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

[H2020 - Grant Agreement No.101016608]

Deliverable D5.6

Network App Certification and Release to Marketplace (Final)

Editor	David Artuñedo (TID)
Contributors	FOGUS, ATOS, NCSRD, INF, MAG, UMA

Version	1.0
Date	December 31, 2024
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 case, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced. In the event of infringement, the consortium reserves the right to take any legal action it deems appropriate.

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

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

REVISION HISTORY

Revision	*Date*	*Responsible*	*Comment*
0.1	*Oct 18th 2023*	*David Artuñedo (TID)*	*Edit ToC*
0.9	*Nov 28th 2023*	*David Artuñedo (TID)*	*First draft with inputs from all partners*
1.0	*Dec 31st 2023*	*David Artuñedo (TID)*	*Final version*

LIST OF AUTHORS

Partner ACRONYM	*Partner FULL NAME*	*Name & Surname*
TID	*TELEFONICA INVESTIGACIÓN Y DESARROLLO*	*Javier Garcia* *David Artuñedo* *Jorge Moratinos*
ATOS	*ATOS IT SOLUTIONS AND SERVICES IBERIA SL*	*Ricardo Marco* *Sonia Castro*
NCSRD	*NATIONAL CENTER FOR SCIENTIFIC RESEARCH "DEMOKRITOS*	*Harilaos Koumaras* *George Makropoulos* *Dimitrios Fragkos*
INF	*INFOLYSIS P.C.*	*Christos Sakkas* *George Theodoropoulos* *Antonios Varkas*
FOGUS	FOGUS INNOVATIONS & SERVICES P.C.	*Dimitris Tsolkas* *Georgios Krommydas* *Anastasia Papafotiou*
MAG	*MAGGIOLI*	*Yiannis Karadimas*
UMA	*UNIVERSITY OF MÁLAGA*	*Almudena Diaz* *Bruno García* *Francisco Luque* *Carlos Andreo* *Mª del Mar Moreno*

GLOSSARY

Abbreviations/Acronym	*Description*
3GPP	*3rd Generation Partnership Project*
API	*Application Programming Interface*
CA	*Certification Authority*
CAPIF	*Common Application Programming Interface Framework*
CI/CD	*Continuous Integration / Continuous Deployment*
IaC	*Infrastructure as Code*
K8s	*Kubernetes*
MD	*Markdown*
Network App	*Network Application*
RPA	*Robotic Process Automation*
TSN	*Time Sensitive Networking*
UI	*User Interface*
URL	*Uniform Resource Locator*
UX	*User Experience*
VPN	*Virtual Private Network*

EXECUTIVE SUMMARY

EVOLVED-5G responds to the *5G PPP ICT-41-2020 5G innovations for verticals with third party services* call, whose main goal is to deliver enhanced experimentation facilities on top of which third party experimenters (e.g., SMEs or any service provider and target vertical users) will have the opportunity to test their applications.

The EVOLVED-5G project accomplishes this vision by encouraging the creation of a Network App ecosystem revolving around a 5G facility which will provide the tools and processes for the development, verification, validation, and certification of Network Apps as well as their validation on top of actual 5G network infrastructures, and mechanisms for market releasing.

This deliverable provides a comprehensive overview of the EVOLVED-5G Certification Environment and Certification Process, focusing on the implementation details and the release mechanism for Network Applications to the marketplace. The document outlines the steps involved in the Certification Process, including the use of certification tools.

Section 2 provides an update on the certification tools originally presented in Deliverable 5.3 [1]. Additionally, it provides enhancements to the description of the Certification Environment, where the Certification Process occurs. This includes updates on the implementation of the Certification Pipeline, designed to automate the certification tests. Finally, the section discusses the Certification report generated at the conclusion of the process.

Section 3 emphasizes the EVOLVED-5G Marketplace as the central platform for interaction between Network App creators and consumers. The "Network App Onboarding" process is exposed, describing the steps for releasing a Network App to the marketplace.

The deliverable concludes in Section 4, marking the final output from Work Package 5 (WP5) in the EVOLVED-5G project. The document encapsulates the latest information on the Certification Process, the production of Certification Reports, and the release of Network Apps to the marketplace, thereby offering a comprehensive understanding of the project's key components and processes.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 PURPOSE OF THE DOCUMENT

The ultimate objective of EVOLVED-5G project and WP5, more specifically, is to Certify the Network Applications developed in the Project and release them to the Marketplace. Thus, the current document *"**Network App Certification and Release to Marketplace*** (Final)" provides details on the Certification Process defined in EVOLVED-5G It outlines the establishment of a Certification Pipeline for implementation and delves into the subsequent Release of certified Network Applications to the Marketplace Moreover, the document provides details on each of the certification tests implemented and the tools used to build these tests, the generation of the Certification Report and the integration with the Open Repository, and finally, describes the lifecycle of the Network Applications in the Marketplace, describing the Release process, the upgrades and deprecation of Network Applications.

1.2 STRUCTURE

This deliverable is organized in the following manner:

- **Section 1. Introduction:** This section describes the deliverable target audience, objectives, and structure.

- **Section 2. Certification Process:** This section describes the Certification Environment focusing on the CI/CD toolset with a collection of software industry leading tools for automation, and the Certification Pipeline that automates all the Certification tests. In addition, this section clarifies the process of generating the Certification Report and provides an overview of its content.

- **Section 3. Release to Marketplace:** This section describes how to release a certified Network Application to the Marketplace and provides a guidance on the management of post publication.

1.3 TARGET AUDIENCE

The release of the deliverable is public, intending to expose the overall EVOLVED-5G ecosystem and Network Apps' Lifecycle design to a wide variety of research individuals and communities. From specific to broader, different target audiences for D5.2 are identified as detailed below:

- **Project Consortium:** To validate that all objectives and proposed technological advancements have been analysed and to ensure that, through the proposed Network App's lifecycle phases and the various environments, further work can be concretely derived. Furthermore, the deliverable sets to establish a common understanding among the consortium with regards to:
  - The validation framework used within the EVOLVED-5G platforms for the management and orchestration of the resources and the procedures in the testbeds.
  - The validation process of the Network Apps is realised by the several tools that are geared towards the automation of the process.
- **Industry 4.0 and FoF (Factories of the Future) vertical groups:** To crystallise a common understanding of technologies, and design principles that underline the development of the Network Apps, and to understand the utilisation of the network Application Programmable

Interfaces (APIs) exposed by the 5G Infrastructure. A non-exhaustive list of Industry 4.0-related groups is as follows:

  - Manufacturing industries (including both large and Small Medium Enterprise (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.

- **Telecom Service Providers:** to engage with verticals and to simplify the way 5G services can be offered to a potential customer or 3rd party service provider.
- **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 the validation process of the Network Apps developed by the project.

2 Certification Process

This section updates the description of the certification tools, that was initially provided in Deliverable 5.3 [1] which are utilized during the Certification Process. It also updates the Certification Environment description in which the Certification Process takes place, the final implementation of the Certification Pipeline built to automate the Certification tests, and finally, the Certification report produced at the end of the process.

2.1 Certification tools

The categories for the Certification tools have not changed from Deliverable 5.3 [1], but some tools have been updated due to license restrictions (e.g., Debriked). The final tools used during Certification are as follows:

- **Software product quality:** SonarQube, Robot Framework and Nmap.
- **Security**: Trivy and SonarQube.
- **Licensing**: Licensecheck.

Additionally, some KPIs have been added to the Certification Report. These KPIs are collected using Prometheus tool and Kubernetes as described in section 2.1.6.

2.1.1 Trivy

Trivy tool has been described in Deliverable 5.3 [1]. There are no additional updates on using this tool.

2.1.2 SonarQube (TID)

SonarQube tool has been described in Deliverable 5.3 [1]. There are no additional updates on using this tool.

2.1.3 Robot Framework

Robot Framework tool has been described in Deliverable 5.3 [1]. There are no additional updates on using this tool.

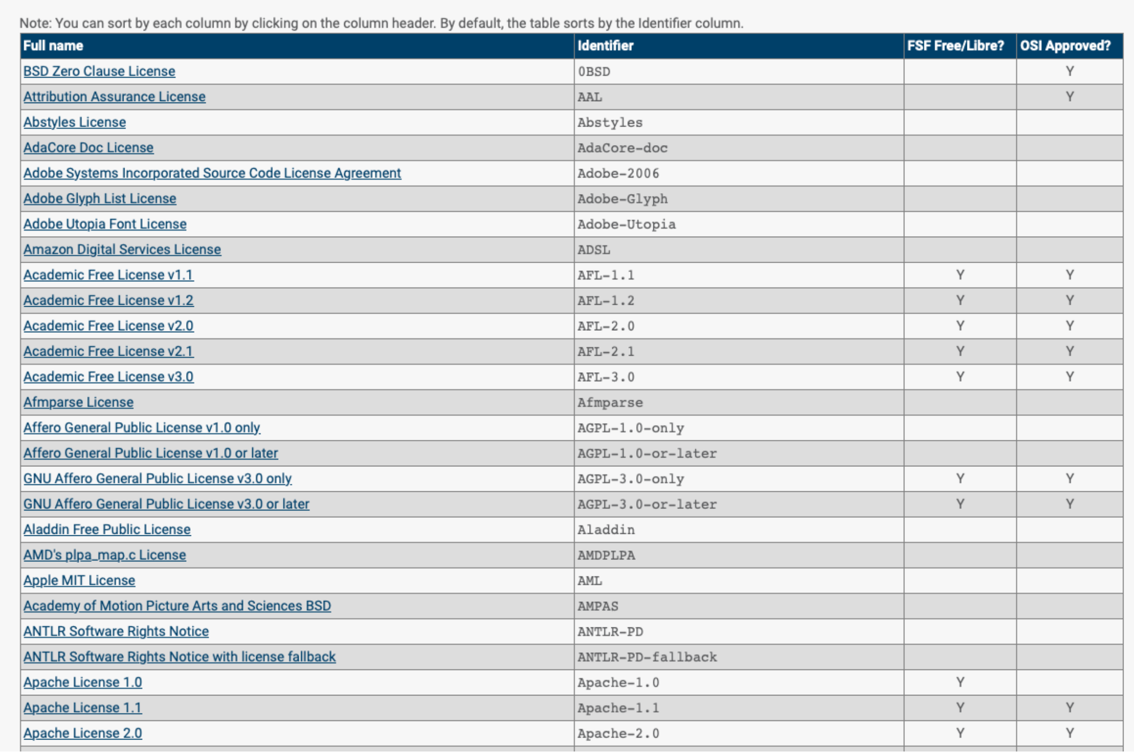
2.1.4 Nmap

NMAP tool has been described in Deliverable 5.3 [1]. There are no additional updates on using this tool.

2.1.5 Licensecheck (TID)

Licensecheck [2] is the new tool selected to replace Debriked. Debriked has been replaced due to license changes in their product. Initially, EVOLVED-5G was using Debriked free cost solution consuming Debriked APIs. The change in Debriked license broke this model, and changed the free cost model to an offline process that was not suitable to be integrated in the automation model that EVOLVED-5G has created for Certification of Network Applications.

Instead of Debriked, we are now using Licensecheck. This is an open-source tool written in Go, that checks for licenses described in Licensecheck License Database. This Database is customizable, but we have used the Licenses information provided by Licensecheck by default, which is based in SPDX Project from Linux Foundation [3]. The SPDX License list can be consulted in the following URL: https://spdx.org/licenses/.

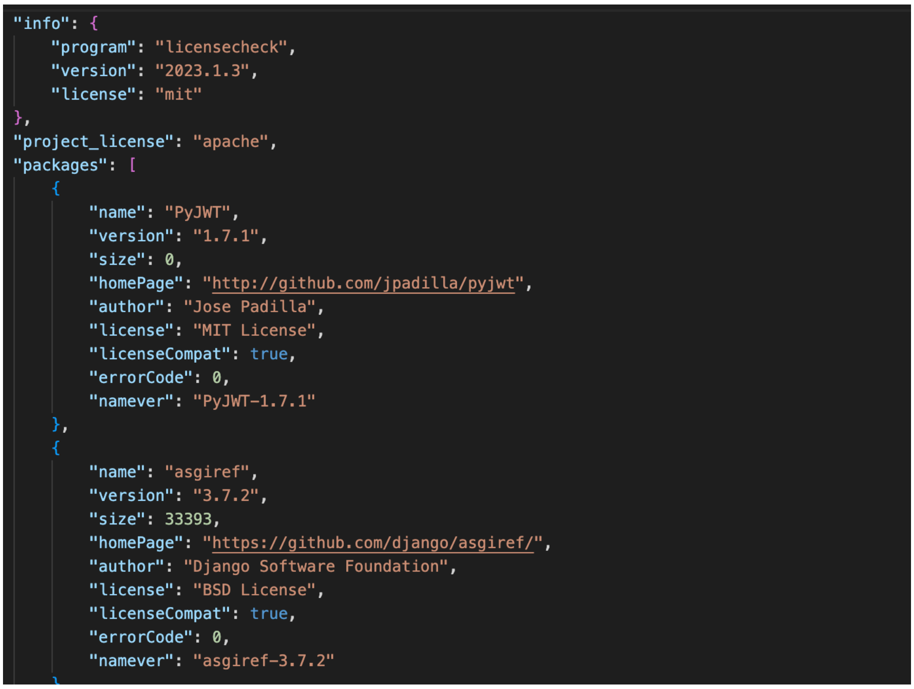


Note: You can sort by each column by clicking on the column header. By default, the table sorts by the Identifier column.

Full name	Identifier	FSF Free/Libre?	OSI Approved?
BSD Zero Clause License	0BSD		Y
Attribution Assurance License	AAL		Y
Abstyles License	Abstyles		
AdaCore Doc License	AdaCore-doc		
Adobe Systems Incorporated Source Code License Agreement	Adobe-2006		
Adobe Glyph List License	Adobe-Glyph		
Adobe Utopia Font License	Adobe-Utopia		
Amazon Digital Services License	ADSL		
Academic Free License v1.1	AFL-1.1	Y	Y
Academic Free License v1.2	AFL-1.2	Y	Y
Academic Free License v2.0	AFL-2.0	Y	Y
Academic Free License v2.1	AFL-2.1	Y	Y
Academic Free License v3.0	AFL-3.0	Y	Y
Afmparse License	Afmparse		
Affero General Public License v1.0 only	AGPL-1.0-only		
Affero General Public License v1.0 or later	AGPL-1.0-or-later		
GNU Affero General Public License v3.0 only	AGPL-3.0-only	Y	Y
GNU Affero General Public License v3.0 or later	AGPL-3.0-or-later	Y	Y
Aladdin Free Public License	Aladdin		
AMD's plpa_map.c License	AMDPLPA		
Apple MIT License	AML		
Academy of Motion Picture Arts and Sciences BSD	AMPAS		
ANTLR Software Rights Notice	ANTLR-PD		
ANTLR Software Rights Notice with license fallback	ANTLR-PD-fallback		
Apache License 1.0	Apache-1.0	Y	
Apache License 1.1	Apache-1.1	Y	Y
Apache License 2.0	Apache-2.0	Y	Y

Figure 1: Example of Licenses included in SPDX

Licensecheck generates a report in JSON format with the results of the analysis. Figure 2 shows how the License information is collected in JSON format:



```
"info": {
    "program": "licensecheck",
    "version": "2023.1.3",
    "license": "mit"
},
"project_license": "apache",
"packages": [
    {
        "name": "PyJWT",
        "version": "1.7.1",
        "size": 0,
        "homePage": "http://github.com/jpadilla/pyjwt",
        "author": "Jose Padilla",
        "license": "MIT License",
        "licenseCompat": true,
        "errorCode": 0,
        "namever": "PyJWT-1.7.1"
    },
    {
        "name": "asgiref",
        "version": "3.7.2",
        "size": 33393,
        "homePage": "https://github.com/django/asgiref/",
        "author": "Django Software Foundation",
        "license": "BSD License",
        "licenseCompat": true,
        "errorCode": 0,
        "namever": "asgiref-3.7.2"
```

Figure 2: Extract of JSON format from Licensecheck results

As this file shows, for each package a License type is specified (MIT, Apache, BSD, …). This information is used by the Certification Pipeline to produce the License information in the Certification Report as described in section 2.3.4.

2.1.6 Prometheus (TID)

Prometheus [4] is an open-source technology designed to provide monitoring and alerting functionality for cloud-native environments, including Kubernetes. It can collect and store metrics as time-series data, recording information with a timestamp. It can also collect and record labels, which are optional key-value pairs.

Prometheus includes:

- various modes of graphing and dashboard support;
- the occurrence of time series collection through a pull model over HTTP;
- a multidimensional data model featuring time series data that is identified with a metric name or with key-value pairs (KVP);
- the ability to use PromQL to support the multidimensionality of the data model;
- autonomous single server nodes and zero reliance on distributed storage;
- discovery of the target through static configuration or service discovery; and
- the ability to push time series through an intermediary gateway.

EVOLVED-5G uses Prometheus to collect metrics from the Network Applications deployed in Kubernetes. We collect metrics at two levels: Namespace (regardless the number of nodes involved) and Pods.

Namespace metrics provide information about the Network Application as a whole regardless of it being deployed across several nodes.

Pods provide information for each container in the Network Application. This way, the developer has information for both the Network Application itself and each of the containers in the Network Application. Prometheus time series values can be queried for specific KPIs. These are the Queries defined in EVOLVED-5G:

CPU Query: this query gets the usage of CPU for the Namespace of the Network Application

```
        "cpu" : {
            "unit": '%',
            "query":'100 * sum (rate (container_cpu_usage_seconds_tota
l {{namespace="{namespace}"}} [5m])) by (instance) / on (instance) mac
hine_cpu_cores'.format(namespace=namespace),
            "type": 'instance'
        }
```

Memory Query: this query gets the amount of memory consumed by the Namespace of the Network Application

```
        "memory"  :  {
            "unit": "%",
            "query": '100 * sum (container_memory_working_set_bytes {{
namespace="{namespace}"}}) by (instance) / on (instance) machine_memor
y_bytes'.format(namespace=namespace),
            "type": 'instance'
        }
```

Memory Usage Query: this query gets the usage of CPU for the Pods (containers) of the Network Application

```
        "memory_usage": {
            "unit": "%",
            "query": 'sum (container_memory_working_set_bytes {{namesp
ace="{namespace}"}}) by (pod) / sum (kube_pod_container_resource_limit
s{{resource="memory", namespace="{namespace}"}}) by (pod)'.format(name
space=namespace),
            "type": 'pod'
        }
```

Network I/O Query: this query gets the Network I/O activity for the Pods (containers) of the Network Application

```
        "net_i/o": {
            "unit": "Bytes",
            "query": 'sum by (pod) (rate (container_network_receive_by
tes_total {{namespace="{namespace}"}} [1m])) + sum by (pod) (rate (con
tainer_network_transmit_bytes_total {{namespace="{namespace}"}} [1m]))
'.format(namespace=namespace),
            "type": 'pod'
        }
```

Memory Failures Query: this query gets the number of times a container has get the Out-of-memory event

```
        "mem_failures": {
            "unit": "Times",
            "query": 'sum by (pod) (changes (container_memory_oom_tota
l {{namespace="{namespace}"}} [5m]))'.format(namespace=namespace),
            "type": 'pod'
        }
```

Block I/O Query: this query gets the amount of data read/written by the container from block devices

```
        "block_i/o": {
            "unit": "Blocks",
            "query": 'sum by (pod) (rate (container_fs_writes_bytes_to
tal {{namespace="{namespace}"}} [1m])) + sum by (pod) (rate (container
_fs_reads_bytes_total {{namespace="{namespace}"}} [1m]))'.format(names
pace=namespace),
            "type": 'pod'
        }
```

2.2 Certification Environment

2.2.1 Malaga Certification Environment

2.2.1.1 Kubernetes Containerised Infrastructure

The Kubernetes cluster in Malaga is multi-master and multi-worker in order to provide high resource availability and fault tolerance. Specifically, it is defined by three master nodes, three worker nodes and a storage node for dynamic storage as shown in Figure 3*Figure 3*

The master nodes are responsible for handling the cluster control operations. Being multi-master, if one node fails, one of the other two nodes can take over the responsibility, avoiding data loss.

The worker nodes are responsible for executing the application workloads. As they are also multi-node, they allow the distribution of workloads among several nodes, improving efficiency and scalability. The storage node also functions as a worker node.

Finally, an HA Proxy acts as a load balancer and manages network traffic on a broader level. It is able to assign a task to a specific nod.

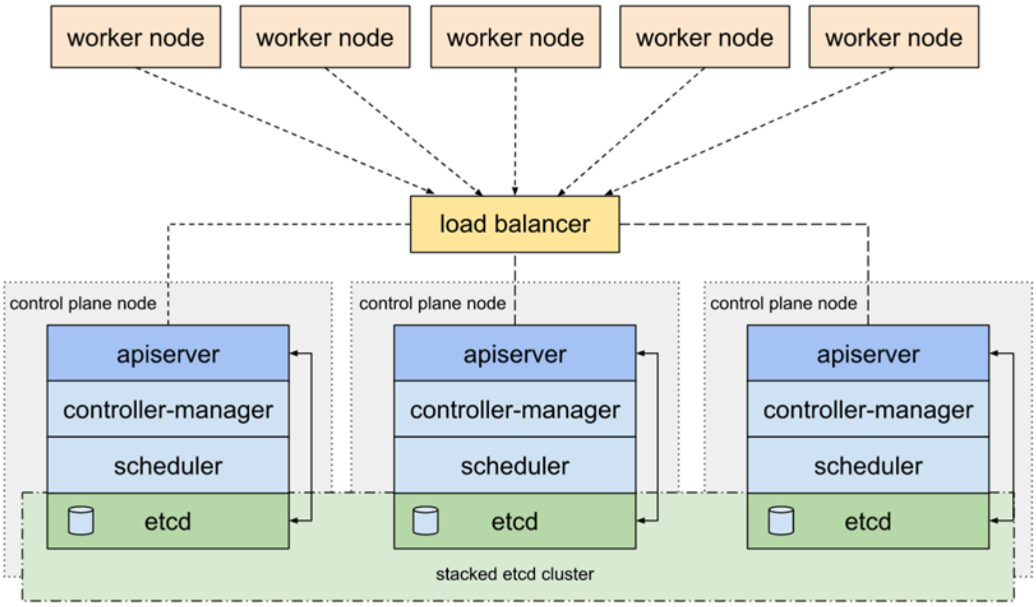


Figure 3: Multi-master and multi-worker Kubernetes Infrastructure in Malaga

The Kubernetes cluster in Malaga is composed of different tools:

- **Namespaces** are a way to organize the cluster into different work areas. Each SME that has deployed in the Malaga cluster has its own namespace isolated from the others. In it, they have deployed their Network Application and, in some cases, the Vertical Application. Access to the namespace is managed through the creation of token. Therefore, the administrator needs to generate a ServiceAccount, Secret, Role and RoleBinding associated to the namespace. Once authenticated with the token, SME's can deploy their Network Application by sending yaml files to the cluster using kubectl.
- **MetalLB** is useful when you need to provide services accessible from outside the Kubernetes cluster. An address pool is established within the MetalLB configmap. Network Applications and Vertical Applications need to be accessible from outside (using VPN) so when applying the service, they must indicate that it is of type LoadBalancer.
- **Calico** is a network policy engine for Kubernetes. It facilitates network connectivity and security policy enforcement between containers by establishing a solution that allows routing traffic between containers in the cluster.
- **Kadalu** is a storage solution for Kubernetes that is used to provide persistent storage for applications. It is designed to be simple to use and manage, and uses technologies such as GlusterFS to provide dynamic storage. It is highly beneficial for those Network Applications that involve storing data consistently, even in situations of cluster failure.
- An **ingress controller** provides a few different options for serving your services on regular ports without conflicts. The ingress controller makes sure that you can connect to the right pod on your worker nodes while letting HAProxy route the request to one of your controllers on a regular port. The nginx-ingress uses a specific port on the cluster which allows access to all other services which have an ingress rule. The reason why an ingress controller is recommended is because Kubernetes services use ports in a very high port range which have to be bound in HAProxy. By using an ingress on a predefined port

only this port has to be defined in HAProxy, allowing easy routing and maintainability of the services.

- **Prometheus/Alertmanager/Grafana**. Prometheus collects metrics, Alertmanager manages alerts generated by Prometheus and sends them to configure notification channels, and Grafana provides a visual interface to create dashboards and visualize data from various sources, including Prometheus. In this way you can monitor the cluster and see the resources being consumed.

2.2.1.2 Open5GENESIS Platform

The Open5Genesis Framework has been deployed in the Málaga Platform, connected to the radio infrastructure and with access to the Kubernetes environment. The latest versions of the framework's components (Dispatcher and Analytics Module Rel_B, ELCM v3.6.3) have been used.

All the components, except for the InfluxDB database that is hosted in a separate machine accessible through wired network, have been deployed in a single machine. The specifications of the machine are as follow:

- CPU: Intel Core i7-7700 – 3.60GHz, RAM: 32 GB
- OS: Windows 10 Pro

The Experiment Life-Cycle Manager and auxiliary orchestrator (OpenTAP 9.13.1) have been installed directly in the host machine, while the Dispatcher and Analytics module are deployed as separate virtual machines under Oracle VM VirtualBox.

The following User Equipment is connected via USB to the computer, for testing of the radio conditions:

- Oneplus 11

2.2.2 Athens Certification Platform

2.2.2.1 Kubernetes Containerised Infrastructure

The Containerised infrastructure of the Athens platform Certification environment is in the Leonardo Lab at the premises of COSMOTE, who is the responsible partner for the Athens platform Certification process. The infrastructure is offered through a Kubernetes cluster that serves as the foundational infrastructure for conducting a wide array of experiments and hosting essential applications, as graphically depicted in Figure 4.

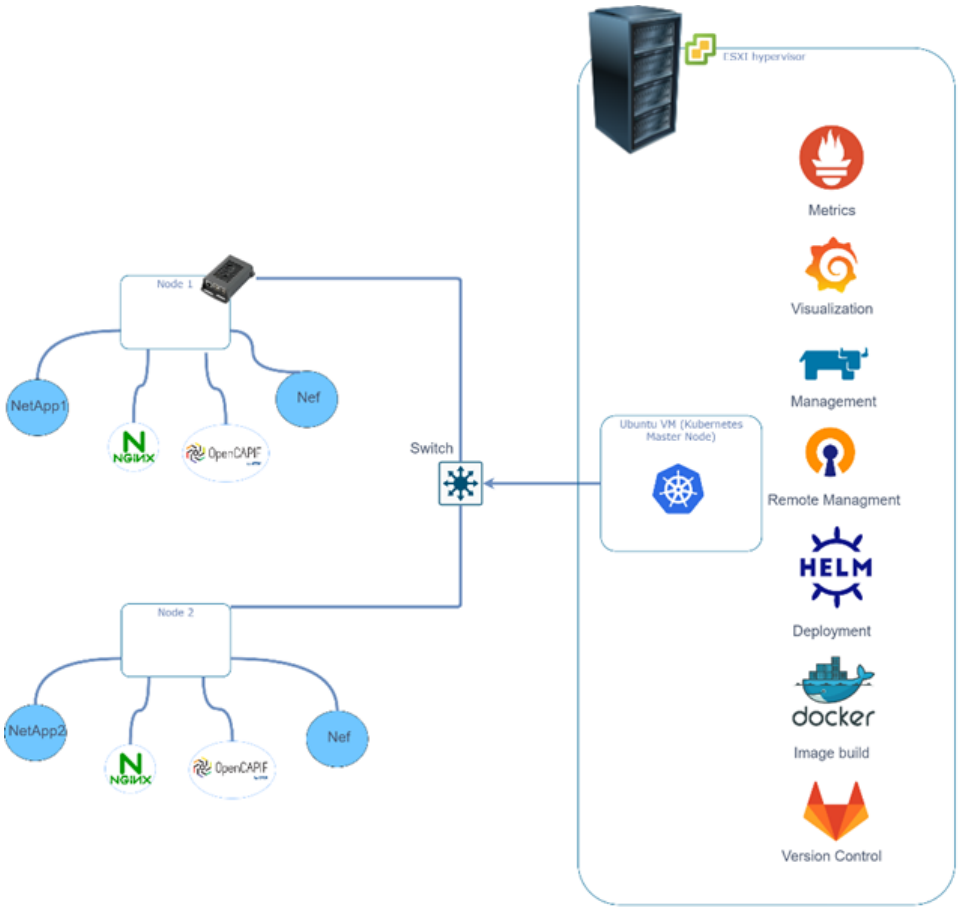


Figure 4: Athens Kubernetes Infrastructure hosted by COSMOTE

The cluster's architecture comprises a master node, operating on an ESXi hypervisor, boasting substantial resources: 16 vCPUs, 32 GB RAM, and a 100 GB hard disk. Supporting this core infrastructure are diverse worker nodes, including both virtual instances hosted across ESXi and OpenStack environments with average capacity of 8 vCPUs, 12 GB RAM and 100 GB Hard disk, as well as hardware nodes equipped with an Intel(R) Atom(TM) x5-Z8350 CPU @ 1.44GHz, 4 cores, 4GB RAM, and 50 GB disks. This heterogeneous setup allows for flexible resource allocation catering to various application requirements.

The cluster's scalability is a key feature, capable of horizontal scaling by seamlessly adding worker nodes to accommodate increased workloads. Furthermore, the flexibility extends to dynamically adjusting resources for virtual nodes based on evolving demands. Figure 4 depicts an indicative placement of the dynamic deployment of workloads, of both the components constituting the environment, such as CAPIF and NEF, and the under certification Network Application pods themselves. Similarly, to the Malaga platform, a number of services are utilised in this K8s deployment:

- The **Rancher Management** Platform was initially used for the first deployment, ensuring a robust and straightforward setup. A more sophisticated method, using **kubeadm** through a custom bash script was finally adopted, adding more versatility to the infrastructure deployment options.
- **Docker Containers** created with Docker offer a streamlined way to package, deploy, and run applications, ensuring consistency across different systems. Docker's containerization technology simplifies the deployment process, enhancing scalability and efficiency in diverse computing environments.
- **Cilium** serves as the primary networking solution, ensuring robust internal connectivity.
- **OpenEBS** as Kubernetes persistent storage solution.

- **Nginx** is used as a load balancer, intelligently routing requests to different nodes within the cluster, ensuring optimal resource utilization and high availability for web applications.
- **OpenVPN** enables secure remote management access, bolstering the cluster's operational accessibility and security.
- **Helm** serves as the Kubernetes package manager, streamlining application deployment and management by packaging applications into "charts," simplifying installation, upgrades, and maintenance for efficient operations within the cluster.
- **Prometheus** is employed for a comprehensive monitoring, to gather critical metrics essential for performance analysis and resource utilization. Visualizing these metrics is streamlined through **Grafana**, providing intuitive dashboards, and aiding in informed decision-making. Additionally, the integration of **LibreNMS** augments our monitoring suite, enabling efficient alerting mechanisms to promptly address any operational irregularities or issues within the cluster.
- **GitLab** is used for configuration files (yaml) and scripts management, and it provides a centralized repository for version control, enabling seamless tracking, collaboration, and versioning of configurations and deployment manifests. This ensures systematic changes, simplifies rollbacks, and fosters team collaboration in maintaining infrastructure configurations across the deployment pipeline.

In summary, the complete set of open-source components engaged in the Kubernetes cluster of the Certification environment of Athens, beyond the project’s developments reported in WP3 and WP4, include:

Component	Version	Reference
Docker	24.0.5	https://www.docker.com/
Rancher 2.7.9	2.7.9	https://www.rancher.com/
Cilium	1.13.4	https://cilium.io/
OpenEBS	3.6.0	https://openebs.io/
Nginx	1.24	https://nginx.org/
OpenVPN	2.5.5	https://openvpn.net/community-downloads/
Helm	3.12.1	https://helm.sh/
Prometheus	2.48	https://prometheus.io/
Grafana	10.1.5	https://grafana.com/
GitLab	16.5.2	https://about.gitlab.com/

Table 1: Athens K8 Components

2.2.2.2 Open5GENESIS Platform

The Open5Genesis Framework has been utilised in the Athens Platform in parallel with the Kubernetes environment. The framework includes the integration of two distinct virtual infrastructure managers (OpenStack and Red Hat OpenStack in NCSRD and COSMOTE, respectively) including the OSM and Slice Manager. This integration is pivotal for Network Service instantiation during network slice deployment, with these Virtual Infrastructure Managers (VIMs) taking on the crucial responsibility of instantiating the virtual machines, essential for executing the assessment of the platform. In the Coordination layer, encompassing five virtual machines, in which every vital Open5Genesis component, such as InfluxDB and Grafana, the Analytics module, ELCM, and the Dispatcher component, is deployed to facilitate the coordination of the measurements.

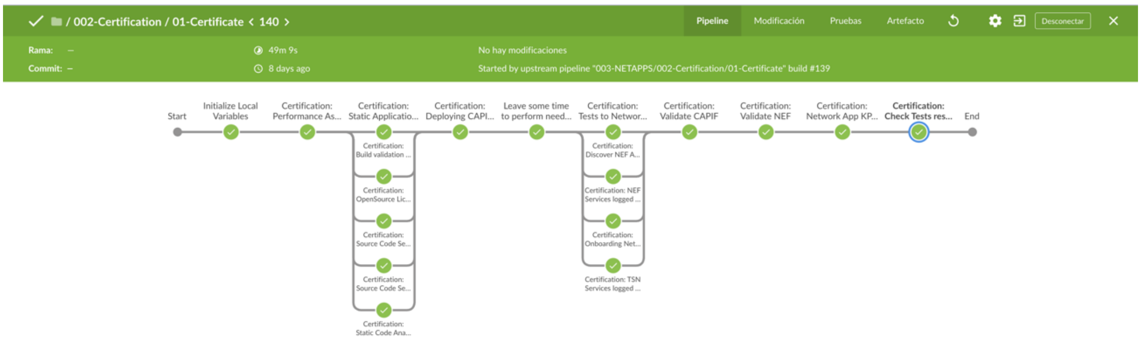
2.3 CERTIFICATION PIPELINE

Certification Pipeline was initially described in Deliverable 5.3 [1]. At that time, the description was based in the design requirements coming from WP2 and WP3. When implementing the Pipeline, some changes have been made to adjust the sequence of the different tests, and some tests having added/removed based in the value perceived.

Certification Pipeline (Figure 5) is the materialization of the Certification Process tests in an automated fashion. EVOLVED-5G uses Jenkins [5] as the automation tool. Jenkins is an industry automation tool standard and Jenkins usage has been already reported in Deliverables 3.2, 3.4 and 5.3.

Certification Process considers that the Network Applications are going to be used in a real 5G environment. Therefore, real 5G environments from Málaga and COSMOTE are used for Certification purposes. As those environments are not exclusive for EVOLVED-5G, the pipeline starts with an environment assessment to certify that the 5G Environment that will be used for testing performs according to expectations.

Once the environment assessment is certified, The Network Applications are analysed for security, secret leakages, vulnerabilities, etc. Only “secured” Network Applications will be deployed in Certification environment. These Network Application analysis is similar to the one performed during Validation phase. As only validated Network Applications can go through Certification, it is expected that Network Applications would be secure enough. This repetition, though, between validation and certification analysis is has been defined on purpose, as Validation and Certification entities would be different companies, and even the quality bar between Validation and Certification could be different.



✓ / 002-Certification / 01-Certificate < 140 >

Rama: –
Commit: –
49m 9s
8 days ago
No hay modificaciones
Started by upstream pipeline "003-NETAPPS/002-Certification/01-Certificate" build #139

Pipeline
Modificación
Pruebas
Artefacto
Desconectar
Start
Initialize Local Variables
Certification: Performance As...
Certification: Static Applicatio...
Certification: Build validation ...
Certification: OpenSource Lic...
Certification: Source Code Se...
Certification: Source Code Se...
Certification: Static Code Ana...
Certification: Deploying CAPI...
Leave some time to perform need...
Certification: Tests to Networ...
Certification: Discover NEF A...
Certification: NEF Services logged ...
Certification: Onboarding Net...
Certification: TSN Services logged ...
Certification: Validate CAPIF
Certification: Validate NEF
Certification: Network App KP...
Certification: Check Tests res...
End

Figure 5: Certification Pipeline visualization in Jenkins

The pipeline will deploy, then, the required 5G tools for completing the Certification tests defined in EVOLVED-5G, namely CAPIF, NEF and TSN. These components provide the APIs for Network Applications to interact with the 5G Network, which is basically the object of the EVOLVED-5G certification.

Network Applications will consume CAPIF, NEF and TSN APIs. CAPIF collects evidence of these interactions between Network Applications and the 5G APIs as NEF and TSN have integrated the Logging functionality from CAPIF as described in Deliverable 3.4 [7], section 3.1.

Jenkins collects the results of all these tests in JSON formats and then uses Jinja tool templates to generate first, markdown documents that are used, then, for generating the PDF report. This process has been described in Deliverable 3.4 [7].

2.3.1 CI/CD Environment

Telefónica´s CICD (Figure 6) environment has been mentioned and described in several deliverables already, such as 3.4 and 5.3. It is a collection of tools orchestrated by Jenkins though the Certification Pipeline described in this section.

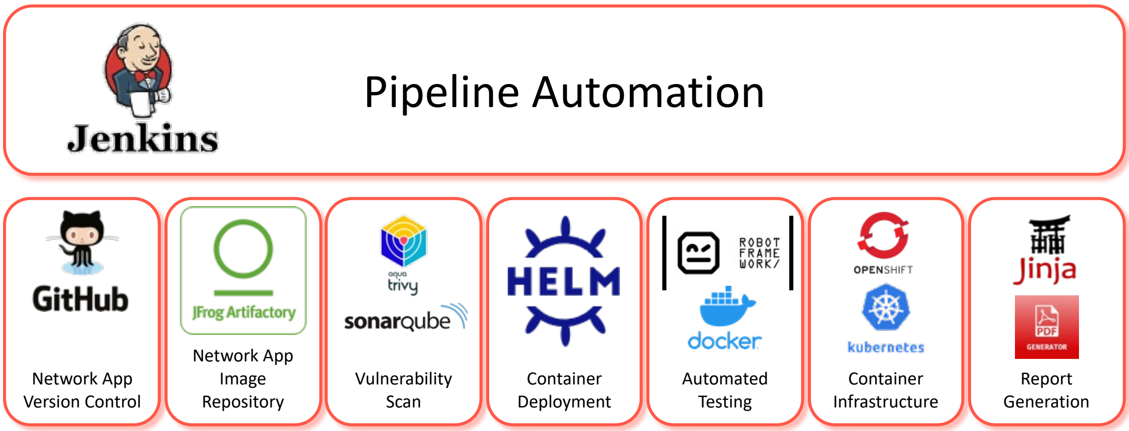
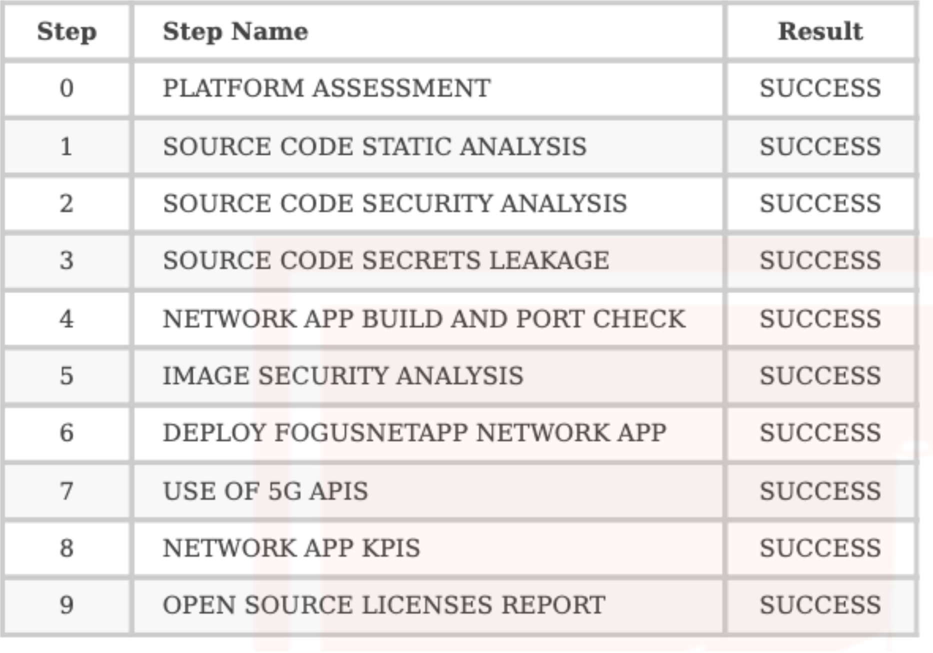


Figure 6: Tools in the CICD Environment

Github and Artifactory are the repositories for Network Applications source code and artifacts. While Github provides the Network Applications and 5G Tools repositories, Artifactory stores all images and temporary results produced during the execution of Validation and Certification processes.
Trivy and SonarQube tools are used during the Network Applications analysis and Helm Charts are used for deploying both, 5G Tools and Network Applications in Certification Environments. OpenShift and Kubernetes are the container platforms selected for deploying the Network Applications and Jinja tool and PDF Generator are used for producing the Certification reports.

2.3.2 Certification steps

The Certification Pipeline has been structure in TEN steps (Figure 7) that are reported in the Certification report Summary page as follows:



Step	Step Name	Result
0	PLATFORM ASSESSMENT	SUCCESS
1	SOURCE CODE STATIC ANALYSIS	SUCCESS
2	SOURCE CODE SECURITY ANALYSIS	SUCCESS
3	SOURCE CODE SECRETS LEAKAGE	SUCCESS
4	NETWORK APP BUILD AND PORT CHECK	SUCCESS
5	IMAGE SECURITY ANALYSIS	SUCCESS
6	DEPLOY FOGUSNETAPP NETWORK APP	SUCCESS
7	USE OF 5G APIS	SUCCESS
8	NETWORK APP KPIS	SUCCESS
9	OPEN SOURCE LICENSES REPORT	SUCCESS

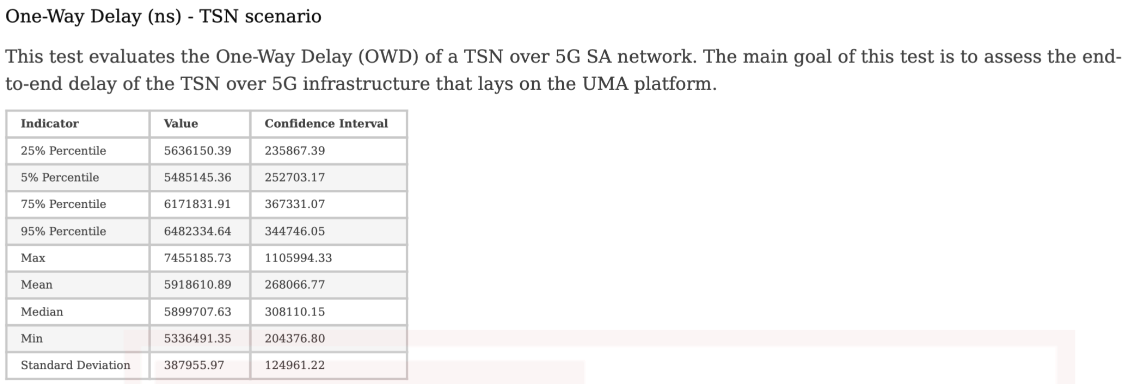
Figure 7: Summary report of Certification Steps

Step 0 – Platform Assessment

This is a step that only takes place in the Certification phase. Certifications, to have value, need to take place in environments that have the conditions required to make the certification tests required to complete the certification. In other words, the environment must not introduce entropy in the testing process.

For this purpose, a platform assessment is executed before starting the certification tests. This platform assessment provides information for the Network Application developers about the conditions in which their Network Applications have been tested in relation to the 5G infrastructure used during the tests.

For applications that use TSN capabilities, TSN measurements reported in D5.4 have been included in the report as a reference. Two sets of KPIs are included for one way delay and Jitter over the 5G Network.

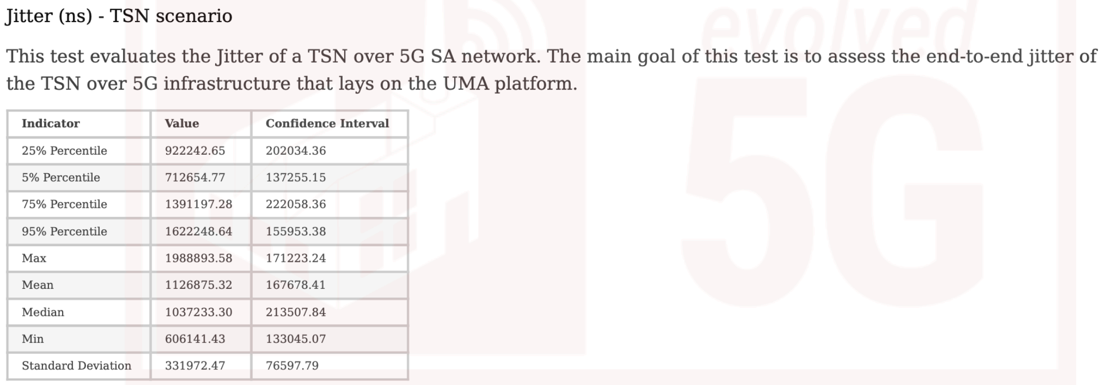


One-Way Delay (ns) - TSN scenario

This test evaluates the One-Way Delay (OWD) of a TSN over 5G SA network. The main goal of this test is to assess the end-to-end delay of the TSN over 5G infrastructure that lays on the UMA platform.

Indicator	Value	Confidence Interval
25% Percentile	5636150.39	235867.39
5% Percentile	5485145.36	252703.17
75% Percentile	6171831.91	367331.07
95% Percentile	6482334.64	344746.05
Max	7455185.73	1105994.33
Mean	5918610.89	268066.77
Median	5899707.63	308110.15
Min	5336491.35	204376.80
Standard Deviation	387955.97	124961.22

Figure 8: TSN One-Way Delay KPIs



Jitter (ns) - TSN scenario

This test evaluates the Jitter of a TSN over 5G SA network. The main goal of this test is to assess the end-to-end jitter of the TSN over 5G infrastructure that lays on the UMA platform.

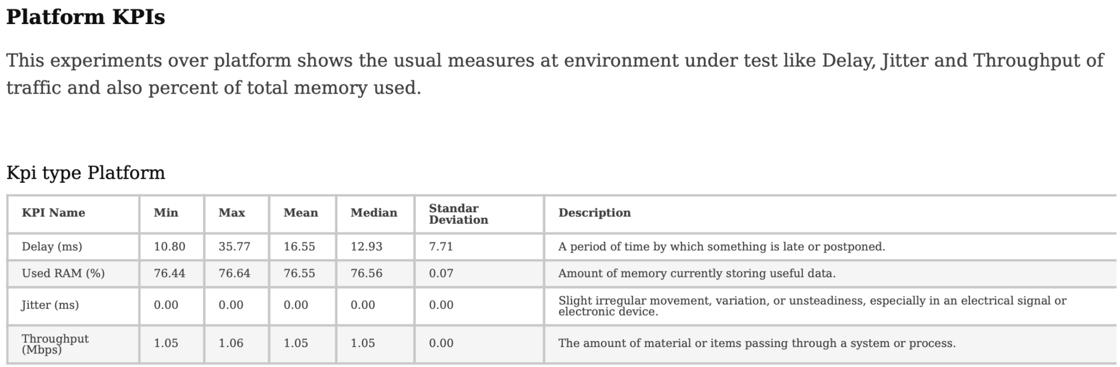
Indicator	Value	Confidence Interval
25% Percentile	922242.65	202034.36
5% Percentile	712654.77	137255.15
75% Percentile	1391197.28	222058.36
95% Percentile	1622248.64	155953.38
Max	1988893.58	171223.24
Mean	1126875.32	167678.41
Median	1037233.30	213507.84
Min	606141.43	133045.07
Standard Deviation	331972.47	76597.79

Figure 9: TSN Jitter KPIs

After TSN KPIs reporting, Jenkins invokes an assessment experiment defined in the Open5Genesis testing framework that is part of each 5G Platform. This experiment calculates several KPIs by using tools such as iPerf, repeating several cycles. The results of these test cycles are consolidated, and Minimum, Maximum, Mean, Median and Standard deviation are calculated from the datasets produced.

The KPIs calculated are:

- Delay (ms): delay in ms calculated from a UE connected to the 5G Network to a server running iPerf.
- UsedRAM (%): amount of memory consumed during the tests.
- Jitter (ms): variation of the delay measured in ms.
- Thoughput (Mbps): amount of Mbps transmitted between the UE connected to the 5G Network and the server running iPerf.



Platform KPIs

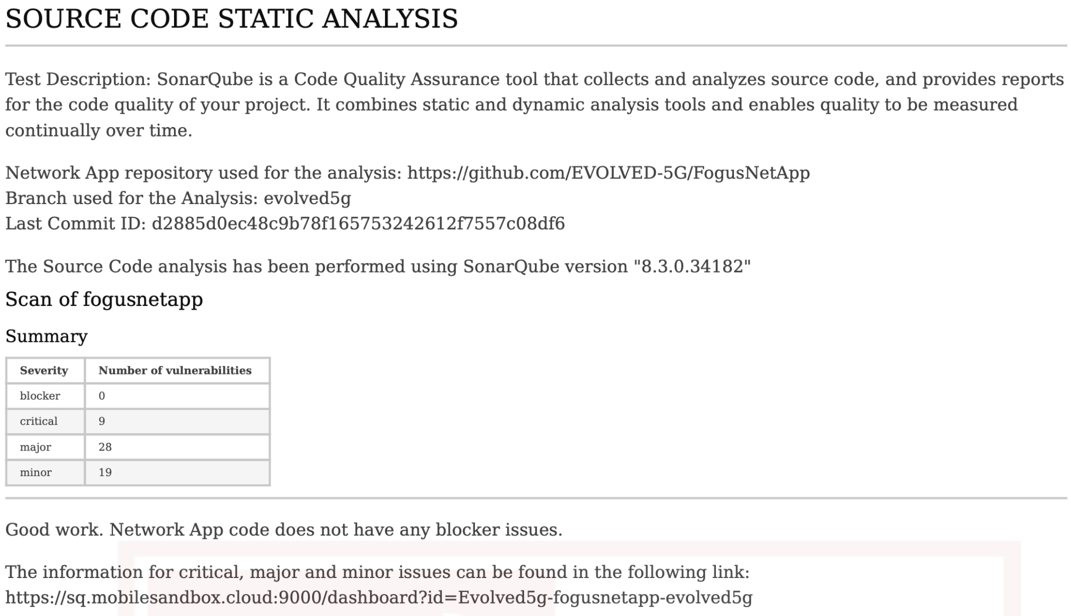
This experiments over platform shows the usual measures at environment under test like Delay, Jitter and Throughput of traffic and also percent of total memory used.

Kpi type Platform

KPI Name	Min	Max	Mean	Median	Standar Deviation	Description
Delay (ms)	10.80	35.77	16.55	12.93	7.71	A period of time by which something is late or postponed.
Used RAM (%)	76.44	76.64	76.55	76.56	0.07	Amount of memory currently storing useful data.
Jitter (ms)	0.00	0.00	0.00	0.00	0.00	Slight irregular movement, variation, or unsteadiness, especially in an electrical signal or electronic device.
Throughput (Mbps)	1.05	1.06	1.05	1.05	0.00	The amount of material or items passing through a system or process.

Figure 10: 5G Platform KPI assessment

Step 1 - Source Code Static Analysis: This step will obtain some quality code metrics in order to ensure good quality in the developed code. For this operation, Jenkins will use SonarQube version "8.3.0.34182". The results of this step are displayed in the Static Code Analysis Results page in the Certification Report (Figure 11).



SOURCE CODE STATIC ANALYSIS

Test Description: SonarQube is a Code Quality Assurance tool that collects and analyzes source code, and provides reports for the code quality of your project. It combines static and dynamic analysis tools and enables quality to be measured continually over time.

Network App repository used for the analysis: https://github.com/EVOLVED-5G/FogusNetApp
Branch used for the Analysis: evolved5g
Last Commit ID: d2885d0ec48c9b78f165753242612f7557c08df6

The Source Code analysis has been performed using SonarQube version "8.3.0.34182"

Scan of fogusnetapp

Summary

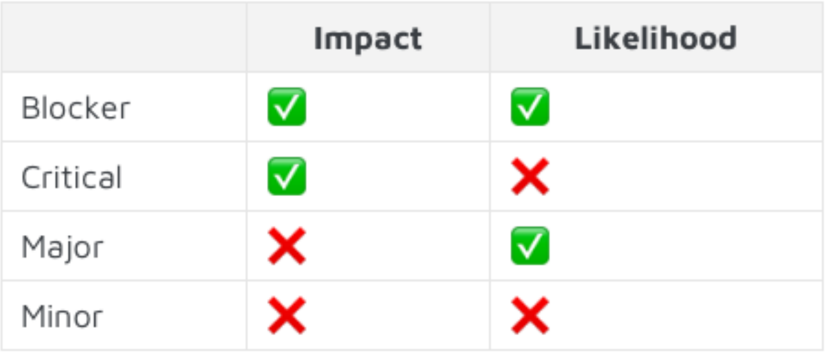
Severity	Number of vulnerabilities
blocker	0
critical	9
major	28
minor	19

Good work. Network App code does not have any blocker issues.

The information for critical, major and minor issues can be found in the following link:
https://sq.mobilesandbox.cloud:9000/dashboard?id=Evolved5g-fogusnetapp-evolved5g

Figure 11: 5G Source Code Static Analysis Report Page

SonarQube reports different levels of severity for the vulnerabilities detected (Figure 12). Severity levels are Blocker, Critical, Major and Minor. These four levels are defined based in two parameters: Impact and Likelihood (translated as Probability to happen).

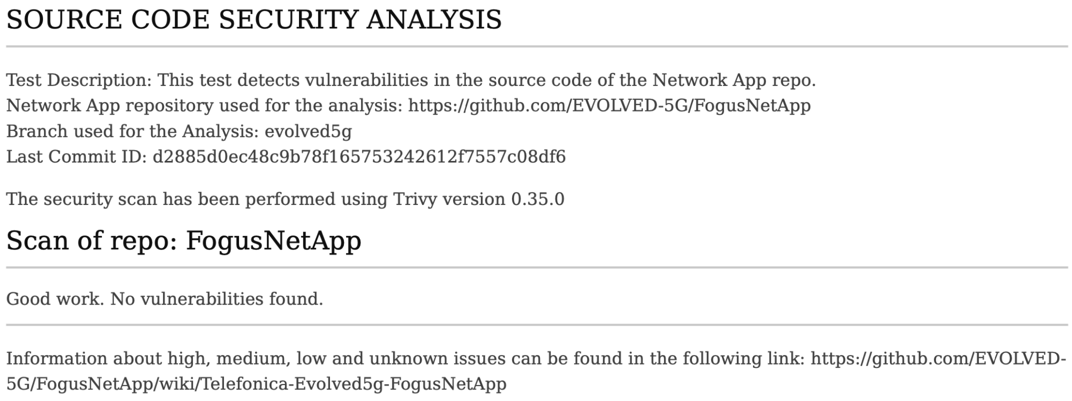


	Impact	Likelihood
Blocker	✓	✓
Critical	✓	✗
Major	✗	✓
Minor	✗	✗

Figure 12: SonarQube Severity assessment criteria

The analysis reports how many occurrences has detected for each level. It is up to the Certification Authority to define the Quality bar demanded for this step to be reported as Success or Fail. In EVOLVED-5G we have set the Quality bar for Blocker issues only. If there are any Blocker issues, the step is reported as Fail and the developer should remove the issues to complete the Certification.

Step 2 – Source Code Security Analysis: This step will scan the Source Code to find vulnerabilities. For this operation, Jenkins will use Trivy Compliance external tool [8] version 0.35.



SOURCE CODE SECURITY ANALYSIS

Test Description: This test detects vulnerabilities in the source code of the Network App repo.
Network App repository used for the analysis: https://github.com/EVOLVED-5G/FogusNetApp
Branch used for the Analysis: evolved5g
Last Commit ID: d2885d0ec48c9b78f165753242612f7557c08df6

The security scan has been performed using Trivy version 0.35.0

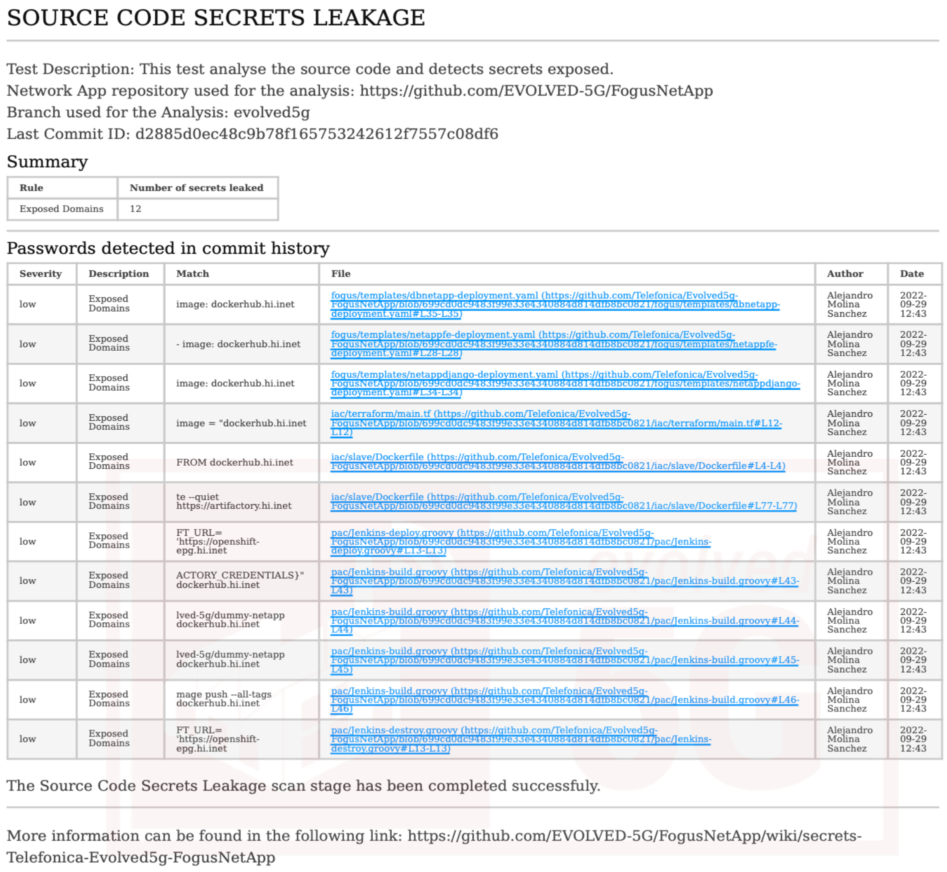
Scan of repo: FogusNetApp

Good work. No vulnerabilities found.

Information about high, medium, low and unknown issues can be found in the following link: https://github.com/EVOLVED-5G/FogusNetApp/wiki/Telefonica-Evolved5g-FogusNetApp

Figure 13: Source Code Security Analysis Report Page

Step 3 - Source Code Secrets Leakage: This step will check if there has been any secret leakage in the git history of the Network App repository. Trivy will be used for this step as well.



SOURCE CODE SECRETS LEAKAGE

Test Description: This test analyse the source code and detects secrets exposed.
Network App repository used for the analysis: https://github.com/EVOLVED-5G/FogusNetApp
Branch used for the Analysis: evolved5g
Last Commit ID: d2885d0ec48c9b78f165753242612f7557c08df6

Summary

Rule	Number of secrets leaked
Exposed Domains	12

Passwords detected in commit history

Severity	Description	Match	File	Author	Date
low	Exposed Domains	image: dockerhub.hi.inet	fogus/templates/dbnetapp-deployment.yaml (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/fogus/templates/dbnetapp-deployment.yaml#L35-L35)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	- image: dockerhub.hi.inet	fogus/templates/netappfe-deployment.yaml (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/fogus/templates/netappfe-deployment.yaml#L28-L28)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	image: dockerhub.hi.inet	fogus/templates/netappdjango-deployment.yaml (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/fogus/templates/netappdjango-deployment.yaml#L34-L34)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	image = "dockerhub.hi.inet	iac/terraform/main.tf (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/iac/terraform/main.tf#L12-L12)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	FROM dockerhub.hi.inet	iac/slave/Dockerfile (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/iac/slave/Dockerfile#L4-L4)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	te --quiet https://artifactory.hi.inet	iac/slave/Dockerfile (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/iac/slave/Dockerfile#L77-L77)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	FT_URL= 'https://openshift-epg.hi.inet	pac/Jenkins-deploy.groovy (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/pac/Jenkins-deploy.groovy#L13-L13)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	ACTORY_CREDENTIALS}" dockerhub.hi.inet	pac/Jenkins-build.groovy (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/pac/Jenkins-build.groovy#L43-L43)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	lved-5g/dummy-netapp dockerhub.hi.inet	pac/Jenkins-build.groovy (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/pac/Jenkins-build.groovy#L44-L44)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	lved-5g/dummy-netapp dockerhub.hi.inet	pac/Jenkins-build.groovy (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/pac/Jenkins-build.groovy#L45-L45)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	mage push --all-tags dockerhub.hi.inet	pac/Jenkins-build.groovy (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/pac/Jenkins-build.groovy#L46-L46)	Alejandro Molina Sanchez	2022-09-29 12:43
low	Exposed Domains	FT_URL= 'https://openshift-epg.hi.inet	pac/Jenkins-destroy.groovy (https://github.com/Telefonica/Evolved5g-FogusNetApp/blob/699cd0dc9483f99e33e4340884d814dfb8bc0821/pac/Jenkins-destroy.groovy#L13-L13)	Alejandro Molina Sanchez	2022-09-29 12:43

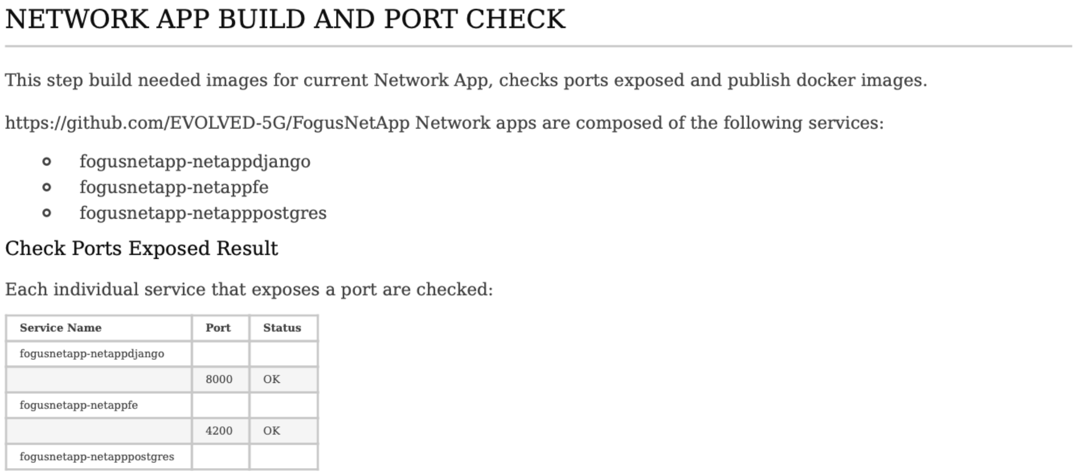
The Source Code Secrets Leakage scan stage has been completed successfuly.

More information can be found in the following link: https://github.com/EVOLVED-5G/FogusNetApp/wiki/secrets-Telefonica-Evolved5g-FogusNetApp

Figure 14: Source Code Secrets Analysis Report Page

This step detects sensible information inside the Network Application that can be used to attack the Network Application. Sensible information can be from usernames and passwords stores in the code to domains exposed that can lead to DDoS attacks.
For each item detected, a Severity level is assigned and a link to artifact where the issue has been detected is provided.

Step 4 – Network App Build and Port Check: Once the source code has been analysed, Network App container images needs to be built and connectivity specified for the Network Apps checked. This is performed in this step, container images are built and uploaded to both EVOLVED-5G Repository (Artifactory) and to the Docker registry in AWS. URLs for generated containers are included in the report. Additionally, open ports declared by the Network Apps are verified to validate that connectivity with the Network App will be working properly once deployed.



NETWORK APP BUILD AND PORT CHECK

This step build needed images for current Network App, checks ports exposed and publish docker images.

https://github.com/EVOLVED-5G/FogusNetApp Network apps are composed of the following services:

- fogusnetapp-netappdjango
- fogusnetapp-netappfe
- fogusnetapp-netapppostgres

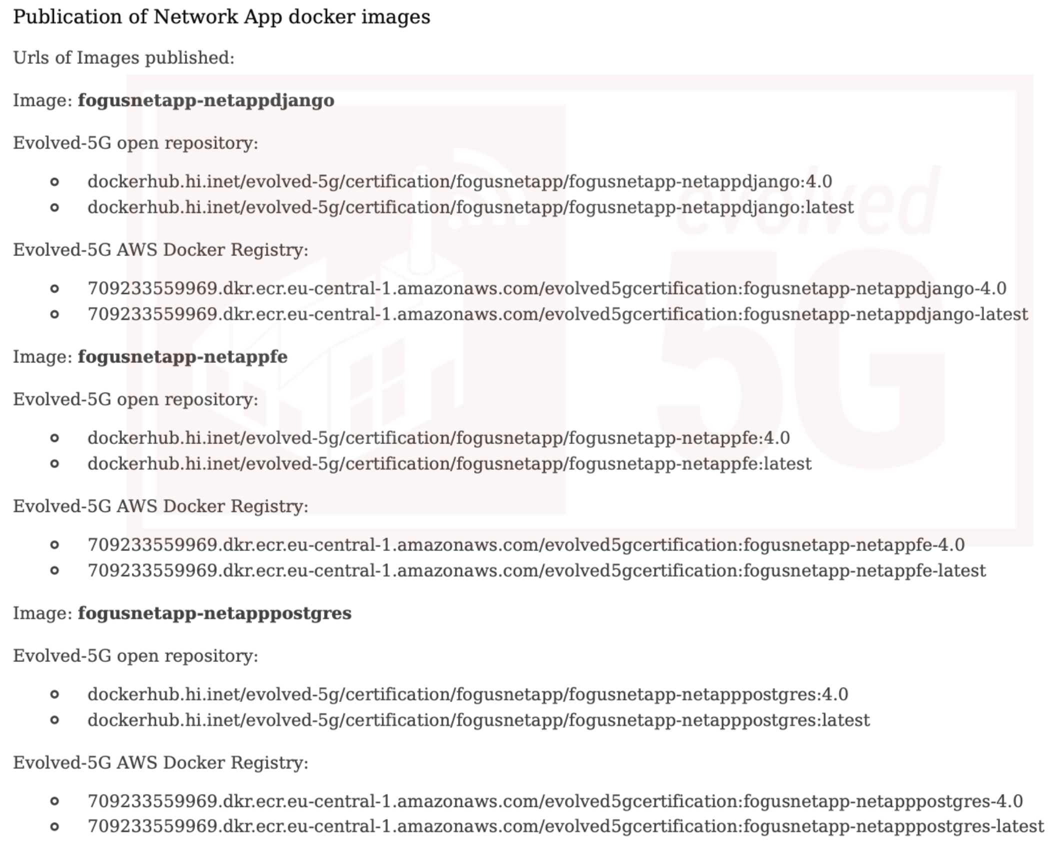
Check Ports Exposed Result

Each individual service that exposes a port are checked:

Service Name	Port	Status
fogusnetapp-netappdjango		
	8000	OK
fogusnetapp-netappfe		
	4200	OK
fogusnetapp-netapppostgres		

Figure 15: Network Application Build and Port Check

After checking the ports opened for communicating with the Network Application, the images generated are published in two Docker repositories, one private and one public. The private repository is inside CICD environment for internal consumption. The public repository is needed to deploy the Network Applications images in Málaga and Athens environments, as images will be pulled from the Kubernetes environments themselves.



Publication of Network App docker images

Urls of Images published:

Image: **fogusnetapp-netappdjango**

Evolved-5G open repository:

- dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappdjango:4.0
- dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappdjango:latest

Evolved-5G AWS Docker Registry:

- 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappdjango-4.0
- 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappdjango-latest

Image: **fogusnetapp-netappfe**

Evolved-5G open repository:

- dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappfe:4.0
- dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappfe:latest

Evolved-5G AWS Docker Registry:

- 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappfe-4.0
- 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappfe-latest

Image: **fogusnetapp-netapppostgres**

Evolved-5G open repository:

- dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netapppostgres:4.0
- dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netapppostgres:latest

Evolved-5G AWS Docker Registry:

- 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netapppostgres-4.0
- 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netapppostgres-latest

Figure 16: Network Applications Container image information

Step 5 - Image Security Analysis: In this step, the generated binary images of the Network App are analysed for security vulnerabilities. This step is mandatory as the container images contain not only Network Application code but the Operative System and libraries that have not been analysed in previous steps. Vulnerabilities can be detected in Libraries used by the Network Application that jeopardize the security of the container image in a production environment. One report page is generated per container image of the Network Application.

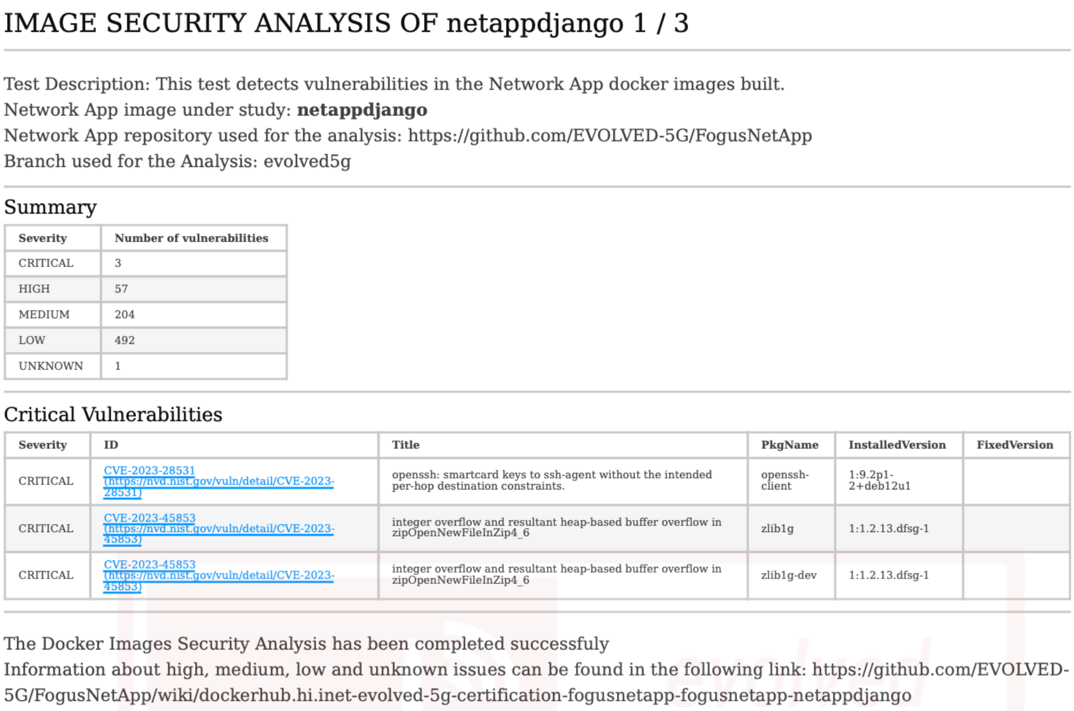


IMAGE SECURITY ANALYSIS OF netappdjango 1 / 3

Test Description: This test detects vulnerabilities in the Network App docker images built.
Network App image under study: **netappdjango**
Network App repository used for the analysis: https://github.com/EVOLVED-5G/FogusNetApp
Branch used for the Analysis: evolved5g

Summary

Severity	Number of vulnerabilities
CRITICAL	3
HIGH	57
MEDIUM	204
LOW	492
UNKNOWN	1

Critical Vulnerabilities

Severity	ID	Title	PkgName	InstalledVersion	FixedVersion
CRITICAL	CVE-2023-28531 (https://nvd.nist.gov/vuln/detail/CVE-2023-28531)	openssh: smartcard keys to ssh-agent without the intended per-hop destination constraints.	openssh-client	1:9.2p1-2+deb12u1	
CRITICAL	CVE-2023-45853 (https://nvd.nist.gov/vuln/detail/CVE-2023-45853)	integer overflow and resultant heap-based buffer overflow in zipOpenNewFileInZip4_6	zlib1g	1:1.2.13.dfsg-1	
CRITICAL	CVE-2023-45853 (https://nvd.nist.gov/vuln/detail/CVE-2023-45853)	integer overflow and resultant heap-based buffer overflow in zipOpenNewFileInZip4_6	zlib1g-dev	1:1.2.13.dfsg-1	

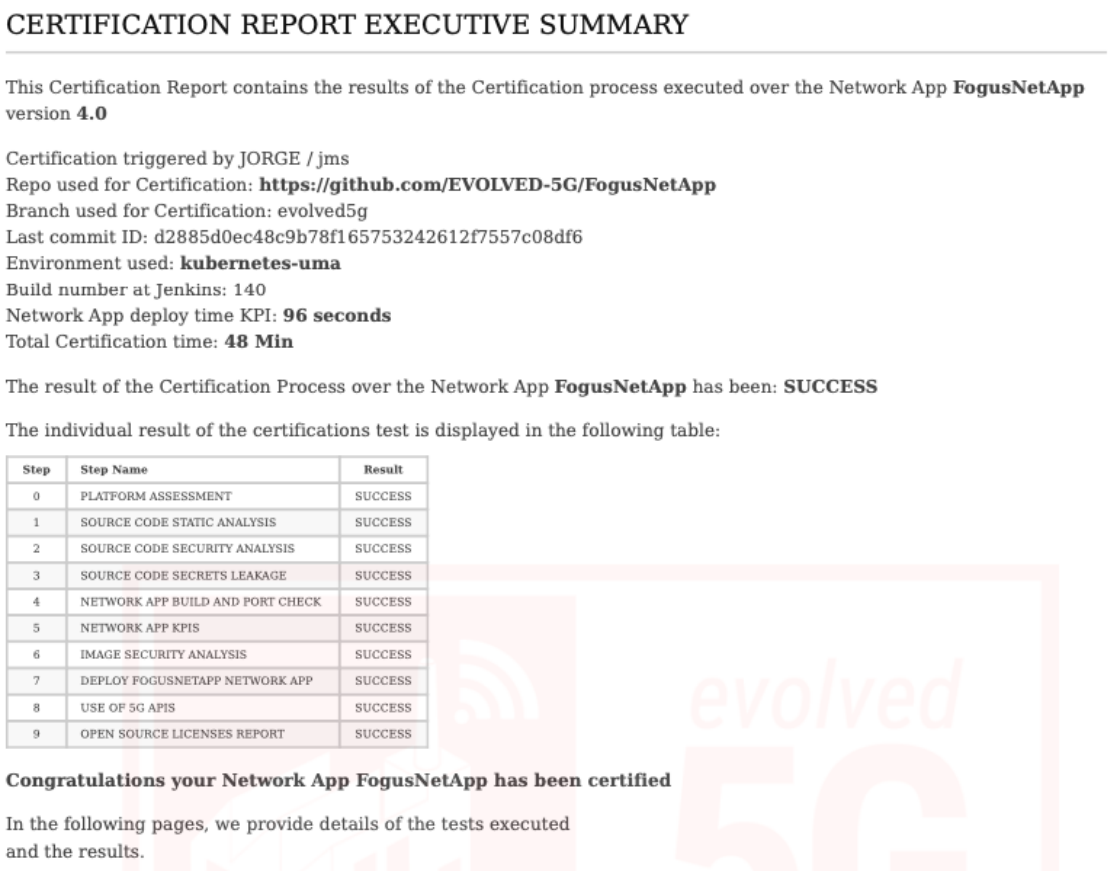
The Docker Images Security Analysis has been completed successfuly
Information about high, medium, low and unknown issues can be found in the following link: https://github.com/EVOLVED-5G/FogusNetApp/wiki/dockerhub.hi.inet-evolved-5g-certification-fogusnetapp-fogusnetapp-netappdjango

Figure 17: Container Image Security Analysis Report Page

Issues reported have also a Severity level assigned by the tool. In this case, the levels are from CRITICAL to LOW with an UNKNOW category for issues that the tool is unable to categorize. The Quality bar is, again, subject to the Certification authority. In EVOLVED-5G we have set the Quality bar at CRITICAL issues in the Network App code itself or CRITICAL issues in Libraries that have an available Fix.

CRITICAL issues in Libraries that do not have a Fix yet are passed, as we cannot demand the developers to fix something that do not have a fix available yet.

Step 6 – Deploy the Network Application: Once the Network Application images have been proved secured, the Network Application is deployed in Kubernetes infrastructure to execute the next steps. The deployment time KPI is reported in the Summary page as displayed in the following picture:



CERTIFICATION REPORT EXECUTIVE SUMMARY

This Certification Report contains the results of the Certification process executed over the Network App **FogusNetApp** version **4.0**

Certification triggered by JORGE / jms
Repo used for Certification: **https://github.com/EVOLVED-5G/FogusNetApp**
Branch used for Certification: evolved5g
Last commit ID: d2885d0ec48c9b78f165753242612f7557c08df6
Environment used: **kubernetes-uma**
Build number at Jenkins: 140
Network App deploy time KPI: **96 seconds**
Total Certification time: **48 Min**

The result of the Certification Process over the Network App **FogusNetApp** has been: **SUCCESS**

The individual result of the certifications test is displayed in the following table:

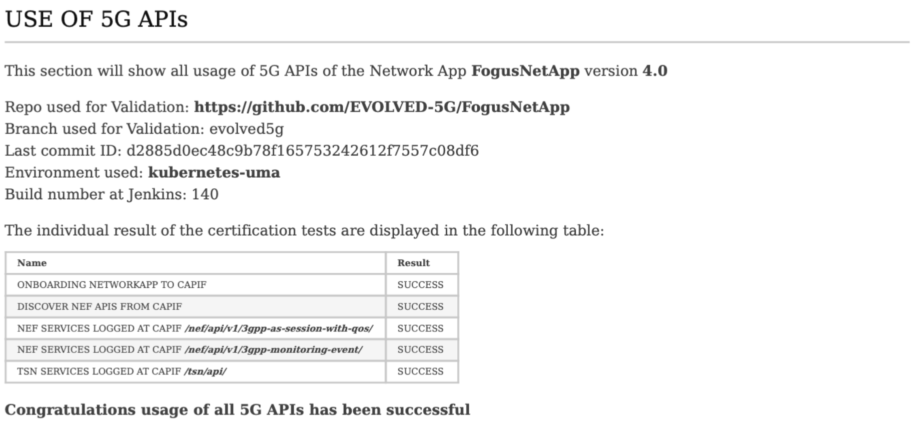
Step	Step Name	Result
0	PLATFORM ASSESSMENT	SUCCESS
1	SOURCE CODE STATIC ANALYSIS	SUCCESS
2	SOURCE CODE SECURITY ANALYSIS	SUCCESS
3	SOURCE CODE SECRETS LEAKAGE	SUCCESS
4	NETWORK APP BUILD AND PORT CHECK	SUCCESS
5	NETWORK APP KPIS	SUCCESS
6	IMAGE SECURITY ANALYSIS	SUCCESS
7	DEPLOY FOGUSNETAPP NETWORK APP	SUCCESS
8	USE OF 5G APIS	SUCCESS
9	OPEN SOURCE LICENSES REPORT	SUCCESS

Congratulations your Network App FogusNetApp has been certified

In the following pages, we provide details of the tests executed and the results.

Figure 18: Summary of the Certification Report containing the Deployment Time KPI.

Step 7 – Usage of 5G APIs. This step is cornerstone of EVOLVED-5G project. It reports the usage of 5G Platform APIs integrated in the Network Application. 5G APIs are typically NEF APIs and TSN APIs that, in EVOLVED-5G, are exposed using CAPIF. On one hand, CAPIF integration is tested to do the Onboarding of the Network Application in CAPIF but also Discovery of the NEF and TSN APIs using CAPIF. Once NEF and TSN APIs have been discovered by the Network Applications, NEF and TSN Endpoints are consumed by the Network Applications. Both NEF Emulator and TSN Application Function report API usage to CAPIF using the Logging API capability in CAPIF Core Function. This method for reporting API usage is convenient as it is Extendible to any other API that in the future would be added to EVOLVED-5G test suite.



USE OF 5G APIs

This section will show all usage of 5G APIs of the Network App **FogusNetApp** version **4.0**

Repo used for Validation: **https://github.com/EVOLVED-5G/FogusNetApp**
Branch used for Validation: evolved5g
Last commit ID: d2885d0ec48c9b78f165753242612f7557c08df6
Environment used: **kubernetes-uma**
Build number at Jenkins: 140

The individual result of the certification tests are displayed in the following table:

Name	Result
ONBOARDING NETWORKAPP TO CAPIF	SUCCESS
DISCOVER NEF APIS FROM CAPIF	SUCCESS
NEF SERVICES LOGGED AT CAPIF ***/nef/api/v1/3gpp-as-session-with-qos/***	SUCCESS
NEF SERVICES LOGGED AT CAPIF ***/nef/api/v1/3gpp-monitoring-event/***	SUCCESS
TSN SERVICES LOGGED AT CAPIF ***/tsn/api/***	SUCCESS

Congratulations usage of all 5G APIs has been successful

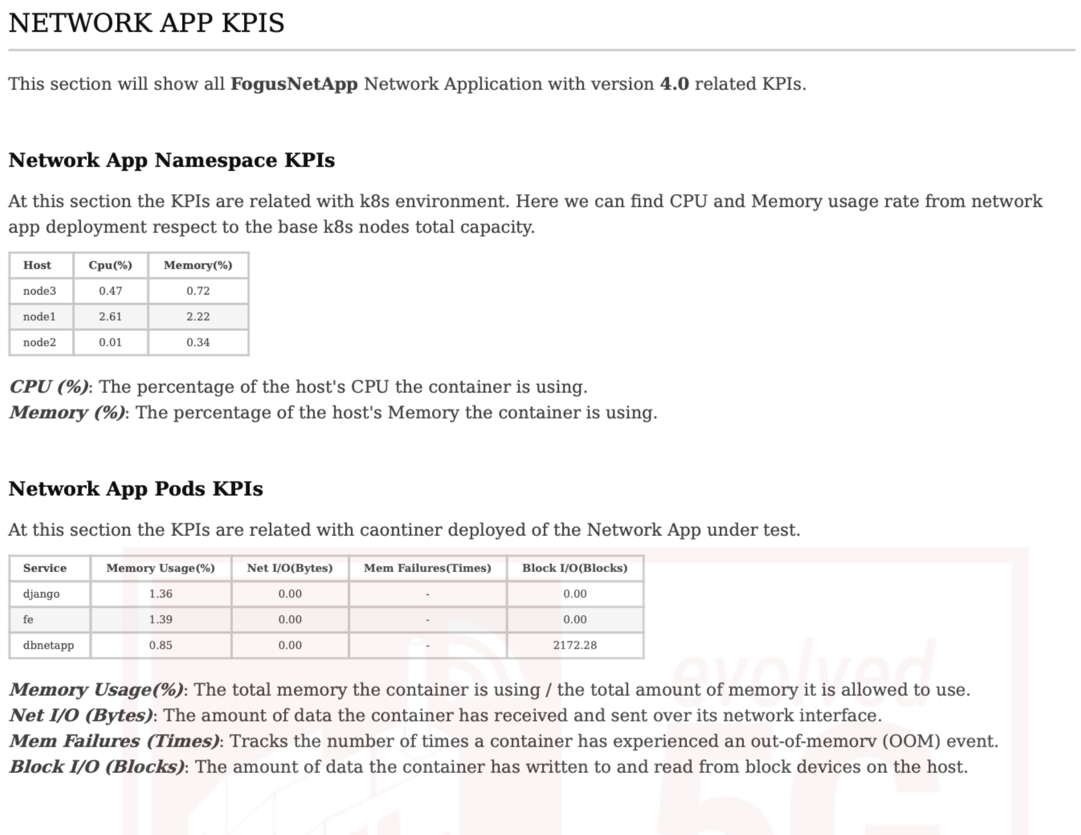
Figure 19: 5G API Usage Analysis Report Page

Step 8 – Network Application KPIs

This is a new step introduced in the Certification Pipeline that provides information about the resources used by the Network Application after it has been deployed in Kubernetes. On one hand, information about the Namespace where the Network Application has been deployed is provided. All the hosts running the Namespace are listed and for each host, the percentage of CPU utilization and the amount of memory consumed is listed. This KPIs provide information about the hosts utilization by the Network Application globally.

Then, an analysis per container is presented. Containers can be deployed in different workers. For each Pod, the KPIs displayed are:

- Memory usage (%): The total memory the container is using / the total amount of memory it is allowed to use.
- Net I/O (Bytes): The amount of data the container has received and sent over its network interface.
- Mem Failures (Times): Tracks the number of times a container has experienced an out-of-memory (OOM) event.
- Block I/O (Blocks): The amount of data the container has written to and read from block devices on the host.



NETWORK APP KPIS

This section will show all **FogusNetApp** Network Application with version **4.0** related KPIs.

Network App Namespace KPIs

At this section the KPIs are related with k8s environment. Here we can find CPU and Memory usage rate from network app deployment respect to the base k8s nodes total capacity.

Host	Cpu(%)	Memory(%)
node3	0.47	0.72
node1	2.61	2.22
node2	0.01	0.34

CPU (%): The percentage of the host's CPU the container is using.
Memory (%): The percentage of the host's Memory the container is using.

Network App Pods KPIs

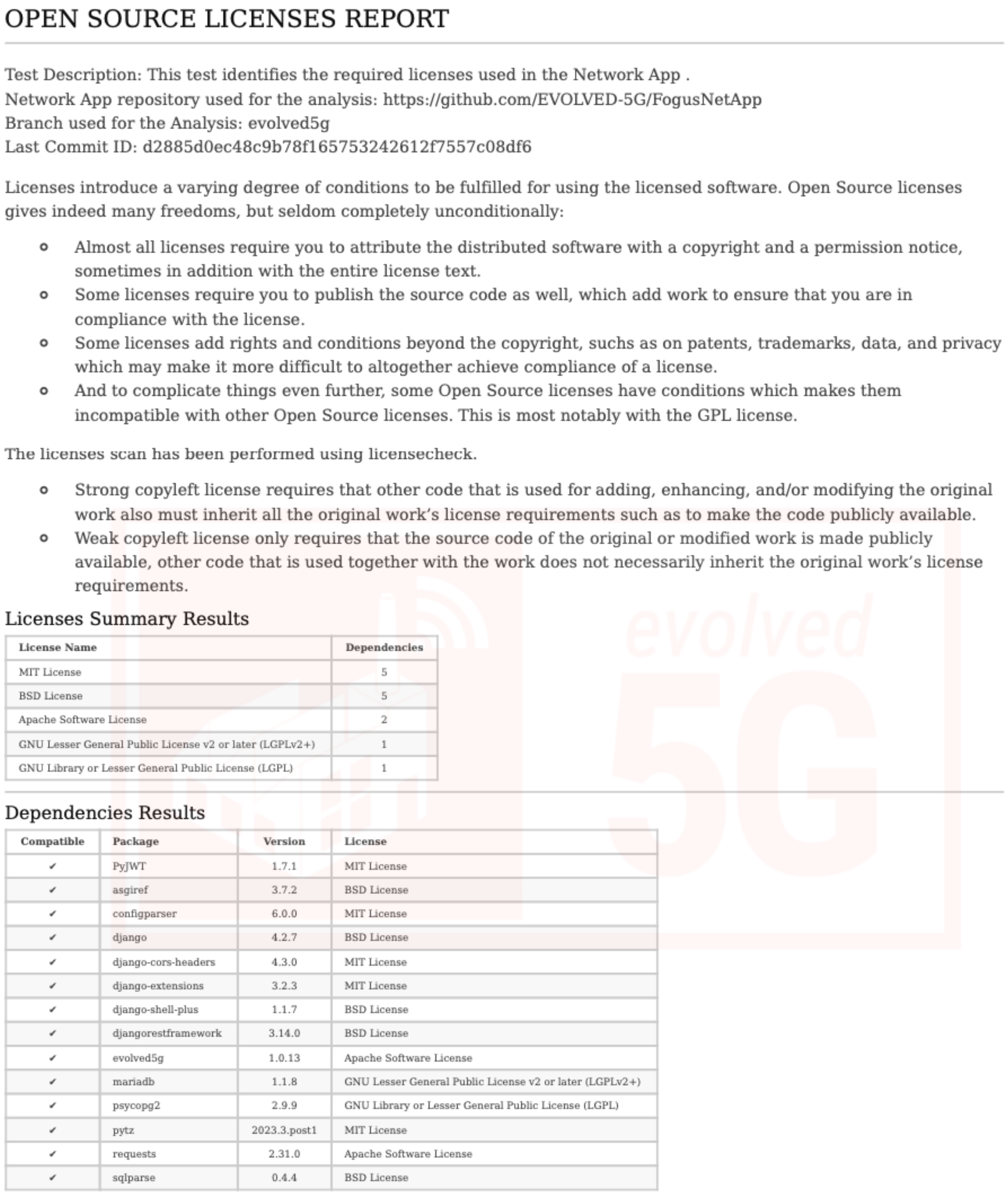
At this section the KPIs are related with caontiner deployed of the Network App under test.

Service	Memory Usage(%)	Net I/O(Bytes)	Mem Failures(Times)	Block I/O(Blocks)
django	1.36	0.00	-	0.00
fe	1.39	0.00	-	0.00
dbnetapp	0.85	0.00	-	2172.28

Memory Usage(%): The total memory the container is using / the total amount of memory it is allowed to use.
Net I/O (Bytes): The amount of data the container has received and sent over its network interface.
Mem Failures (Times): Tracks the number of times a container has experienced an out-of-memorv (OOM) event.
Block I/O (Blocks): The amount of data the container has written to and read from block devices on the host.

Figure 20: Network Applications KPIs

Step 9 – Open-Source License Report: Final step is to collect information about open-source Licenses used by the Network Application. Though initially Debriked was selected as the tool to report this information, a License change in Debriked during the project force us to change to use Licensecheck [2] tool instead, that offers similar information. This last-minute change reveals the flexibility and adaptability of the EVOLVED-5G Certification environment and the integration facilities to adapt new tools. The information provided is informative both for Developers and for potential users of the Network Applications.



OPEN SOURCE LICENSES REPORT

Test Description: This test identifies the required licenses used in the Network App .
Network App repository used for the analysis: https://github.com/EVOLVED-5G/FogusNetApp
Branch used for the Analysis: evolved5g
Last Commit ID: d2885d0ec48c9b78f165753242612f7557c08df6

Licenses introduce a varying degree of conditions to be fulfilled for using the licensed software. Open Source licenses gives indeed many freedoms, but seldom completely unconditionally:

- Almost all licenses require you to attribute the distributed software with a copyright and a permission notice, sometimes in addition with the entire license text.
- Some licenses require you to publish the source code as well, which add work to ensure that you are in compliance with the license.
- Some licenses add rights and conditions beyond the copyright, suchs as on patents, trademarks, data, and privacy which may make it more difficult to altogether achieve compliance of a license.
- And to complicate things even further, some Open Source licenses have conditions which makes them incompatible with other Open Source licenses. This is most notably with the GPL license.

The licenses scan has been performed using licensecheck.

- Strong copyleft license requires that other code that is used for adding, enhancing, and/or modifying the original work also must inherit all the original work's license requirements such as to make the code publicly available.
- Weak copyleft license only requires that the source code of the original or modified work is made publicly available, other code that is used together with the work does not necessarily inherit the original work's license requirements.

Licenses Summary Results

License Name	Dependencies
MIT License	5
BSD License	5
Apache Software License	2
GNU Lesser General Public License v2 or later (LGPLv2+)	1
GNU Library or Lesser General Public License (LGPL)	1

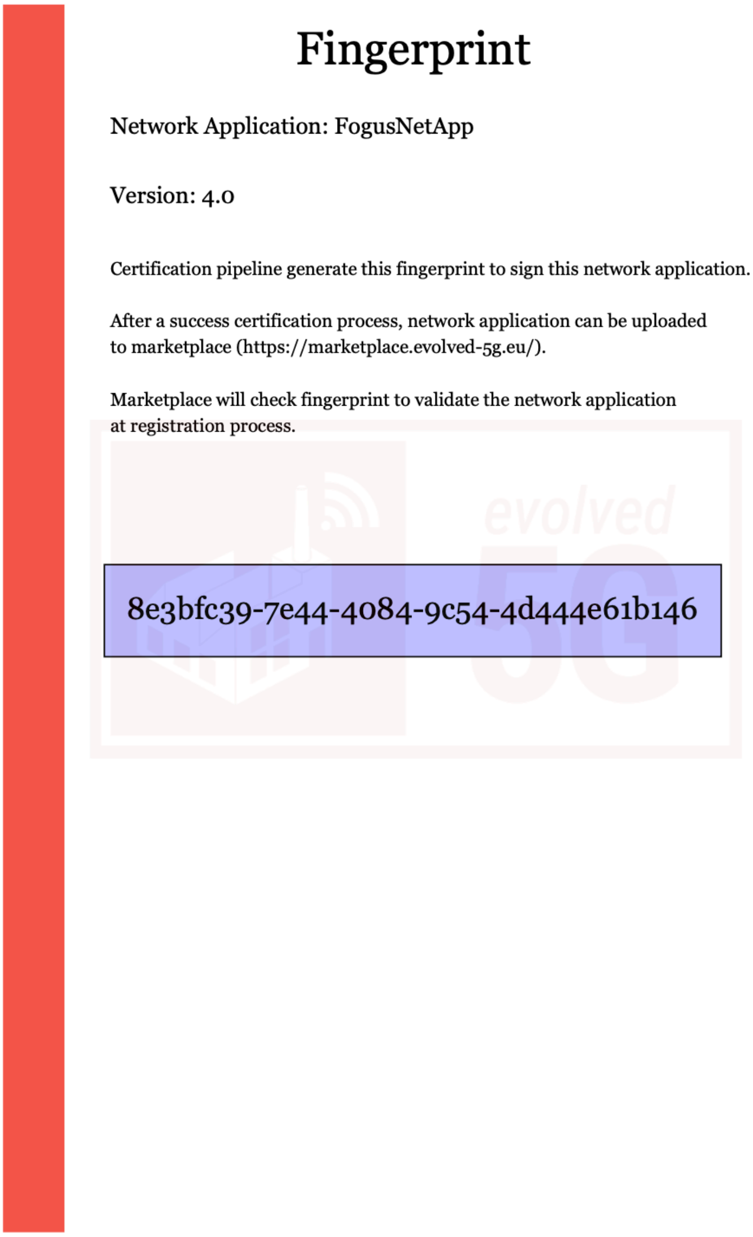
Dependencies Results

Compatible	Package	Version	License
✓	PyJWT	1.7.1	MIT License
✓	asgiref	3.7.2	BSD License
✓	configparser	6.0.0	MIT License
✓	django	4.2.7	BSD License
✓	django-cors-headers	4.3.0	MIT License
✓	django-extensions	3.2.3	MIT License
✓	django-shell-plus	1.1.7	BSD License
✓	djangorestframework	3.14.0	BSD License
✓	evolved5g	1.0.13	Apache Software License
✓	mariadb	1.1.8	GNU Lesser General Public License v2 or later (LGPLv2+)
✓	psycopg2	2.9.9	GNU Library or Lesser General Public License (LGPL)
✓	pytz	2023.3.post1	MIT License
✓	requests	2.31.0	Apache Software License
✓	sqlparse	0.4.4	BSD License

Figure 21: Open-Source Licenses Analysis Report Page

Fingerprint: The Certification Report includes a page with a Fingerprint ID. This Fingerprint ID is critical for the publication of the Network Application in the Marketplace. This Fingerprint ID relates the Software images of the Network Application in the repository with the Certification issued as part of the Certification process. Marketplace will ask for this ID when publishing a Network Application and will verify that this Fingerprint ID is stored in the Artifactory folder containing the Certified images of the Network Application.

Every Certification execution generates a new Fingerprint ID. Developers must use the latest Fingerprint received if they have performed several certifications of the same version.



Fingerprint

Network Application: FogusNetApp

Version: 4.0

Certification pipeline generate this fingerprint to sign this network application.

After a success certification process, network application can be uploaded to marketplace (https://marketplace.evolved-5g.eu/).

Marketplace will check fingerprint to validate the network application at registration process.

8e3bfc39-7e44-4084-9c54-4d444e61b146

Figure 22: Fingerprint for the Publication of the Network Application in the Marketplace

2.3.3 Open-repository Onboarding of Certified Network Applications

Upon successful certification, the Network Application is onboarded to the Certification area of the Open Repository implemented in Artifactory. This step is mandatory to be able to publish a Network Application in the EVOLVED-5G Marketplace.

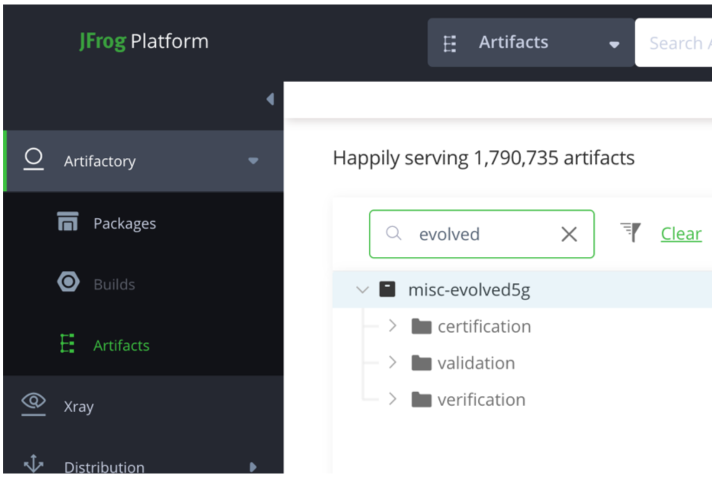


Figure 23: Open Repository Certification area

The Certification area in the Open Repository contains one folder per Network Application that has executed the Certification Pipeline.

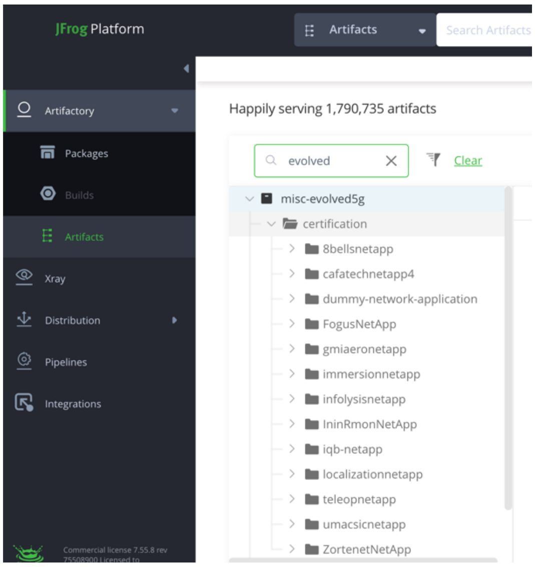
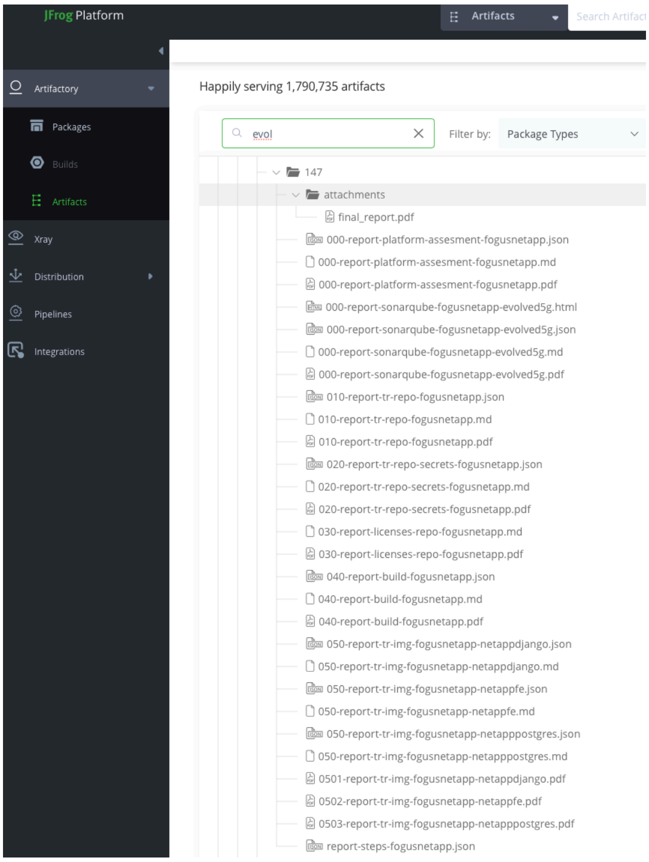


Figure 24: Open Repository Certification showing Network Application folders

For each Certification, Jenkins creates a folder within the Application Network folder, with the execution ID that will contain all temporary results of the Certification tests executed by the pipeline. These results are stored as JSON objects in the folder. For each JSON file, we use Jinja tool to generate a Markdown document with the information in the JSON file but in a readable by human’s format.



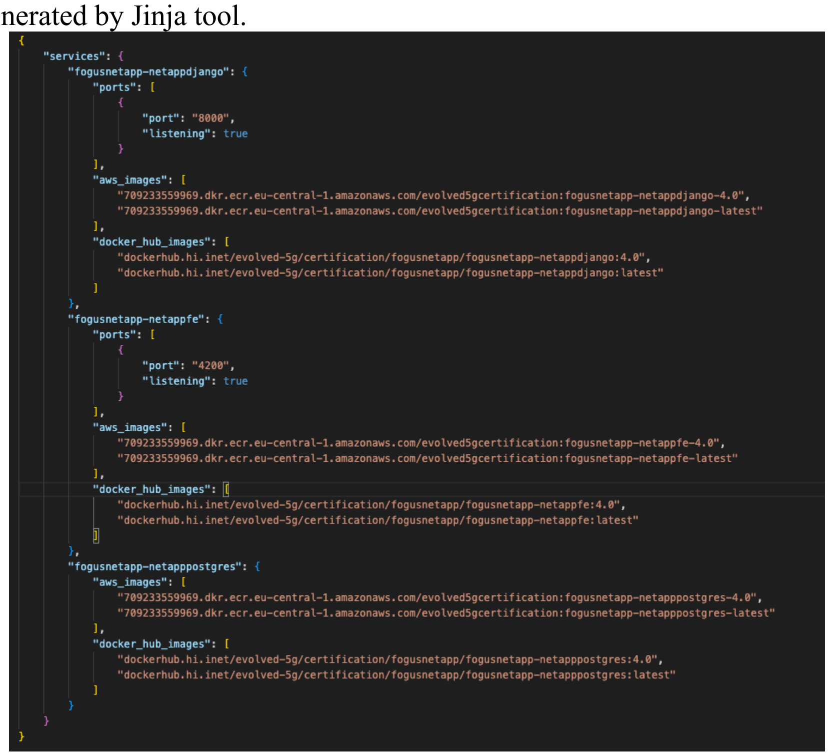
Happily serving 1,790,735 artifacts

evol

```
147
  attachments
    final_report.pdf
  000-report-platform-assesment-fogusnetapp.json
  000-report-platform-assesment-fogusnetapp.md
  000-report-platform-assesment-fogusnetapp.pdf
  000-report-sonarqube-fogusnetapp-evolved5g.html
  000-report-sonarqube-fogusnetapp-evolved5g.json
  000-report-sonarqube-fogusnetapp-evolved5g.md
  000-report-sonarqube-fogusnetapp-evolved5g.pdf
  010-report-tr-repo-fogusnetapp.json
  010-report-tr-repo-fogusnetapp.md
  010-report-tr-repo-fogusnetapp.pdf
  020-report-tr-repo-secrets-fogusnetapp.json
  020-report-tr-repo-secrets-fogusnetapp.md
  020-report-tr-repo-secrets-fogusnetapp.pdf
  030-report-licenses-repo-fogusnetapp.md
  030-report-licenses-repo-fogusnetapp.pdf
  040-report-build-fogusnetapp.json
  040-report-build-fogusnetapp.md
  040-report-build-fogusnetapp.pdf
  050-report-tr-img-fogusnetapp-netappdjango.json
  050-report-tr-img-fogusnetapp-netappdjango.md
  050-report-tr-img-fogusnetapp-netappfe.json
  050-report-tr-img-fogusnetapp-netappfe.md
  050-report-tr-img-fogusnetapp-netapppostgres.json
  050-report-tr-img-fogusnetapp-netapppostgres.md
  0501-report-tr-img-fogusnetapp-netappdjango.pdf
  0502-report-tr-img-fogusnetapp-netappfe.pdf
  0503-report-tr-img-fogusnetapp-netapppostgres.pdf
  report-steps-fogusnetapp.json
```

Figure 25: Open Repository Certification showing temporary JSON results and markdowns

As an illustrative example, we show in Figure 26 the JSON file and corresponding Markdown document generated by Jinja tool.



```
{
    "services": {
        "fogusnetapp-netappdjango": {
            "ports": [
                {
                    "port": "8000",
                    "listening": true
                }
            ],
            "aws_images": [
                "709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappdjango-4.0",
                "709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappdjango-latest"
            ],
            "docker_hub_images": [
                "dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappdjango:4.0",
                "dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappdjango:latest"
            ]
        },
        "fogusnetapp-netappfe": {
            "ports": [
                {
                    "port": "4200",
                    "listening": true
                }
            ],
            "aws_images": [
                "709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappfe-4.0",
                "709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappfe-latest"
            ],
            "docker_hub_images": [
                "dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappfe:4.0",
                "dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappfe:latest"
            ]
        },
        "fogusnetapp-netapppostgres": {
            "aws_images": [
                "709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netapppostgres-4.0",
                "709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netapppostgres-latest"
            ],
            "docker_hub_images": [
                "dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netapppostgres:4.0",
                "dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netapppostgres:latest"
            ]
        }
    }
}
```

Figure 26: JSON file generated by Jenkins step Network App build and Ports Check

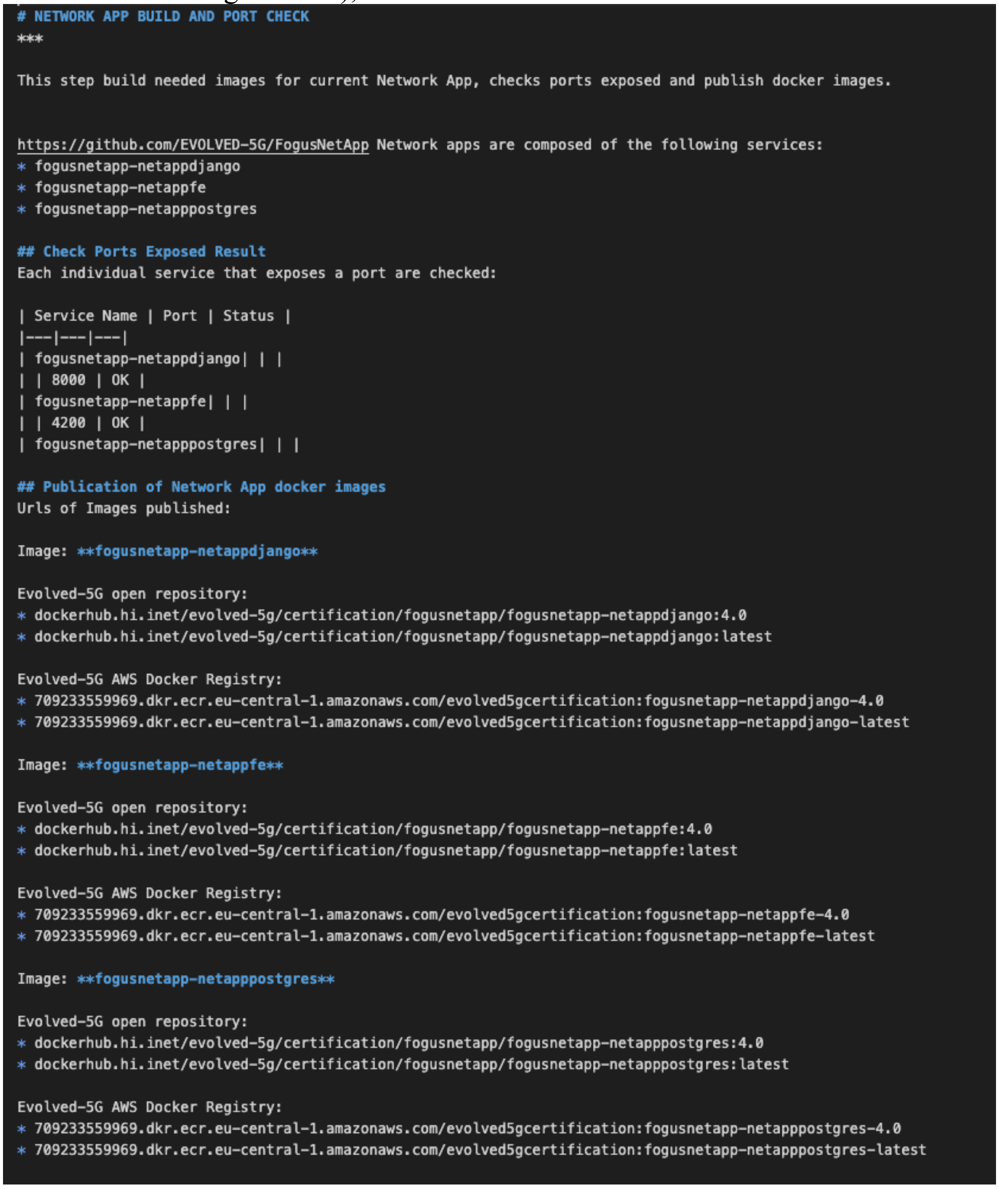
This sample file contains three services (containers):

- fogusnetapp-netappdjango

- fogusnetapp-netappfe
- fogusnetapp-netapppostgres

For each container, the open ports are listed. The first container has the port 8080 opened (listening), the second container has the port 4200 opened and the third container has no ports opened.
Also, for each container, the AWS ECR URLs and the docker hub URLs of the images are provided.

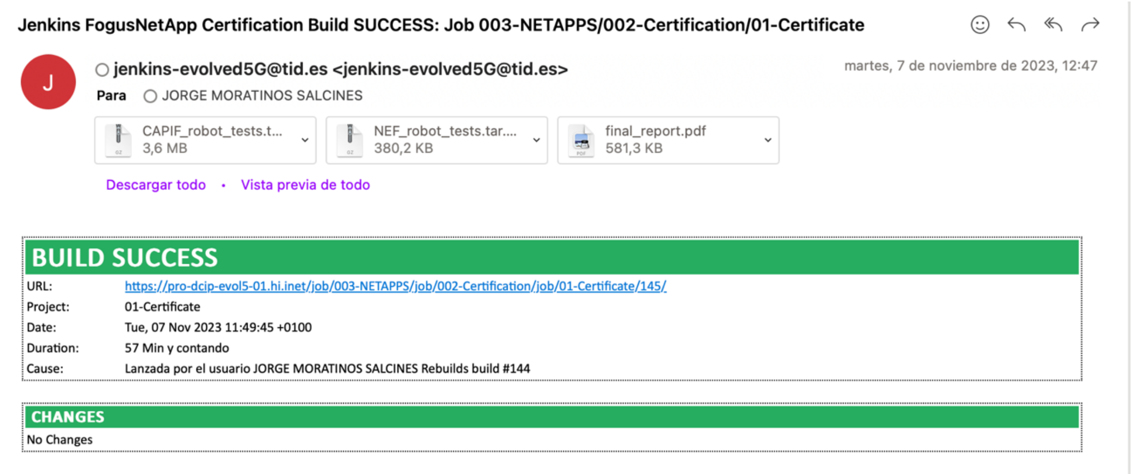
This file is used by Jinja tool to generate the corresponding Markdown document. The Markdown document contains the same information as the JSON file, ordered by categories (first opened ports, then container images URLs), but in a human readable format.



```
# NETWORK APP BUILD AND PORT CHECK
***

This step build needed images for current Network App, checks ports exposed and publish docker images.


https://github.com/EVOLVED-5G/FogusNetApp Network apps are composed of the following services:
* fogusnetapp-netappdjango
* fogusnetapp-netappfe
* fogusnetapp-netapppostgres

## Check Ports Exposed Result
Each individual service that exposes a port are checked:

| Service Name | Port | Status |
|---|---|---|
| fogusnetapp-netappdjango| | |
| | 8000 | OK |
| fogusnetapp-netappfe| | |
| | 4200 | OK |
| fogusnetapp-netapppostgres| | |

## Publication of Network App docker images
Urls of Images published:

Image: **fogusnetapp-netappdjango**

Evolved-5G open repository:
* dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappdjango:4.0
* dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappdjango:latest

Evolved-5G AWS Docker Registry:
* 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappdjango-4.0
* 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappdjango-latest

Image: **fogusnetapp-netappfe**

Evolved-5G open repository:
* dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappfe:4.0
* dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netappfe:latest

Evolved-5G AWS Docker Registry:
* 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappfe-4.0
* 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netappfe-latest

Image: **fogusnetapp-netapppostgres**

Evolved-5G open repository:
* dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netapppostgres:4.0
* dockerhub.hi.inet/evolved-5g/certification/fogusnetapp/fogusnetapp-netapppostgres:latest

Evolved-5G AWS Docker Registry:
* 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netapppostgres-4.0
* 709233559969.dkr.ecr.eu-central-1.amazonaws.com/evolved5gcertification:fogusnetapp-netapppostgres-latest
```

Figure 27: Markdown document generated using Jinja tool from JSON file

This text plain file is easier to convert to PDF format using PDF Generator. With all the partial results that generate the JSON files, Jenkins translate these JSON files into Markdown documents that are used to generate the Certification report. This report is stored also in the Certification area of the Open Repository, in a folder with name “attachments”. This PDF file is the one that will be attached in the email to the Network Application developer communicating the Certification results.



Jenkins FogusNetApp Certification Build SUCCESS: Job 003-NETAPPS/002-Certification/01-Certificate

jenkins-evolved5G@tid.es <jenkins-evolved5G@tid.es>

martes, 7 de noviembre de 2023, 12:47

Para ○ JORGE MORATINOS SALCINES

CAPIF_robot_tests.t... 3,6 MB

NEF_robot_tests.tar.... 380,2 KB

final_report.pdf 581,3 KB

Descargar todo • Vista previa de todo

BUILD SUCCESS

URL:	https://pro-dcip-evol5-01.hi.inet/job/003-NETAPPS/job/002-Certification/job/01-Certificate/145/
Project:	01-Certificate
Date:	Tue, 07 Nov 2023 11:49:45 +0100
Duration:	57 Min y contando
Cause:	Lanzada por el usuario JORGE MORATINOS SALCINES Rebuilds build #144

CHANGES

No Changes

Figure 28: Email to Network Application Developer communicating the certification results

The certified images are stored in a folder with the version of the Network Application as the folder name. This folder contains, as well, the fingerpring.json file. This fingerprint file contains the Fingerprint ID described in section 2.3.2.

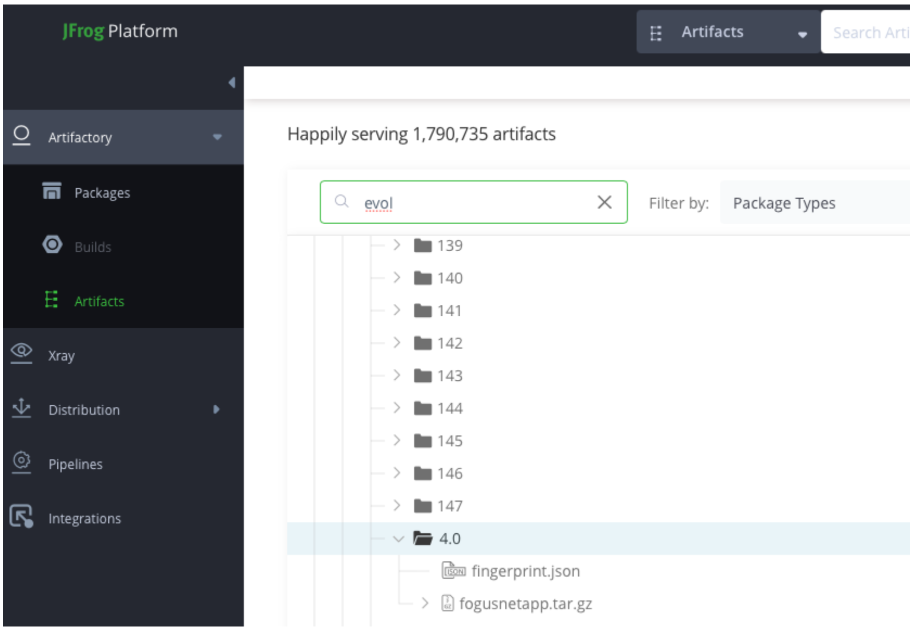


Figure 29: Open Repository Certification: Fingerprint file and Network Application images

Fingerprint.json file contains only two information elements:

- certificationid: is a unique ID generated for this specific certification execution and is the one displayed in the Fingerprint section of the Certification Report, and the ID needed to publish the Network Application in the Marketplace.
- Version: is the version of the Network Application that has been certified.

```
{
  "certificationid": "8e3bfc39-7e44-4084-9c54-4d444e61b146",
  "Version": "4.0"
}
```

Figure 30: Fingerprint.json content

2.3.4 Certification Report

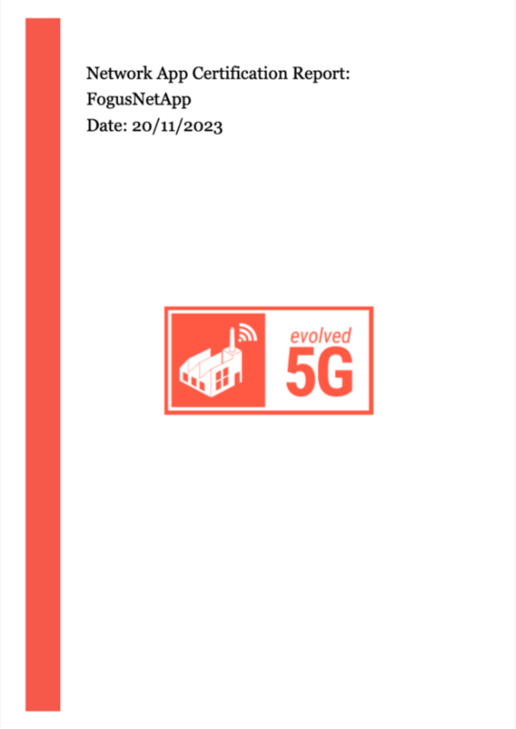
This section outlines how the Certification Report is generated and what is the structure of the Certification Report file.

2.3.4.1 Report Generation

The Certification Tests has been described already in 2.3.2. For each step in pipeline, a capture of the results included in the certification report has been outlined. The certification report is built from the JSON files generated in each step in the pipeline. JSON files are translated first into Markdown documents using Jinja templates, and then a PDF is generated for each Markdown document. All pages are integrated into a single PDF file that we call Certification Report. Next section describes the Certification Report structure with the pages containing the results of the Certification steps.

2.3.4.2 Report summary structure

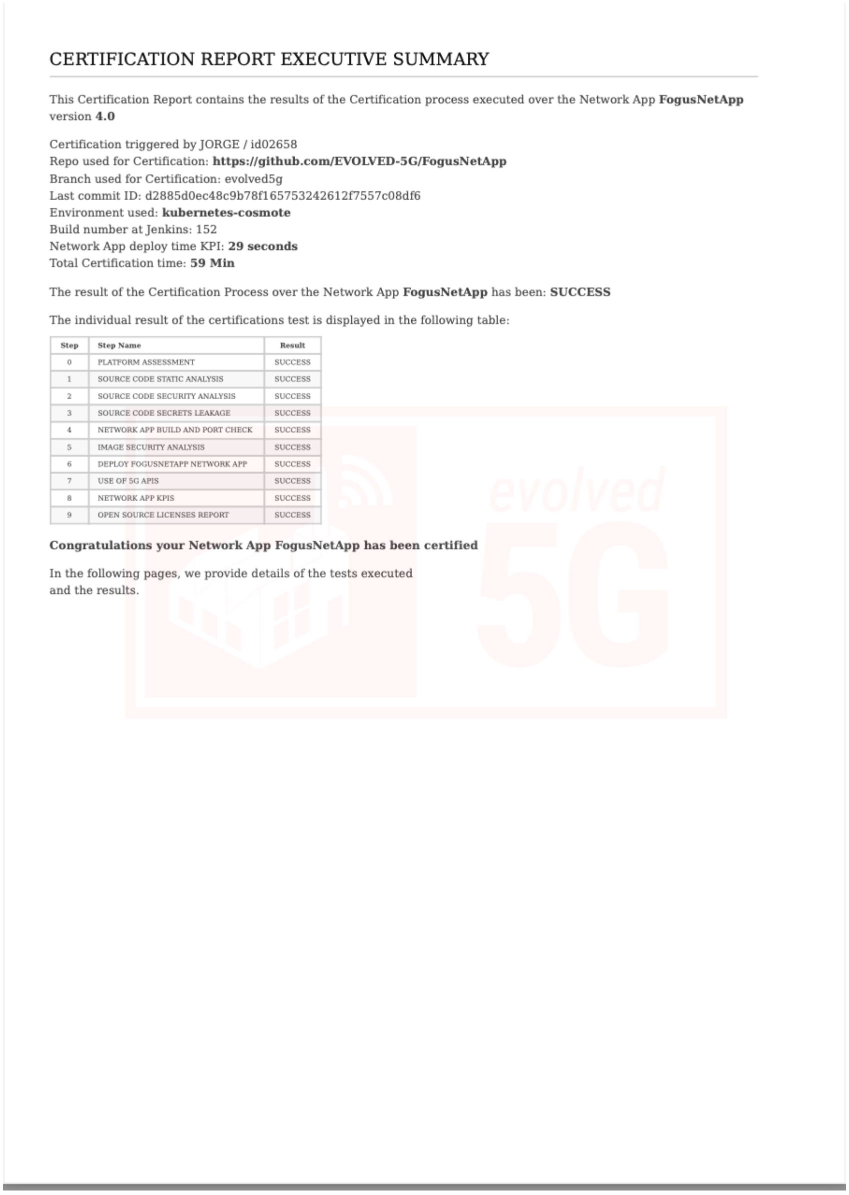
The Certification Report first page is the Cover page. It is a simple page that contains the Name of the Network Application and the Date of the execution of the Certification process.



Network App Certification Report:
FogusNetApp
Date: 20/11/2023

Figure 31: Cover page of the Certification Report

After the cover page, we find the Executive Summary of the Certification Process. This summary contains the administrative information of the execution of the certification process as well as the results of the Certification tests.



CERTIFICATION REPORT EXECUTIVE SUMMARY

This Certification Report contains the results of the Certification process executed over the Network App **FogusNetApp** version **4.0**

Certification triggered by JORGE / id02658
Repo used for Certification: **https://github.com/EVOLVED-5G/FogusNetApp**
Branch used for Certification: evolved5g
Last commit ID: d2885d0ec48c9b78f165753242612f7557c08df6
Environment used: **kubernetes-cosmote**
Build number at Jenkins: 152
Network App deploy time KPI: **29 seconds**
Total Certification time: **59 Min**

The result of the Certification Process over the Network App **FogusNetApp** has been: **SUCCESS**

The individual result of the certifications test is displayed in the following table:

Step	Step Name	Result
0	PLATFORM ASSESSMENT	SUCCESS
1	SOURCE CODE STATIC ANALYSIS	SUCCESS
2	SOURCE CODE SECURITY ANALYSIS	SUCCESS
3	SOURCE CODE SECRETS LEAKAGE	SUCCESS
4	NETWORK APP BUILD AND PORT CHECK	SUCCESS
5	IMAGE SECURITY ANALYSIS	SUCCESS
6	DEPLOY FOGUSNETAPP NETWORK APP	SUCCESS
7	USE OF 5G APIS	SUCCESS
8	NETWORK APP KPIS	SUCCESS
9	OPEN SOURCE LICENSES REPORT	SUCCESS

Congratulations your Network App FogusNetApp has been certified

In the following pages, we provide details of the tests executed
and the results.

Figure 32: Executive Summary page

The executive summary starts with the name of the Network Application being certified and the Version number.

This Certification Report contains the results of the Certification process executed over the Network App **FogusNetApp** version **4.0**

After this, it lists the name of the person/UserID that has triggered the certification. This UserID identifies the person in charge of the execution of the Certiication Process.

Certification triggered by JORGE / id02658

Then, the information about the repository used for the certification is dispayed, with the Github URL and the branch used inside the repository.

Repository used for Certification: **https://github.com/EVOLVED-5G/FogusNetApp** Branch used for Certification: evolved5g

The next information listed is the last commit included in this branch. This identifies the last change included in this version being certified.

Last commit ID: d2885d0ec48c9b78f165753242612f7557c08df6

Next is the information about the Certification environment used during the Certification. It can be UMA or COSMOTE, which are the two options for certification.

Environment used: **kubernetes-cosmote**

Next information is the ID number in Jenkins of the Pipeline execution. This number is useful in case some logs need to be inspected about the Certification execution. It also matches with the folder in Open Repository (Artifactory) where all the temporary results are stored during the certification.

Build number at Jenkins: 152

Next information tells the KPI Deployment Time. It measures the time it takes to deploy the Network Application in Kubernetes.

Network App deploy time KPI: **29 seconds**

Last information is the total time it took the Certification to complete. This measures the time it takes to complete all the steps in the Certification Pipeline.

Total Certification time: **59 Min**

Finally, the overall result of the Certification Process is displayed for the Network Application Developers. The Certification can be SUCCESS in case all Certification steps have been successful, or FAILURE in case any of the steps fails.

The result of the Certification Process over the Network App **FogusNetApp** has been: **SUCCESS**

A summary of the results of the certification steps in the pipeline is then provided for an overall evaluation of the results, as displayed in Figure 32.

This summary is followed by the final result of the certification:

Congratulations your Network App FogusNetApp has been certified

After the Executive Summary page, the following pages present the information and results of the different steps tested by the Certification Pipeline. The title of each page corresponds to one of the steps summarized in the Executive Summary.

First step (0) is the Platform Assessment. This assessment has been described as Step 0 in section 2.3.2, and the measurement process and technology has been described in deliverable 5.4 [6]. Basically, Jenkins performs a platform assessment based in Open5Genesis framework running an experiment that collects information about the 5G platform. It runs several tests collecting measurements for Latency, Jitter and Throughput and provides the statistical values of the results: Minimum, Maximum, Mean, Median and Standard Deviation.

Next page shows the results of the Source Code Static Analysis performed using SonarQube. This test has been described as Step 1 in section 2.3.2.

After that, Source Code Security Analysis is reported. This step is performed using Trivy tool. This test has been described as Step 2 in section 2.3.2.

Next is the Source Code Secrets Leakage, that is performed as well with Trivy tool. This test has been described as Step 3 in section 2.3.2.

After that, Network App Build and Port Check is reported. This step builds the Network Application images and checks that the ports described in the Dockerfile of the application are opened, so that connectivity to the Network Application is working. To check this connectivity, NMAP tool is used. This test has been described as Step 4 in section 2.3.2.

After the images have been built, Step 5 performs a Security Analysis of the images. Binary images contain not only the code from the Network Application but many libraries, the operative system, etc that might also bring vulnerabilities. Again, the severity of the vulnerabilities goes from CRITICAL to LOW with an UNKNOWN category. The Quality bar used in EVOLVED-5G is similar to Step 5. Only CRITICAL issues in the code of the Network Application will make this step Fail.

There will be a section for each of the containers defined in the Network Application. Each container will be listed as N/M being N the number between 1 and M and M the total number of containers.

Once the Network Application security has been analysed, it is safe to deploy the Network Application in Kubernetes and continue the testing with the 5G APIs. The Network Application is deployed along with CAPIF, NEF Emulator and TSN Application Function and API usage is collected form CAPIF. CAPIF APIs themselves generate Logs that are used to detect the CAPIF APIs used by the Network Application, while NEF Emulator and TSN AF uses the Logging APIs from CAPIF to report API usage.

The set of APIs detected are:

- ONBOARDING NETWORKAPP TO CAPIF: It means that the Network Application have successfully onboarded CAPIF and obtained his Invoker ID.
- DISCOVER NEF APIS FROM CAPIF: It means that the Network Application have successfully Discovered NEF and TSN APIs from CAPIF and obtained the API descriptors to use them.
- NEF SERVICES LOGGED AT CAPIF /nef/api/v1/3gpp-as-session-with-qos/: It means that the Network Application has consumed 3gpp-as-session-with-qos API from NEF Emulator.
- NEF SERVICES LOGGED AT CAPIF /nef/api/v1/3gpp-monitoring-event/: It means that the Network Application has consumed 3gpp-monitoring-event API from NEF Emulator.
- TSN SERVICES LOGGED AT CAPIF /tsn/api/: It means that the Network Application has consumed /tsn/api API from TSN AF.

Next page will show the Network Application KPIs. While deployed in Kubernetes, several KPIs are collected using Prometheus to report resources used by the Network Application during Certification tests. This has been described as Step 8 in section 2.3.2.

Finally, the license report is attached. Licensecheck tool is used to detect the licenses required to execute the Network Application. Licenses are explained in section 2.1.5. This is informative for the Network Application developer in EVOLVED-5G. We have not defined any Quality bar for avoiding specific type of licenses although it can be done.

The last page with relevant information is the fingerprint page. It shows the ID of the certification needed to publish the Network Application in the Marketplace.

Last page is the back cover with EVOLVED-5G logo.

3 Release to Marketplace

3.1 Network App Release to Marketplace

The EVOLVED-5G Marketplace (https://marketplace.evolved-5g.eu/) is the main interaction point between Network App creators and Network App consumers.

It targets 3 different user profiles:

1) The Network App creators: Developers publishing their Network Apps to a public catalog.
2) The Network App consumers: Users purchasing and using the Network Apps.
3) The Marketplace administrators: A group of administrators that have elevated access to view the platform's overall KPIs.

In this chapter focus is made on Network App creators as it describes step by step how the "Network App Onboarding" works.

The "Network App Onboarding" is the process of releasing an existing Network App to the Marketplace. It is assumed that the Network App creator has already published the Network App to the Open Repository and has received a fingerprint code as described in the previous chapters.

Network App Onboarding is implemented as wizard and hence it contains a series of steps the Network App creators must follow in the user interface. It is available at the URL https://marketplace.evolved-5g.eu/create-netapp. Each of the steps are described below:

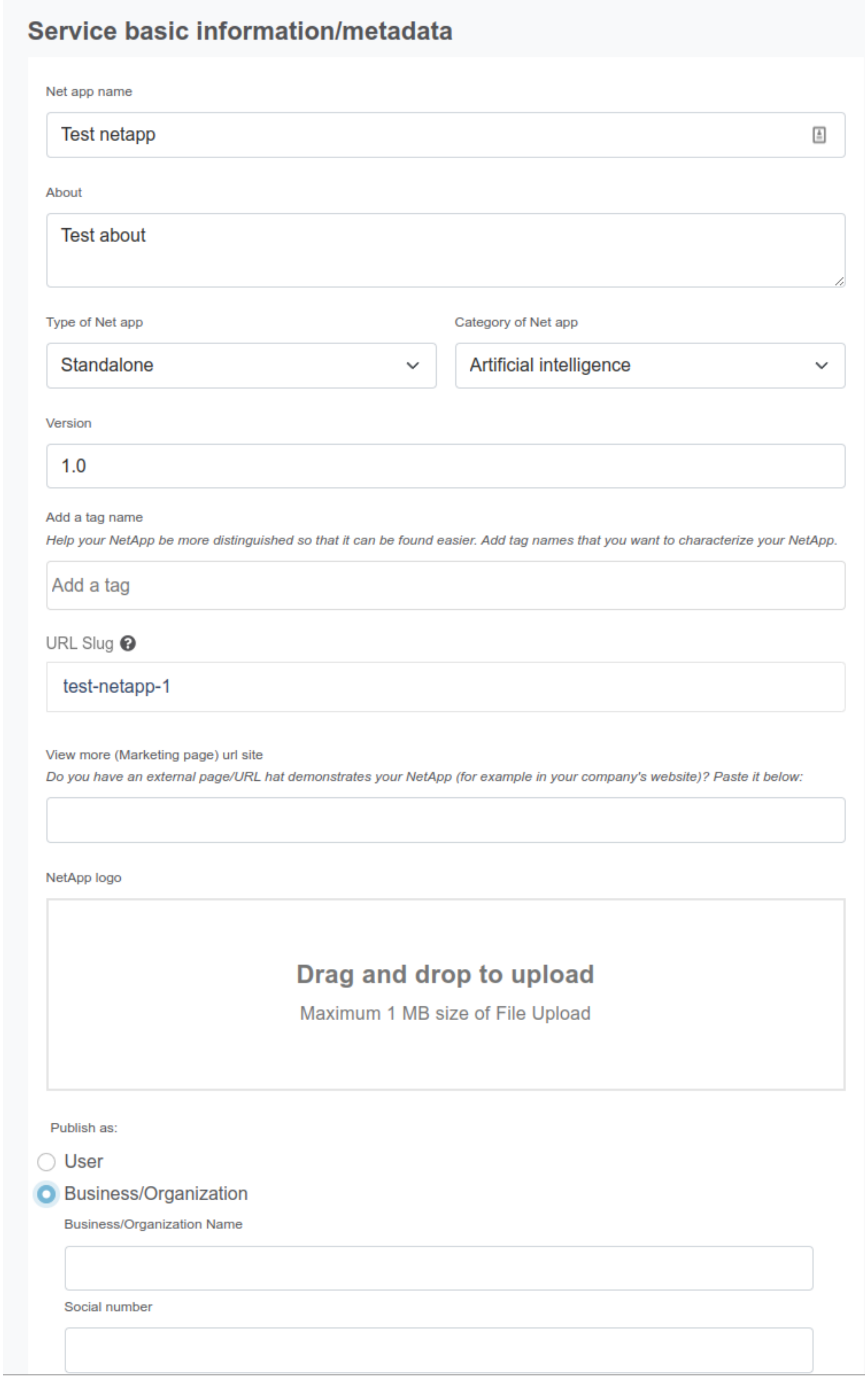
3.1.1 Step 1 - Service basic information/metadata

During this step, the user is prompted to enter Network App's basic details as well as some metadata. These details are:

1. Network App name.
2. Network App about text.
3. Type of the Network App (standalone or non-standalone).
4. Category of the Network App (currently, the categories offered are: Artificial Intelligence, Cyber security & cryptography, Identity and verification, Messaging services, Mobile carrier lending and advances, Mobile carrier subscriptions, Other
5. Version of the Network App.
6. Network App tag list (a list of tags that will help the users search for Network Apps with a corresponding tag).
7. URL slug. This describes the user-friendly URL via which the Network App will be accessed by. For example, in the following URL: https://marketplace.evolved-5g.eu/netapp-details/test-net-app , the slug part is "test-net-app".
8. Network App "view more" URL (a URL that the user will be able to visit in order to read more about the Network App, ex. a company page).

9. Network App logo image file.
10. Network App publisher (either user or a Business/Organization).

In the case of Business/Organization, the user is then prompted to also enter the Business/Organization name and Social Number. If the Network App creator decides to create a Network App as a Business entity and not an individual publisher, this will result as the Business being shown as the Network App publisher. These fields are also described in the following screenshot:



Service basic information/metadata

Net app name: Test netapp

About: Test about

Type of Net app: Standalone

Category of Net app: Artificial intelligence

Version: 1.0

Add a tag name
Help your NetApp be more distinguished so that it can be found easier. Add tag names that you want to characterize your NetApp.

Add a tag

URL Slug: test-netapp-1

View more (Marketing page) url site
Do you have an external page/URL hat demonstrates your NetApp (for example in your company's website)? Paste it below:

NetApp logo

Drag and drop to upload
Maximum 1 MB size of File Upload

Publish as:
○ User
● Business/Organization

Business/Organization Name

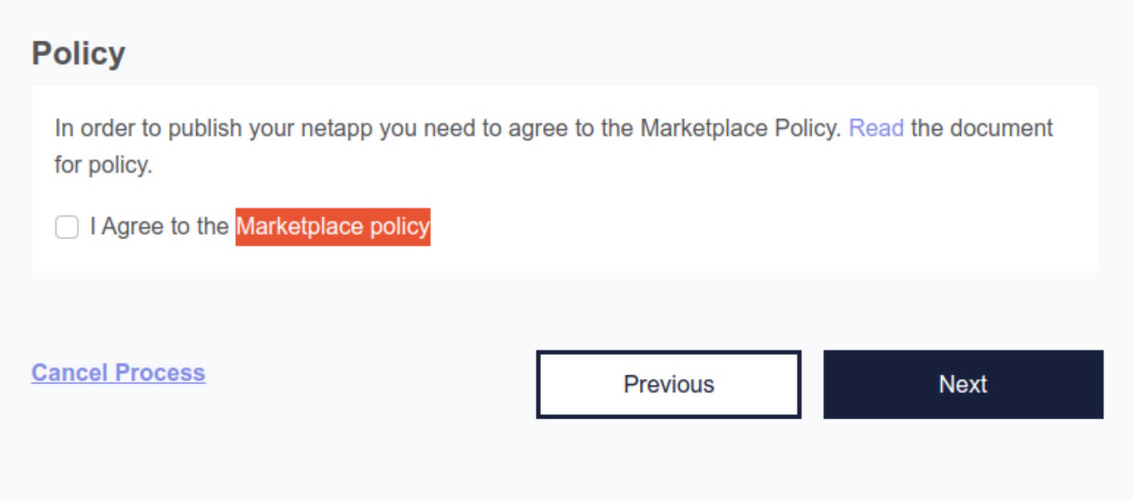
Social number

Figure 33: Service Basic Information Screen

After the user fills the fields and clicks on “Next”, in order to proceed to the next step of the Onboarding Wizard, the app performs a first-level validation, ensuring that all the required fields have the correct input. For example, the URL slug should be unique in the platform.

3.1.2 Step 2 - Marketplace Policy

During the second step, the user sees a minimal form that asks them to ensure that they comply with the Marketplace policy. This is done by clicking the relevant checkbox as show in Figure 34.



Policy

In order to publish your netapp you need to agree to the Marketplace Policy. Read the document for policy.

□ I Agree to the Marketplace policy

Figure 34: Marketplace policy

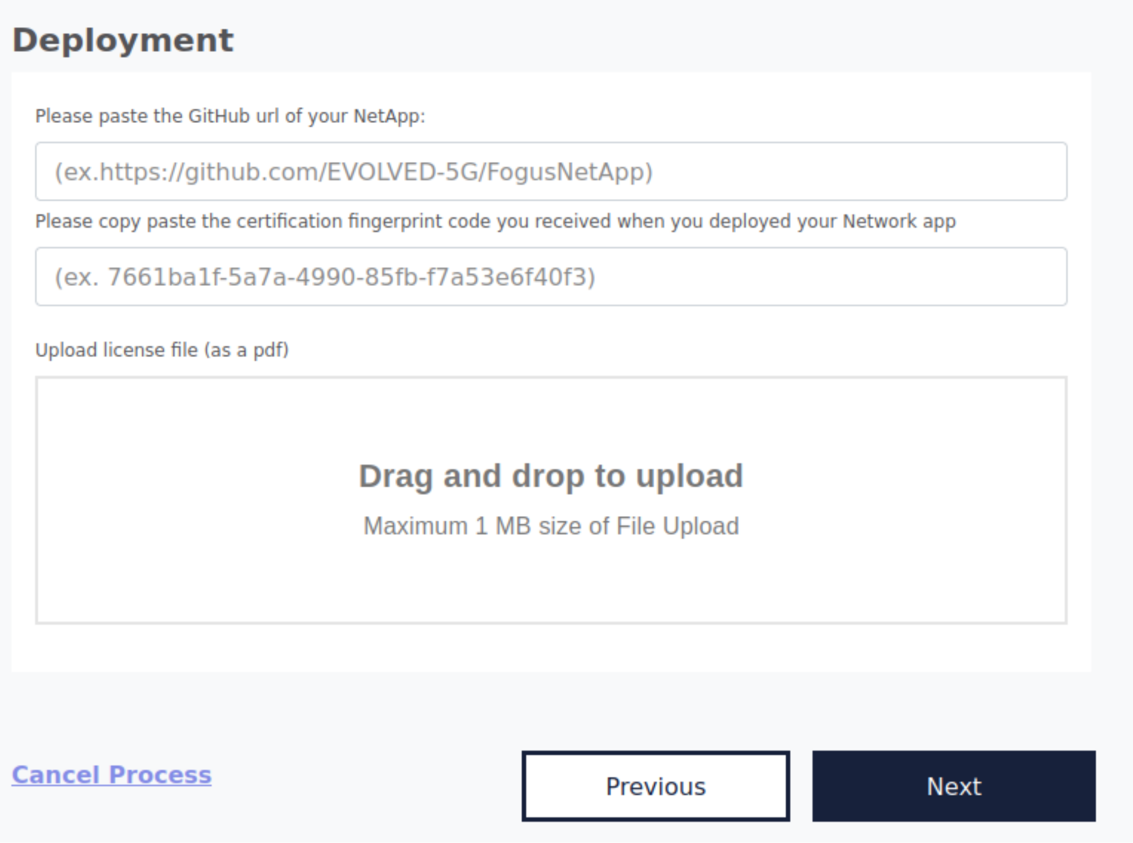
The checkbox should be filled in order to continue to the third step.

3.1.3 Step 3 - Deployment

During this step, the Marketplace’s component, namely “Certification report genuineness check” interacts with the Open Repository to make sure that the Network App has gone through the process of validation and certification and, that it has been published in the Open repository, thus making it ready to be consumed and published to the Marketplace. The communication between the Marketplace and the Open Repository occurs under a private network. The Open Repository exposes an appropriately private http/https endpoint that allows the marketplace to retrieve related information about a specific Network App. The endpoint has the following form http://artifactory.hi.inet/artifactory/{netapp_name_in_github}/{version}/fingerpring.json where {netapp_name_in_github} is the name of the Network App as it appears in the Evolved5G GitHub repository and {version} is the version of the Network App. Using this endpoint, the Marketplace is able to retrieve a file in JSON format (fingerprint.json) that contains a code (from now on to be referred to as “fingerprint”) that is related to that specific Network App.

Only the Network App creators know this fingerprint code. The Network App creators receive the fingerprint code via an email, while deploying the Network App to the Open Repository.

The Marketplace requires the user to paste this fingerprint code in the user interface and it automatically validates its existence by interacting with the Open Repository in the background. The process is presented in the screenshot below:



Deployment

Please paste the GitHub url of your NetApp:

(ex.https://github.com/EVOLVED-5G/FogusNetApp)

Please copy paste the certification fingerprint code you received when you deployed your Network app

(ex. 7661ba1f-5a7a-4990-85fb-f7a53e6f40f3)

Upload license file (as a pdf)

Drag and drop to upload

Maximum 1 MB size of File Upload

Figure 35: Deployment of the Network App in the Marketplace

In this step, the user needs to enter the following information:

1. The GitHub URL of the Network App: The netapp_name_in_github will be extracted from this URL.
2. The Fingerprint Code.
3. Upload the License file (optional).

Marketplace will then make a call to the Open Repository in order to receive the existing fingerprint file for the given Network App. If the fingerprint.json file is found and the fingerprint code in the file is the same as the fingerprint code entered by the user, then the Network App is considered as verified with the Open Repository. This process is depicted in the following diagram:

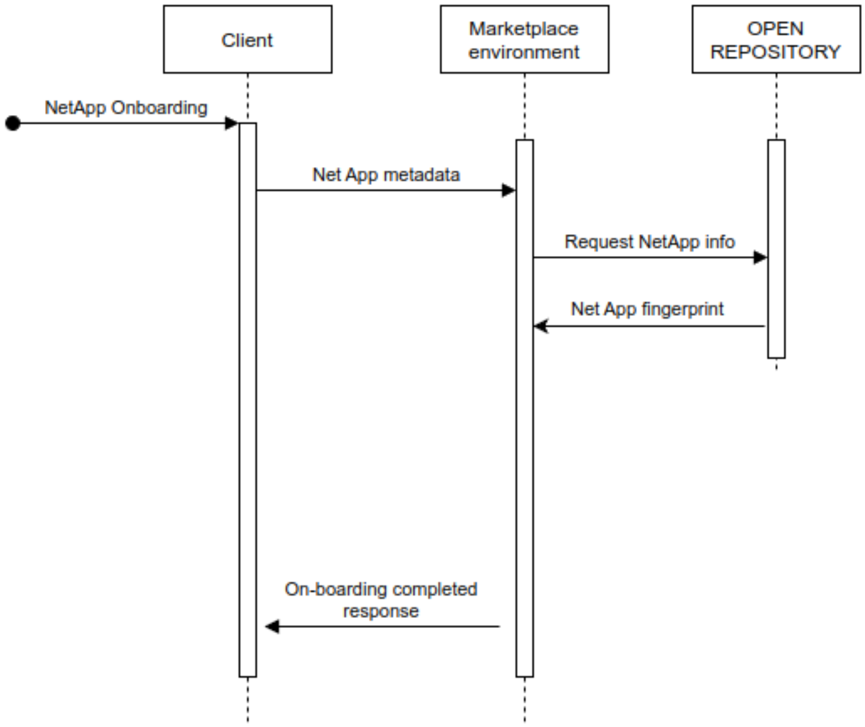


Figure 36: Verification OpenRepository - Marketplace

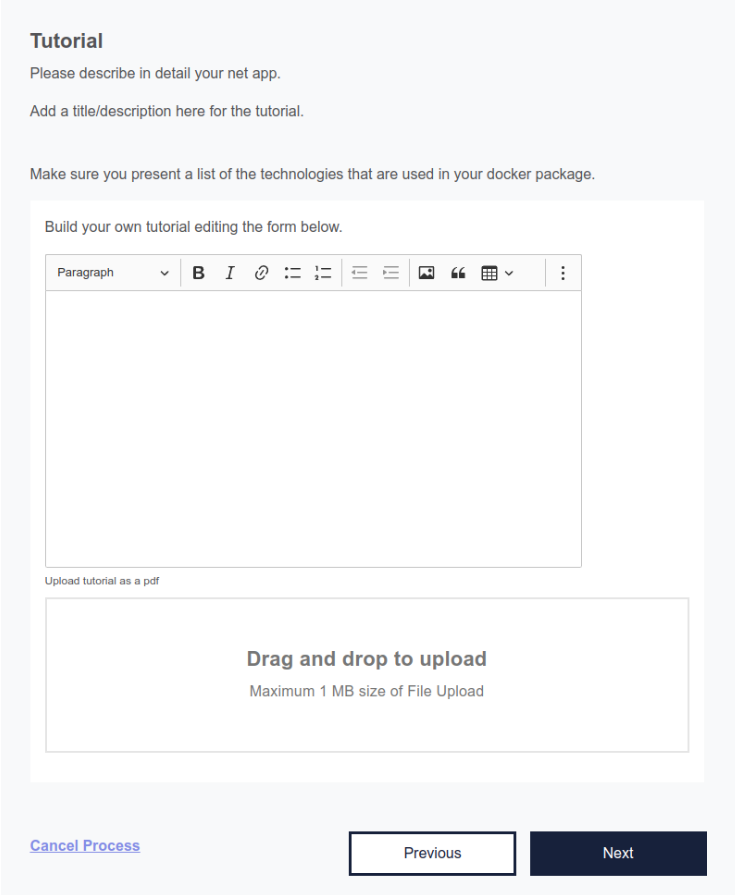
If the fingerprint.json file is not found, or if the fingerprint code is different than the one in fingerprint.json, then the platform informs the user accordingly, with an error message.

3.1.4 Step 4 - Tutorial

In the next step, the user is prompted to enter the details for the tutorial of the Network App. This can be done in two ways, by entering the text in a text area or by uploading a file in PDF format for the tutorial (optional). The essence of the tutorial is very important.

The Network App creator needs to describe exactly how the Network App and its corresponding services are deployed, used, and utilized. So, the tutorial is actually what the user who will use the Network App will need to read and understand, in order to use the Network App.

For example, the tutorial might include some practical examples with the required commands that the user needs to follow in order to use the Network App Docker image. The fourth step looks like the following screenshot:



Tutorial

Please describe in detail your net app.

Add a title/description here for the tutorial.

Make sure you present a list of the technologies that are used in your docker package.

Build your own tutorial editing the form below.

Upload tutorial as a pdf

Drag and drop to upload

Maximum 1 MB size of File Upload

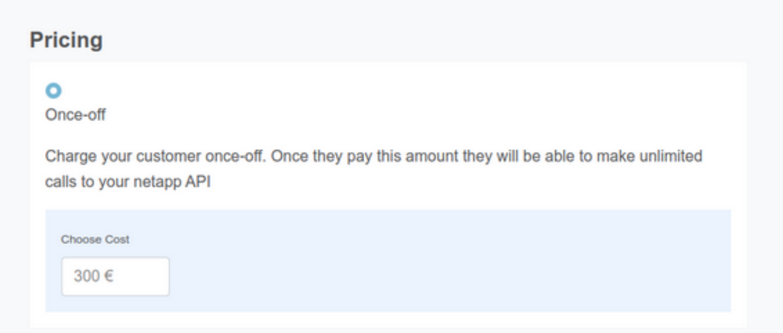
Figure 37: Tutorial Marketplace screen

3.1.5 Step 5 - Pricing Wizard

This is the last step of the Network App creation process. In this step, the user is prompted to enter the pricing details for their Network App. It is worth mentioning that during the EVOLVED-5G project no payments will take place in the platform. All the Network Apps will have zero cost. Though the purpose of the pricing wizard is to demonstrate the available options a Network App creator could have under a production setting, where the Network App consumer is charged in order to use the Network App. Using Price wizard, Network App creators can help Network App consumers understand how they will be charged if they choose to use the Network App.

Marketplace supports two pricing modes: The pricing can be either “Once off”, meaning that the buyer will have to pay a one-time fee, or “Pay as you go”, meaning that the buyer will have to periodically pay a variable amount of money depending on the use they make.

If the Network App creator selects “Once off”, they only need to enter the amount in euros as depicted below:



Pricing

● Once-off

Charge your customer once-off. Once they pay this amount they will be able to make unlimited calls to your netapp API

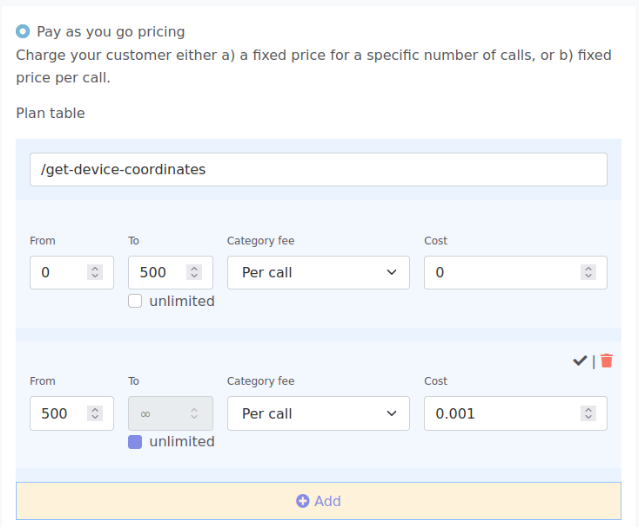
Choose Cost: 300 €

Figure 38: Pricing wizard Marketplace

If the Network App creator selects “Pay as you go pricing”, they then have to describe the charge with either a) a fixed price for a specific number of calls, or b) fixed price per call. The Network App creator can then add specific endpoints of the Network App service and describe a payment scheme for the API calls per endpoint. Let’s consider a scenario where Network App can be used in order to track devices in a 5G network. The Network App exposes an API that allows Network App consumers to request for the coordinates of a specific device by making a call to an endpoint “/get-device-coordinates”. The Network App creators want to charge based on the number of requests that are made in this endpoint. Using the Price Wizard, they can make the following configuration:

“The first 500 requests to endpoint /get-device-coordinates is free. All subsequent requests have a fixed cost of 0.001€ per call”

The following picture illustrates this scenario:



● Pay as you go pricing

Charge your customer either a) a fixed price for a specific number of calls, or b) fixed price per call.

Plan table

/get-device-coordinates

From	To	Category fee	Cost
0	500 □ unlimited	Per call	0
500	∞ ☑ unlimited	Per call	0.001

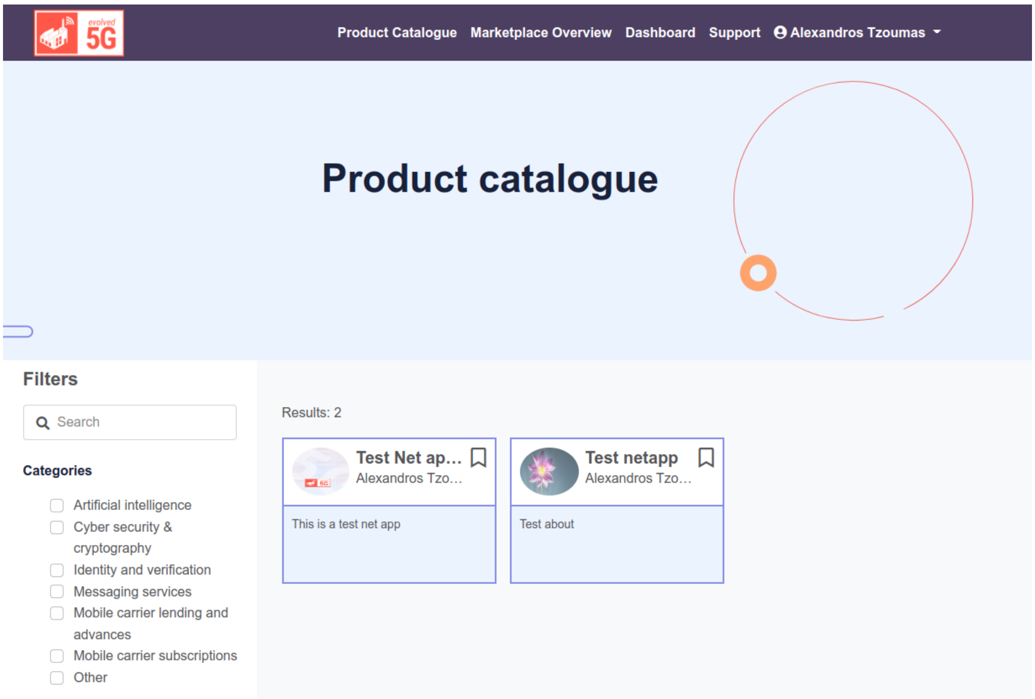
Figure 39: Network App price example

3.1.6 Step 6 - Release to the Marketplace

Once all the above-mentioned steps are completed, the Network App is Onboarded to the Marketplace, but it is still not publicly available to the Product Catalogue. Via the Edit Network App page a Network App creator can the Network App status from “private” to “public”, in order to make the Network App available to the Product catalogue.

3.2 Purchase of a Network App in the Marketplace

The Product Catalog page consists of a list of published Network Apps which have been verified with the Open Repository and are available for purchase. For each Network App listed, the user can click and view all the relevant details, understand what the Network App does and which problem it solved, as well as review the pricing information:



Product catalogue

Filters

Search

Categories

□ Artificial intelligence
□ Cyber security & cryptography
□ Identity and verification
□ Messaging services
□ Mobile carrier lending and advances
□ Mobile carrier subscriptions
□ Other

Results: 2

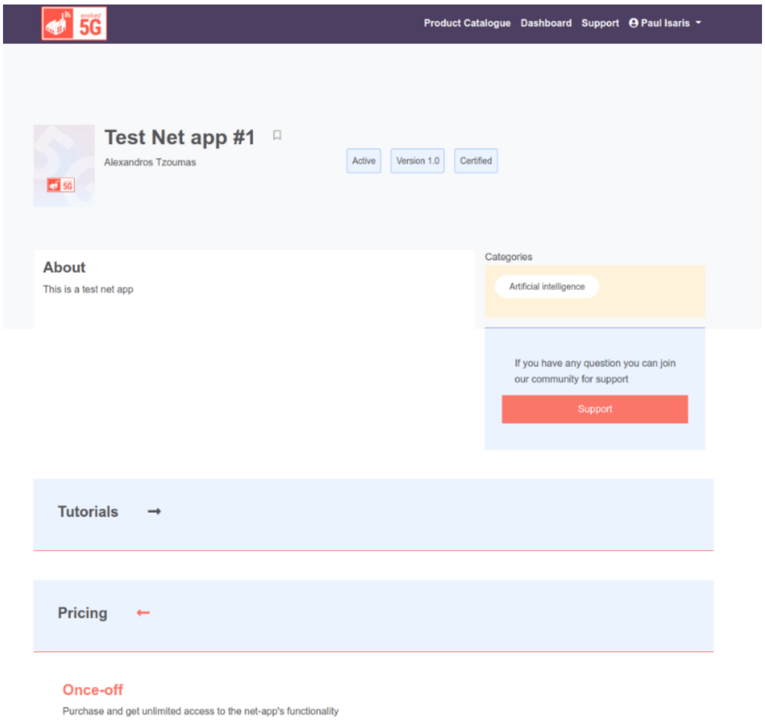
Test Net ap…
Alexandros Tzo…
This is a test net app

Test netapp
Alexandros Tzo…
Test about

Figure 40: Marketplace Product Catalogue

In Figure 40, registered users can search for Network Apps by using the filters on the left-side menu.

When the user clicks on one of the Network Apps, they are taken to the Network App public page, where they can read more about the Network App, as well as purchase it:



Test Net app #1
Alexandros Tzoumas

Active | Version 1.0 | Certified

About
This is a test net app

Categories
Artificial intelligence

If you have any question you can join our community for support

Tutorials →

Pricing ←

Once-off
Purchase and get unlimited access to the net-app's functionality

Figure 41: Purchase final screen

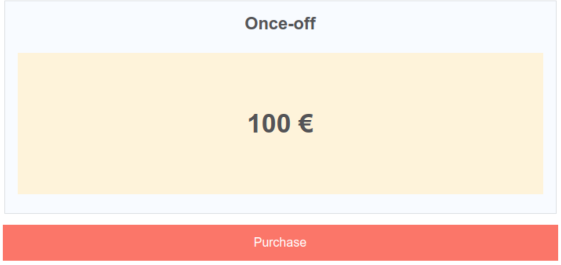
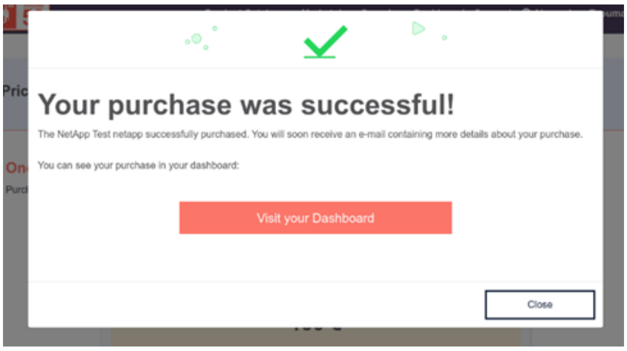


Figure 42: Network App purchase final screen

If the user clicks "Purchase", the purchase is then completed in the background, and the user sees a confirmation message:



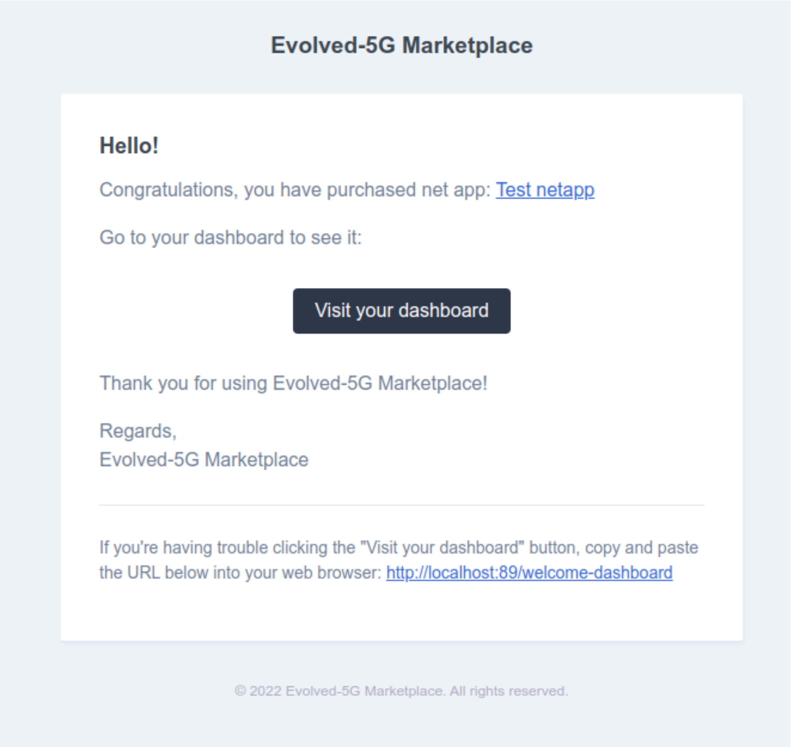
Your purchase was successful!

The NetApp Test netapp successfully purchased. You will soon receive an e-mail containing more details about your purchase.

You can see your purchase in your dashboard:

Figure 43: Purchase succeeded

Now the user has completed the Network App purchase. They will receive an automated email confirming the purchase:



Evolved-5G Marketplace

Hello!

Congratulations, you have purchased net app: Test netapp

Go to your dashboard to see it:

Thank you for using Evolved-5G Marketplace!

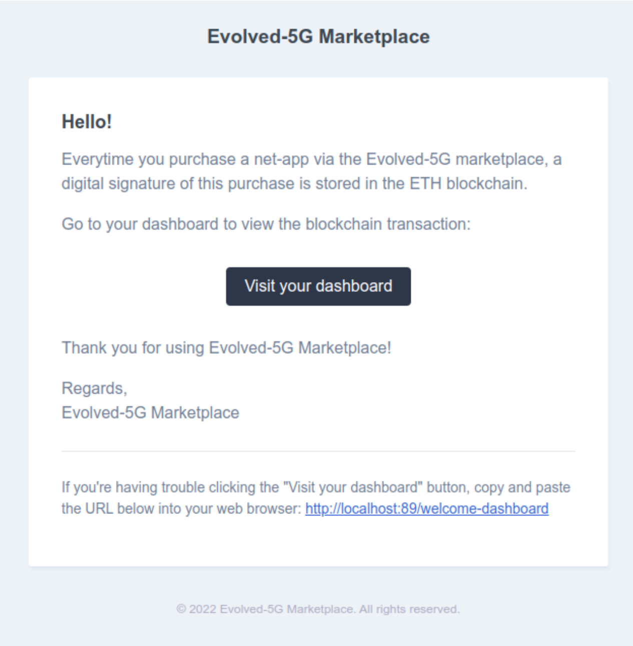
Regards,
Evolved-5G Marketplace

If you're having trouble clicking the "Visit your dashboard" button, copy and paste the URL below into your web browser: http://localhost:89/welcome-dashboard

© 2022 Evolved-5G Marketplace. All rights reserved.

Figure 44: Dashboard Marketplace

Then, in the background, the app connects to the Ethereum Network, in order to log a digital signature of this purchase to the Blockchain Network. When the Blockchain transaction is completed, the user receives another automated email, notifying them about the transaction:



Evolved-5G Marketplace

Hello!

Everytime you purchase a net-app via the Evolved-5G marketplace, a digital signature of this purchase is stored in the ETH blockchain.

Go to your dashboard to view the blockchain transaction:

Thank you for using Evolved-5G Marketplace!

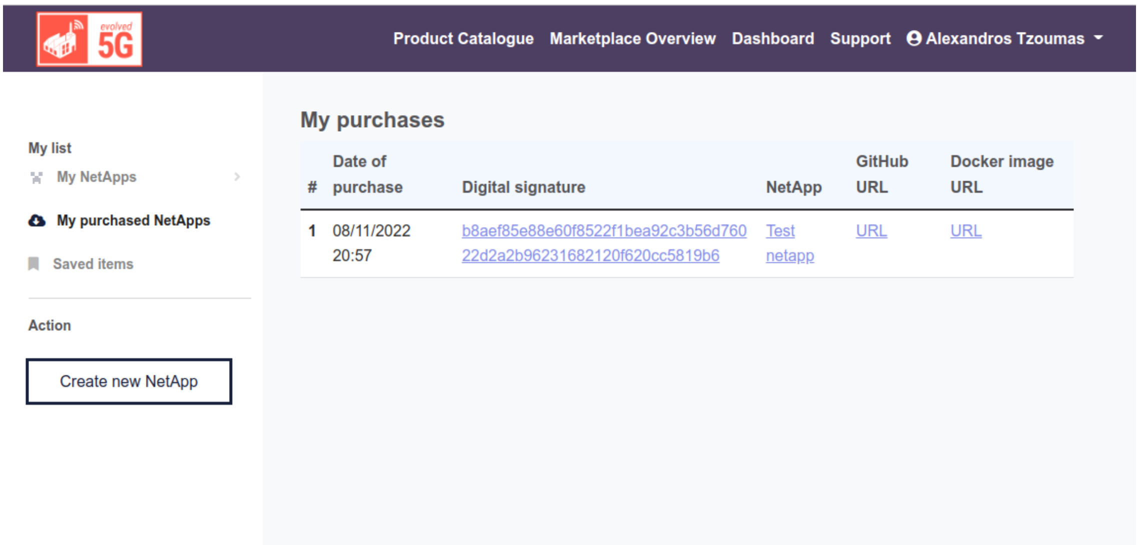
Regards,
Evolved-5G Marketplace

If you're having trouble clicking the "Visit your dashboard" button, copy and paste the URL below into your web browser: http://localhost:89/welcome-dashboard

© 2022 Evolved-5G Marketplace. All rights reserved.

Figure 45: Purchase confirmation

This hash string is then shown as a **Digital Signature** in the "My purchased Network Apps" page, which is accessible via the "Dashboard" page:

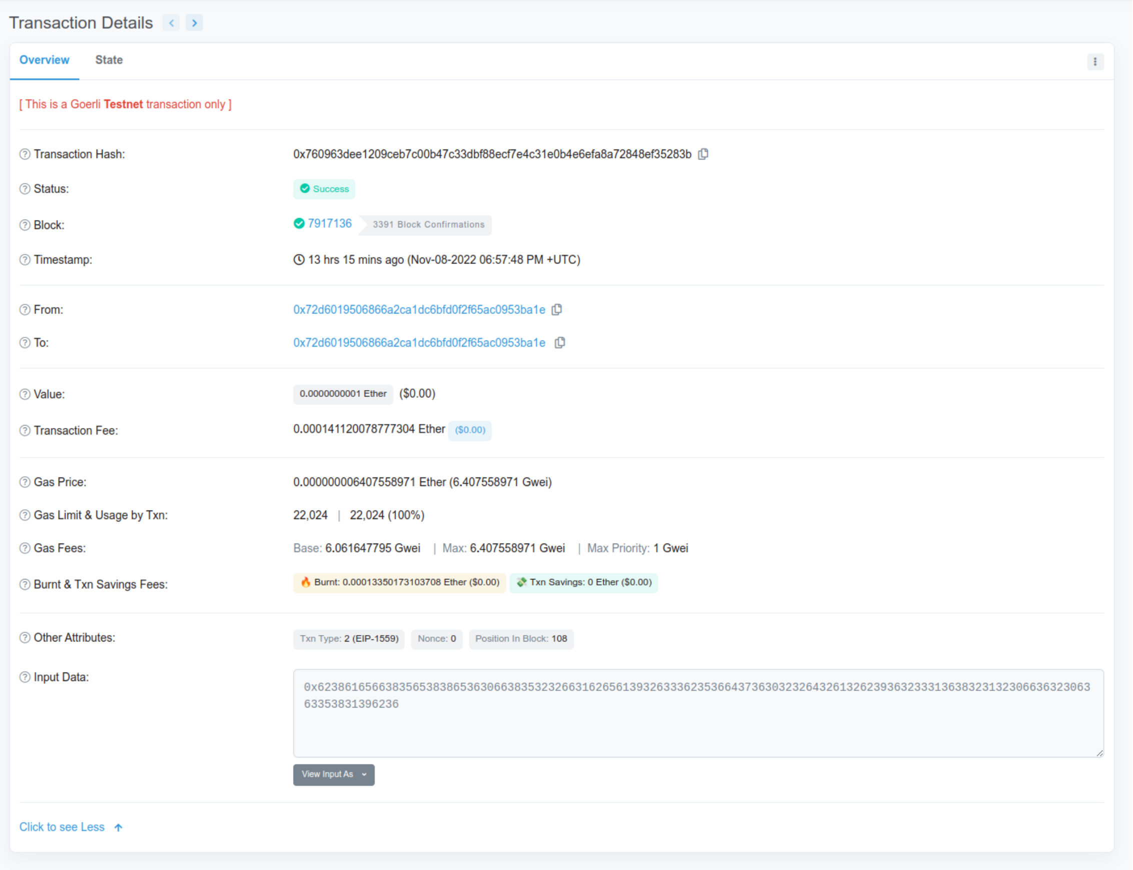


My purchases

#	Date of purchase	Digital signature	NetApp	GitHub URL	Docker image URL
1	08/11/2022 20:57	b8aef85e88e60f8522f1bea92c3b56d760 22d2a2b96231682120f620cc5819b6	Test netapp	URL	URL

Figure 46: Marketplace Digital Signature

When the user clicks on the Digital Signature link, they are taken to an Etherscan page, where they can view the corresponding Blockchain transaction:



[This is a Goerli **Testnet** transaction only]

Transaction Hash:	0x760963dee1209ceb7c00b47c33dbf88ecf7e4c31e0b4e6efa8a72848ef35283b
Status:	Success
Block:	7917136 3391 Block Confirmations
Timestamp:	13 hrs 15 mins ago (Nov-08-2022 06:57:48 PM +UTC)
From:	0x72d6019506866a2ca1dc6bfd0f2f65ac0953ba1e
To:	0x72d6019506866a2ca1dc6bfd0f2f65ac0953ba1e
Value:	0.0000000001 Ether ($0.00)
Transaction Fee:	0.000141120078777304 Ether ($0.00)
Gas Price:	0.000000006407558971 Ether (6.407558971 Gwei)
Gas Limit & Usage by Txn:	22,024 | 22,024 (100%)
Gas Fees:	Base: 6.061647795 Gwei | Max: 6.407558971 Gwei | Max Priority: 1 Gwei
Burnt & Txn Savings Fees:	Burnt: 0.00013350173103708 Ether ($0.00) Txn Savings: 0 Ether ($0.00)
Other Attributes:	Txn Type: 2 (EIP-1559) Nonce: 0 Position In Block: 108
Input Data:	0x623861656638356538386536306638353232663162656139326333623536643736303232643261326239363233313638323132306636323063 63353831396236

Figure 47: Marketplace Blockchain details

Finally, on the same page, the user can download the docker related files of the Network App in order to start using the Network App.

3.3 Network App Version Upgrade

Network App creators specify the Network App version during the deployment phase of the Network App. Once the Network App is deployed, the Network App creator has to copy and paste this version number during the Onboarding Process of the Network App.

The Onboarding Process in the Marketplace is always tightened to one specific version. So, if Network App creators want to support multiple versions of the same Network App, they will have to create separate entries in the Product Catalogue. Each version will have a separate single page in the Marketplace and the differences between the supported versions should be explained in the tutorials.

3.4 Removing a Network App from the Marketplace

A Network App can take two statuses in the Marketplace:

a) Public (available for purchase and view by all platform users)
b) Private (available and shown only to the Network App creator)

When a Network App creator decides not to list their Network App anymore, they can select to make it “Private”, by editing the Network App status. When set to private, the Network App is not removed or deleted from the Marketplace's database, but it is restricted to be shown only to its creator and is not available in the Marketplace.

This is useful in scenarios where multiple versions have been uploaded to the marketplace, but only the latest is visible to the product catalogue. At the same time Network App consumers will have access to any older version they may have purchased.

4 Conclusions

This deliverable is the last deliverable coming out from WP5 in EVOLVED-5G. It collects the latest details about the implementation of the Certification Process, the production of the Certification Reports for certified Network Applications, and how these applications can be released to the Marketplace. Once an application is released, it can be purchased by any Marketplace user, it can be upgraded when new versions of the applications are produced, and it can be removed from the marketplace when discontinued.

WP5 has been the work package where WP2, WP3 and WP4 have materialised and integrated the EVOLVED-5G vision. While WP2 defined the requirements and outlined the architecture for EVOLVED-5G, WP3 created the Development experience and the tools for Developers to enable the creation of EVOLVED-5G Network Applications. WP4 has been the place for SMEs to develop, build, test and integrate their developments.

All results from these WPs have materialised in WP5. Task 5.1 has extended Málaga and Athens platforms with additional components (such as TSN in Málaga) and additional tests defined (e.g., NEF, TSN and CAPIF) and the platforms have supported the SMEs in testing their applications with real 5G infrastructure.

Task 5.2 has integrated the environments coming from Task 5.1 with the testing and the development of the Validation Process that includes the Validation pipeline. This Validation process uses the tools developed un in WP3 such as CAPIF, NEF and TSN, but also commercial tools such as SonarQube or Trivy for quality assessment. Manual validation of the Network Applications has taken place as well as the Automated Validation of the Network Applications generating the Validation Reports included in deliverable 5.5.

Finally, Task 5.3 has added the Certification Process and the corresponding Certification Pipeline that generates the Certification reports (see Annex) and has integrated this certification process with the Marketplace developed in WP3 to enable the publication of the Network Applications in the Marketplace which materializes the vision of EVOLVED-5G as a project.

EVOLVED-5G has created the Platforms, Tools and Processes to enable the Validation and Certification of Network Applications built upon Programmable 5G Networks.

5 REFERENCES

[1] EVOLVED-5G, Deliverable 5.3 “Network Apps Certification and Release to Marketplace (Intermediate)”

[2] Licensecheck tool: https://github.com/google/licensecheck

[3] SPDX Project from Linux Foundation: https://spdx.dev/about/overview/

[4] Prometheus: https://prometheus.io

[5] Jenkins: https://www.jenkins.io

[6] EVOLVED-5G, Deliverable 5.4 “System level evaluation and KPI analysis (Final)

[7] EVOLVED-5G, Deliverable 3.4 “Network Apps Certification Tools and Marketplace development (final)”

[8] https://trivy.dev

Annex – Certification Reports of the Network Applications

Along with this deliverable, we provide the Certification Reports issued for all the Network Applications Developed during the Project. Each Certification report is a separate PDF file and the corresponding filenames are illustrated in this table:

Network Application	Certification Report File
FogusNetApp	final_report_FOGUS_COSMOTE.pdf
GmiAeroNetApp	final_report_GMI_COSMOTE.pdf
InfolysisNetApp	final_report_INFOLYSIS_COSMOTE.pdf
IninRmonNetApp	final_report_ININ_COSMOTE.pdf
ZortenetNetApp	final_report_ZORTENET_COSMOTE.pdf
8BellsNetApp	final_report_8Bells.pdf
CafaTechNetApp4	final_report_CAFATECH.pdf
ImmersionNetApp	final_report_IMM.pdf
IQB-NetApp	final_report_IQB.pdf
TeleopNetApp	final_report_PAL_Teleopenapp.pdf
UmaCsicNetApp	final_report_UMA.pdf
LocalizationNetApp	final_report_UML_Localization.pdf

Table 2: Network Applications and Certification Reports Table