAA1004	LOCATION_REPORTING	10.0.0.1
5	10001@domain.com	AAAAA1006	LOCATION_REPORTING	10.0.0.1
6	10001@domain.com	AAAAA1001	LOCATION_REPORTING	10.0.0.1
7	10001@domain.com	AAAAA1002	LOCATION_REPORTING	10.0.0.1
8	10001@domain.com	AAAAA1003	LOCATION_REPORTING	10.0.0.1
9	10001@domain.com	AAAAA1004	LOCATION_REPORTING	10.0.0.1
10	10001@domain.com	AAAAA1006	LOCATION_REPORTING	10.0.0.1
11	10001@domain.com	AAAAA1001	LOCATION_REPORTING	10.0.0.1
12	10001@domain.com	AAAAA1002	LOCATION_REPORTING	10.0.0.1

Figure 56. List of returned callbacks

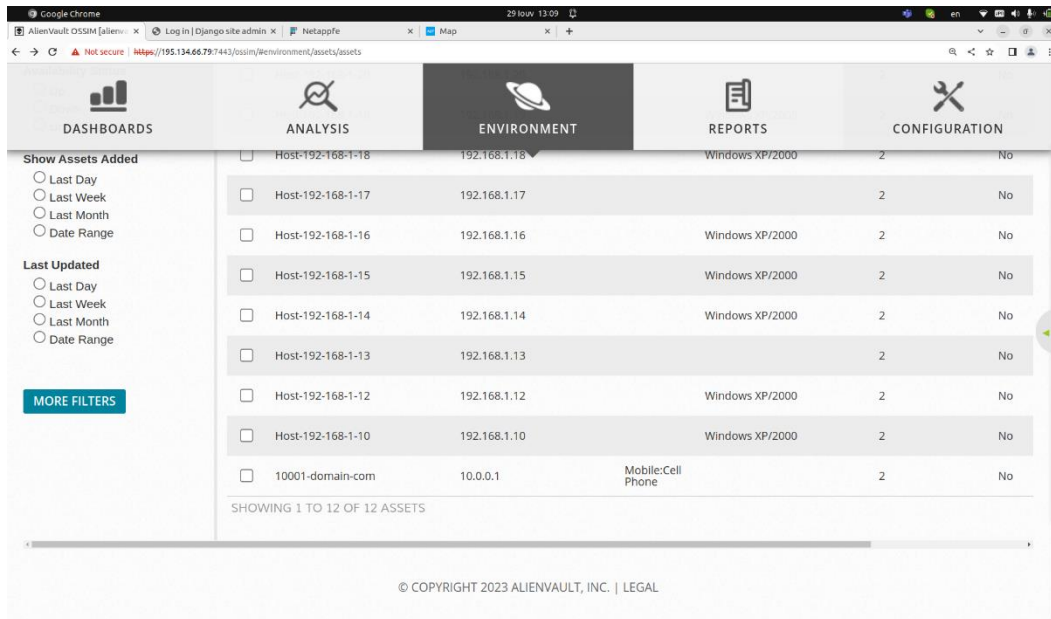


Figure 57. UE presented in list of OSSIM assets

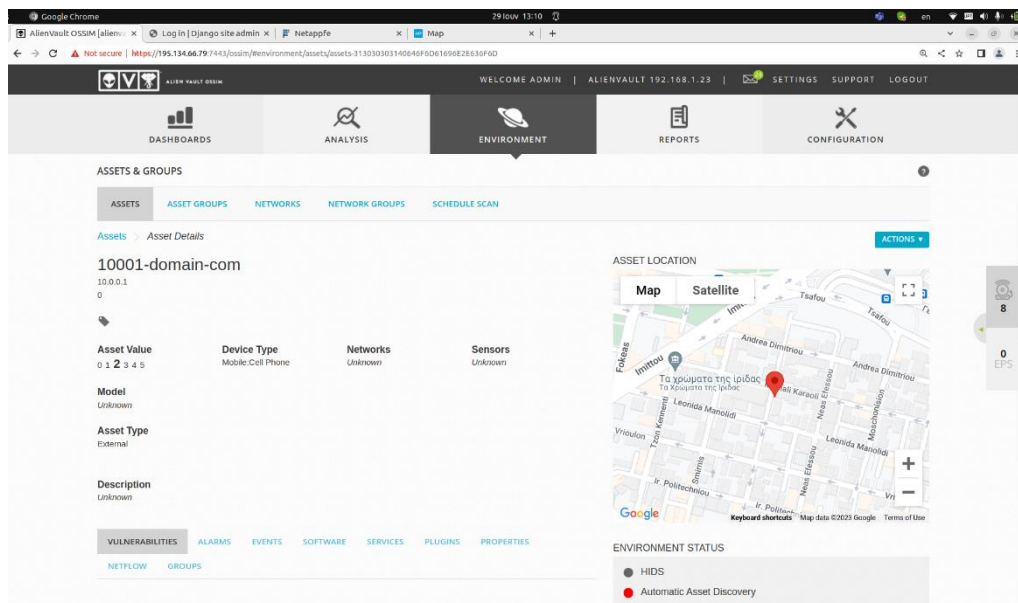


Figure 58. Location of UE depicted in OSSIM

### 4.6.3 Identity and Access Management Network Application

After the successful deployment of CAPIF, NEF, IQB's Network App, Keycloak and the callbacks server in the NCSRD Kubernetes platform, the functionality of the network application and the proper communication of the components had to be tested. First unit tests were run by executing a shell command inside the pod. Then, further testing was performed using Postman towards the url: "https://iqbitnetapp.com". Finally, a Monitoring subscription was set up in order to test the proper reception of callbacks upon the UEs cell change. The following Figures present the aforementioned process regarding the unit tests and the testing of the end points respectively.

```

root@kali: ~# curl -i http://10.10.10.10:8080/
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 16384
Date: Mon, 11 Jun 2024 10:10:10 GMT
Server: Apache/2.4.18 (Ubuntu)
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Permitted-Cross-Domain-Policies: none
X-Request-Id: 1a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6q7r8s9t0u1v2w3x4y5z6a7b8c9d0e1f2g3h4i5j6k7l8m9n0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d9e0f1g2h3i4j5k6l7m8n9o0p1q2r3s4t5u6v7w8x9y0z1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f3g4h5i6j7k8l9m0n1o2p3q4r5s6t7u8v9w0x1y2z3a4b5c6d7e8f9g0h1i2j3k4l5m6n7o8p9q0r1s2t3u4v5w6x7y8z9a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x
```

Figure 59. IQB UnitTests

[illegible]

Figure 60. IQB Endpoints Testing

The following actions were also tested:

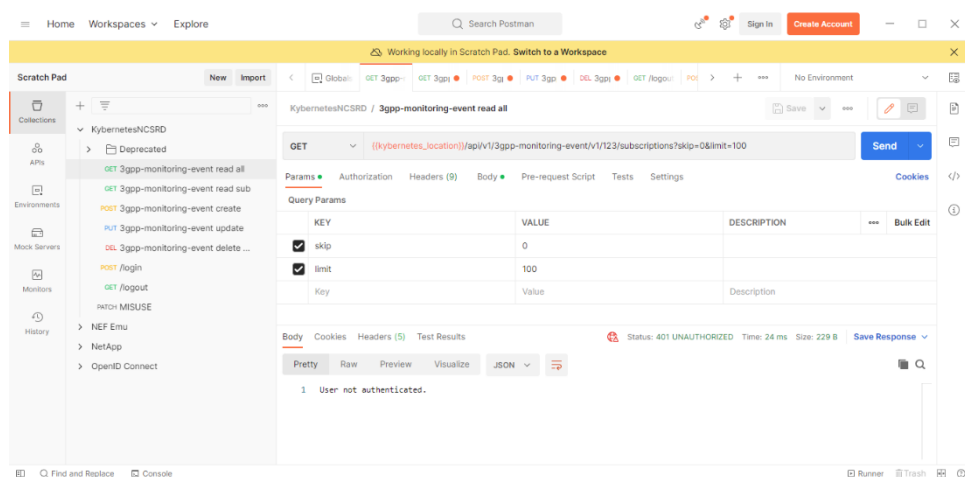


Figure 61. IQB Unauthorized API consumption

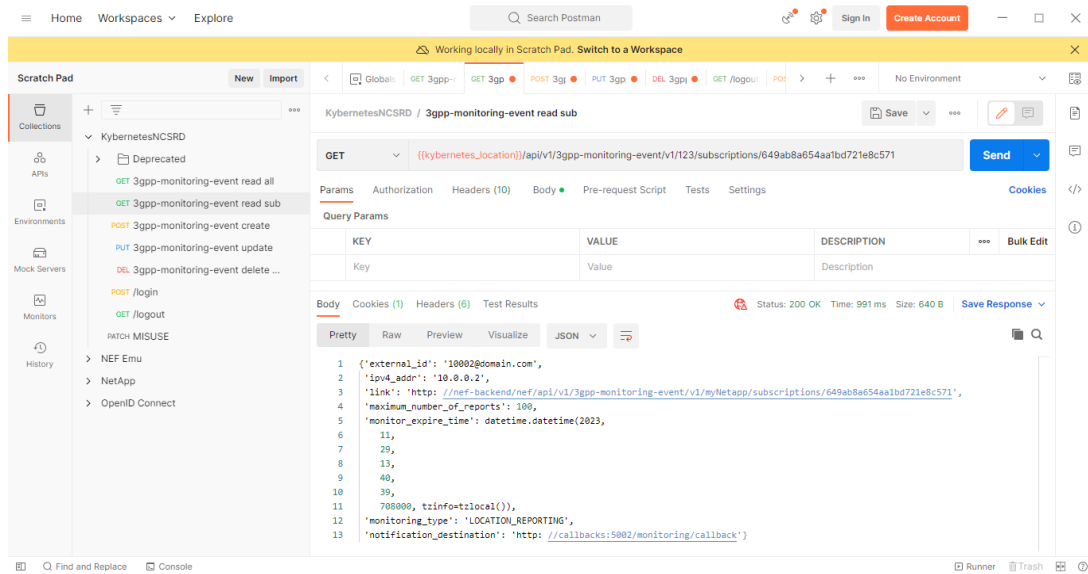


Figure 62. IQB Reading specific subscription

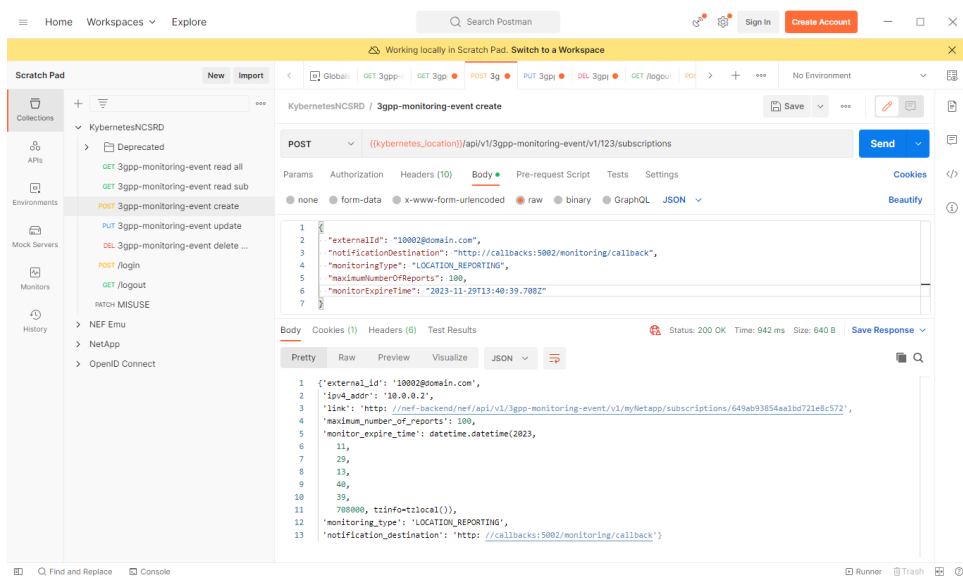


Figure 63. IQB Creating a subscription

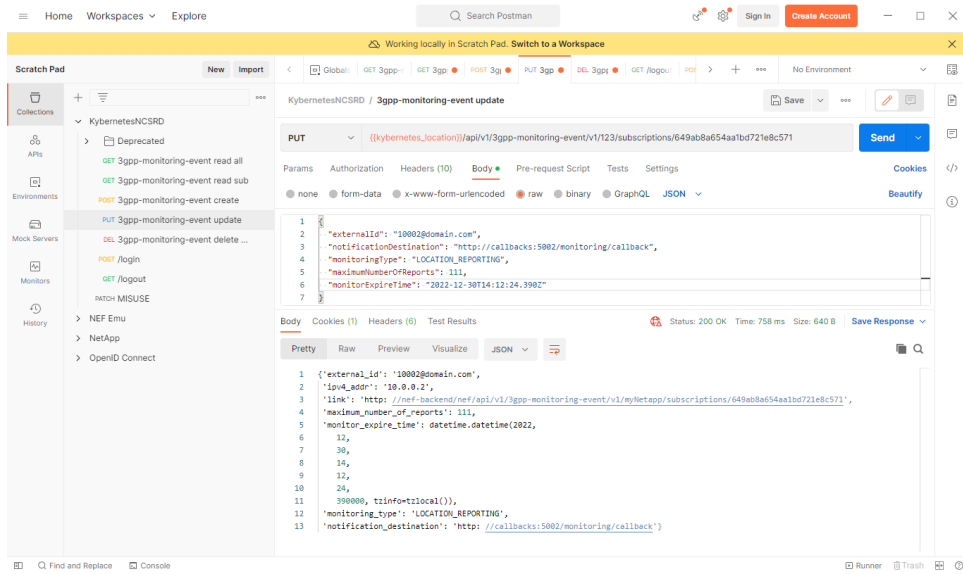


Figure 64. IQB Updating a subscription

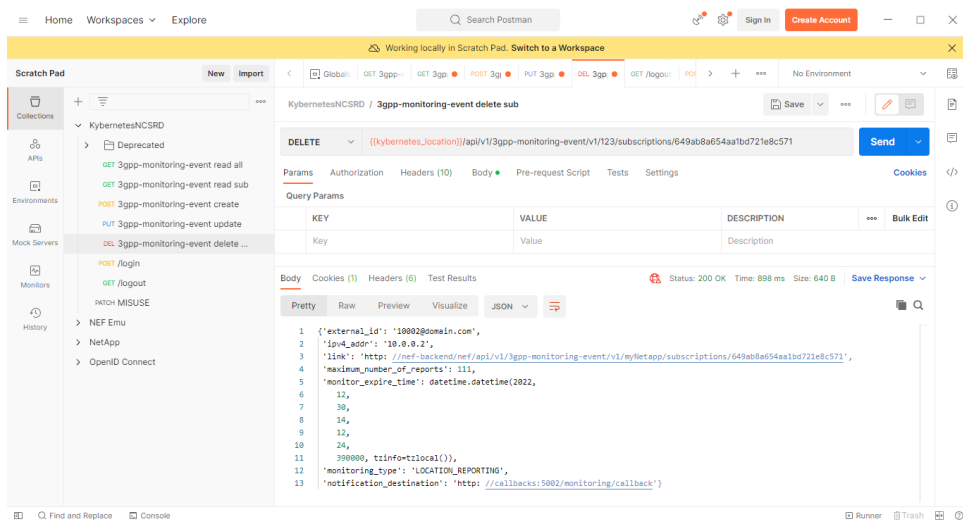
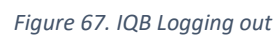


Figure 65. IQB Deleting specific subscription



45

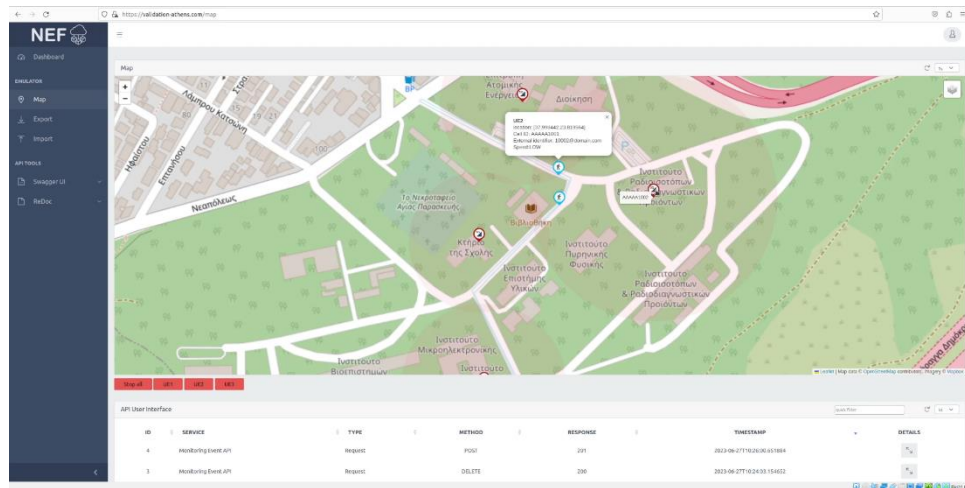


Figure 69. IQB The UE is about to leave cell 10001

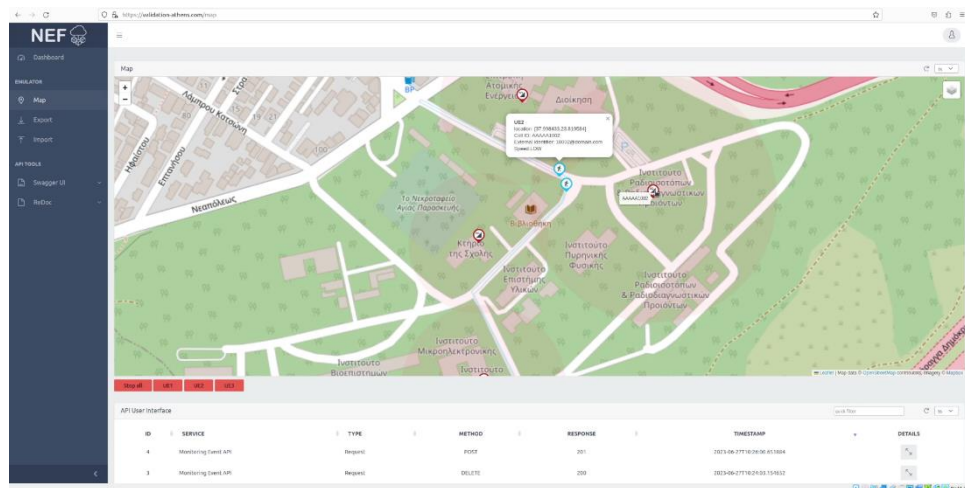


Figure 70. IQB The UE connects to cell 10002

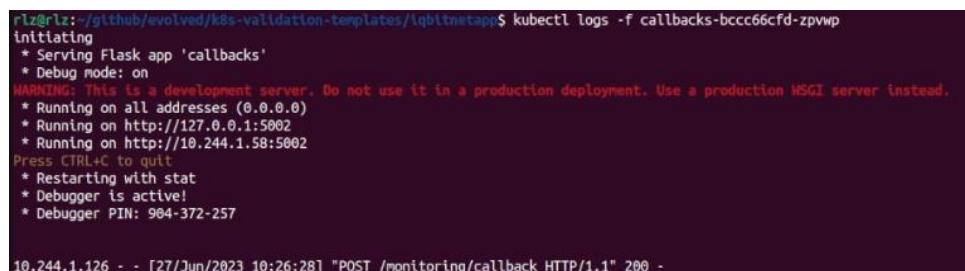


Figure 71. IQB The callback for cell change is received



## 5 CONCLUSION AND NEXT STEPS

---

The work presented in this deliverable describes in detail the final prototype of the Network Apps developed within the Security Guarantees and Risk Analysis pillar in the EVOLVED-5G context, driven by Task 4.4. Moreover, detailed descriptions of the two iterations of integration tests that the Network Apps have undergone on top of the EVOLVED-5G infrastructure, specifically on the Athens platform, are provided. With the second round of integration tests, the Networks Apps of the SEC pillar have reached their final stage, interacting with the last versions of NEF and CAPIF through the SDK and communicating with their respective vApp(s). The three SME use-cases have been also validated and such results highlight the fact that the Network Apps reached a mature enough state to be used by other SMEs through the Evolved-5G Marketplace.

The next steps will take place within the scope of WP5. SMEs have already started to use the validation pipeline to check their Network Apps. When they pass the final validation and certification steps, the Network Apps will be ready to be made publicly available through the EVOLVED-5G Marketplace. This way, other professionals and researchers will be able to use these applications, learn from them and build their own following the EVOLVED-5G methodology and pipelines.

### **Network App1: Traffic Management**

The 8BELLS Traffic Management application extends the functionality of a standard firewall by also taking into account any parameters that can be exposed by the 5G network. By leveraging these parameters, we can implement advanced security policies towards a Zero-Trust approach.

As demonstrated, the integration of a firewall with the 5G network control plane can greatly enhance the functionality and usefulness, for example by taking into account 5G network conditions (e.g., high traffic, congestion, etc.) and adjusting the firewall rules accordingly.

In the future, leveraging additional parameters exposed by the 5G network APIs, could create more complex security policies to cater to any Industry 4.0 use case. Such use cases could for example take into account the user device location within the 5G network in applying customized security rules.

During the two rounds of integration activities that took place, as described in the relevant sections previously, it was also proven that the overall deployment is flexible, quick, easy to test, and can easily be upgraded (hot fixes), due to the dockerization of the Network Application and the Kubernetes deployment environment.

### **Network App2: Secure and trusted event management system**

FOGUS SIEM platform (OSSIM) offers a solution for security and event monitoring in a network infrastructure. Extending the SIEM system with 5G capabilities (by adding some plugins to SIEM platform and implementing the Network Application) gave the SIEM system the opportunity to access 5G security information, such as real-time location monitoring of UE and feedback on the security status of the 5G NPN devices, through the native 5G APIs.

Also, the increased network performance of the 5G network (e.g., in latency and bandwidth) enables the faster acquisition of this information. As a result, FOGUS, with the development of its Network Application, managed to bridge the communication gap between SIEM and 5G NPN



devices, thus offering a more complete security management solution for modern unified networks (including Ethernet, Wi-Fi, 5G etc.).

### **Network App3: IQB Identity and Access Management**

The IQB Network application provides an enhancement to the existing CAPIF protocol by incorporating an OpenID Connect layer on top of the OAuth2.0 protocol proposed by CAPIF. Furthermore, the solution provides single sign-on functionality between CAPIF instances. Due to the increased performance of the 5G network, this added security value can be incorporated without noticeable overhead on the speed of execution of the network processes. The solution is an easy-to-deploy and scalable solution since it is a containerized implementation.

## **6 BIBLIOGRAPHY**

---

- [1] Parks Associates, «Overcoming Roadblocks to Industry 4.0,» 2022.
- [2] 3GPP, «Release 15,» 2019. [En línea]. Available: <https://www.3gpp.org/specifications-technologies/releases/release-15>.
- [3] 3GPP, «Location and positioning,» 2022. [En línea]. Available: <https://www.3gpp.org/technologies/location-and-positioning>.