Deliverable D4.1
5G Security testbed architecture

Project name	5G Enablers for Network and System Security and Resilience	
Short name	5G-ENSURE	
Grant agreement	671562	
Call	H2020-ICT-2014-2	
Delivery date	30.06.2016	
Dissemination Level:	Public	
Lead beneficiary	b<>com	Sergio Morant, Sergio.morant@b-com.com
Authors	b<>com : Sergio Morant Orange: Jean-Philippe Wary VTT: Esa Piri	

European Commission

5G PPP

Executive summary

One of the major challenges of the 5G-ENSURE project is to provide a testbed environment allowing to evaluate and validate the efficiency of the 5G-ENSURE security enablers in order to address the security requirements of 5G Networks.

This deliverable provides the description of the testbed leveraging on results achieved at project level regarding both security enablers targeted but also security architecture as well as taking advantage of sources of information coming from 5G-PPP (e.g. taking into account the recommendations of 5G-PPP architecture Working Group see "5G-PPP Architecture whitepaper" [1]). It also describes the framework provided by the partners involved in the testbed activities including the hardware and the proposed services.

Another important aspect covered by the document is the interconnection of the testbed at the following levels: partner's testbed facilities interconnection, partner's remote access, Internet access and possibly interconnection with other existing 5G-PPP testbeds.

The last topic covered by the deliverable is the operational procedures required to drive the common activities on the 5G security testbed and the different roles that have been yet identified to accomplish these activities.

Foreword

This deliverable provides the foundations to build the 5G-ENSURE testbed that will host the 5G security Enablers of the project and enable the testing activities from M12 to M24. The contents of this document will also help the developers and the rest of the partners to understand how the functionalities are going to be delivered. The first vision of the roadmap for the 5G security enabler's integration on the testbed is provided.

Disclaimer

The EC flag in this deliverable is owned by the European Commission and the 5G PPP logo is owned by the 5G PPP initiative. The use of the flag and the 5G PPP logo reflects that 5G-ENSURE receives funding from the European Commission, integrated in its 5G PPP initiative. Apart from this, the European Commission or the 5G PPP initiative have no responsibility for the content.

Copyright notice

© 2015-2017 5G-ENSURE Consortium

Abbreviations

AAA	Authentication, Authorization and Accounting
AKA	Authentication and Key Agreement
AP	Access Point
API	Application Programming Interface
COTS	Commercial off the self
CRON	Command Run On
DNS	Domain Name Service
EAP	Extensible Authentication Protocol
EC	European Commission
ePC	evolved Packet Core
ETSI	European Telecommunications Standards Institute
HLD	High Level Design
HPC	High Performance Computing
HSS	Home Subscriber Server
IKE	Internet Key Exchange
IT	Information Technology
KVM	Kernel Virtual Machine
LDAP	Lightweight Directory Access Protocol
LTE	Long Term Evolution
LTS	Long Term Support
MME	Mobility Management Entity
MTU	Maximum Transmission Unit
NAS	Network-Attached Storage
NDA	Non-Disclosure Agreement
NFS	Network File System
NFV	Network Function Virtualization
NREN	National Research & Education Network
NTP	Network Time Protocol
OAI	Open Air Interface
OS	Operating System
OVS	Open Virtual Switch
PAM	Pluggable Authentication Module
PDN	Packet Data Network
P-GW	PDN Gateway
PPP	Public Private Partnership
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RTT	Round-Trip Time
SAN	Storage Area Network
SDN	Software Defined Network
SFC	Service Function Chaining
S-GW	Serving Gateway
SIM	Subscriber Identity Module
SSH	Secure Shell

TSC	Time Stamp Counter
UE	User Entity
USRP	Universal Software Radio Peripheral
VNF	Virtual Network Function
VPN	Virtual Private Network

Contents

1 Introduction

The following chapters of this document cover the requirements identified for the testbed deployment and operation. These requirements have been collected since the beginning of the 5G-ENSURE project (see section 2.1), aggregated, improved and reviewed by all partners.

The collection of enabler's hosting requirements began with a hosting criteria evaluation survey which was distributed to the Enabler Owners. The outcomes are summarised in Appendix B. If new requirements for the 5G security testbed are identified later during the project, they will be evaluated and the strategy of implementation will be agreed between the parties. This document essentially focuses on Release R1 enablers (as defined inside deliverables D3.1 [2] and D3.2 [3]).

Note that the ultimate goal of the testbed is to validate not only the enablers, but also the entire 5G-ENSURE architecture which they support. However, due to the time plan of the overall project (with the architecture work running in parallel) it is simply impossible to capture all architectural requirements at this point in time. However, this report has been reviewed by the Architecture taskforce (Task 2.4 of 5G-ENSURE), based on the current architectural working hypothesis. As the draft architecture is later completed (D2.4, M12), additional feedback will provided as necessary for possible later refinements of the testbed. Note also that already now, it is clear that certain architectural aspects cannot be *fully* tested within the scope of 5G-ENSURE. For example, network slicing is already now identified as a key component of the 5G architecture. However, 5G-ENSURE is not developing a *complete* set of enablers for network slicing and therefore all aspects of network slicing will not be possible to test. The inability to fully test an important architectural concept such as slicing may at first seem unsatisfactory. However, note that the ongoing discussions in 3GPP still leave it quite open whether a complete solution for network slicing will *ever* be fully standardized. Thus, the potential validation of any concrete and complete implementation of slicing would surely give interesting insights, but is not critical at this time.

As the testbed is in its early deployment stage, some of the identified tools are not yet operational. Thus, the description of these tools and their operational procedures are described at a high level in order to provide to the reader with a comprehensive description even though they are not yet available.

During this phase of testbed architecture design, we have identified the need to establish and define specific NDA and Charter to improve the content of existing 5G-ENSURE Project Consortium Agreement. Those two additional documents will be defined and delivered inside the document D4.2 "Test plan". They will cover the need to describe and establish rules for:

- interconnection between remote systems / platforms and the testbed,
- the resources (HW and SW) delivering and allocation between partners and users, and relative access rights management,
- the usage of the testbed and partners respective Intellectual Property Rights protection,
- Data and Confidential Information management, Privacy and result's ownership,
- Prevention of abnormal behaviours and process to handle potential conflict.

The whole document is organized in 4 major parts: the testbed architecture, the testbed hosting framework, the testbed connectivity and the testbed operational procedures.

1.1 General definitions

This section covers the definition of a set of terms used in the following chapters of this deliverable:

1.1.1 Orchestration

It is the automated arrangement, coordination, and management of complex computer systems, middleware and services [4].

Within the context of the 5G-ENSURE testbed, it refers to the tools allowing the automatic deployment and configuration of all the enabler instances running on the testbed. This role is not to confound with the one described for 5G networks MANO environment by ETSI NFV group.

1.1.2 Hypervisor

It is a computing node with virtualization hosting capabilities. This means that it is able to run Virtual Machines (i.e. VMs) on top of its Operating System.

1.1.3 Black box

A black box is a device, system or object which can be viewed in terms of its inputs and outputs (or transfer characteristics), without any knowledge of its internal workings. Its implementation is "opaque" (black) [5].

Within the perimeter of the testbed, it refers to components of the architecture that are delivered as functional block but which are not suitable to be modified.

1.1.4 White box

Machine or system whose internal structure or processing is known in addition to the knowledge about its inputs, outputs, and the relationship between them [6].

Within the scope of the testbed, this term refers to functional components that are suitable to be adapted to support new features like those ones brought by the 5G security enablers developed within 5G-ENSURE.

1.1.5 Service function chaining

A service function chain defines an ordered or partially ordered set of abstract service functions (SFs) and ordering constraints that must be applied to packets, frames, and/or flows selected as a result of classification.

An example of an abstract service function is a firewall. The implied order may not be a linear progression as the architecture allows for SFCs that copy to more than one branch, and also allows for cases where there is flexibility in the order in which service functions need to be applied. The term "service chain" is often used as shorthand for "service function chain" [7].

1.2 Testbed roles definition

All the roles that will be used to define the operational procedures are described in this section.

1.2.1 Testbed User

Partners requiring to run activities or have access to the 5G-ENSURE testbed. They have access to the testbed resources and can request assistance when dealing with the 5G-ENSURE testbed operational procedures described in this chapter.

1.2.2 Testbed Owner

It is the partner providing one of the nodes hosting the 5G-ENSURE security testbed. Current testbed owners are VTT and b<>com.

1.2.3 Testbed Support

It provides assistance to testbed users with regard to testbed activities. This applies typically to the execution issues related to the 5G-ENSURE testbed operational procedures described in this chapter. It should be composed of 5G-ENSURE Testbed partners.

1.2.4 Testbed Operator

It manages enablers' deployment within testbed and the operational status of the infrastructure. Its mission is mainly the orchestration of the framework and the provided tools. It is mainly composed of testbed owner technical operators.

1.2.5 Enabler Owner

It is the owner of one of the 5G security enablers delivered by the project. They are testbed members and they are involved on some specific procedures dealing with the security enabler management. Respective Enabler Owners are defined in D3.2 [3].

1.2.6 Test plan Editor

It is a user that contributes to the editing of the test plan for the project security enabler validation.

1.2.7 Test plan Executor

It is a user that participates to the execution of the test plan and the collection of the results.

2 Security Testbed architecture

2.1 5G Security testbed requirements

Since the kick-off of the 5G-ENSURE project, the testbed requirements have been collected through five majors steps described here after:

step 1 – During D2.1 [8] investigation phase, all contributing partners were requested to contribute to the hosting criteria evaluation survey, in order to identify the Release R1 subset of enablers and collect the first set of requirements for designing the testbed to meet 5G-ENSURE needs. As discussed above, due to the time plan for the architecture (D2.4), the uses-cases defined in D2.1 [9] so far as also served as one of the main sources to capture architecturally related requirements.

step 2 – During D3.1 [2] investigation phase, the integration framework needs for Release R1 and Release R2 were collected, and the testbed requirements identified during step 1 were improved.

step 3 – During Plenary meeting in Kista (15th to 17th March 2016), the testbed team proposed to rely on OpenAirInterface™ [10] as integration framework which was agreed. During the same meeting, specific 5G network elements requirements for enabler's deployment were identified.

step 4 – Collection of implicit testbed needs identified during the development of D3.2 [3].

step 5 - Final validation of enablers testbed requirements during the Plenary meeting in Oxford (7th to 9th of June 2016).

The collected requirements have driven the design phase of the proposed testbed as described in the following chapter. The result of all this work related to the enablers hosting requirements collection can be found on Annex B.

The testbed includes capabilities of service function chaining since Release 1 (based on Release 1 enabler subcomponents). The capability of service function chaining (SFC) will be naturally integrated in Release 2 as depicted in **Figure 1**.

2.2 Testbed core architecture

The 5G-ENSURE project proposes to define an end-to-end testbed architecture. The Release 1 of the testbed architecture is based on enablers requirements defined in D3.2 [3] and aggregated in chapter 2.1.

Service function chaining will be naturally integrated in Release 2 as depicted in **Figure 1**.

In order to provide an operational environment, b<>com will provide its Unifier Gateway. The Unifier Gateway is a prototype providing a scalable distributable Converged Access Control and Core (for Wi-Fi and LTE-x RATs), based on SDN and NFV technologies for new deployments and with the additional capability to control legacy equipment (Wi-Fi APs, switches…).

Figure 1 shows the proposed architecture, where Network Functions such MME, HSS, etc are virtualized and run in different Virtual Machines as Virtualized Network Functions (VNFs) [11].

It may be noted that the final 5G architecture (defined by 3GPP) may not use exactly the same functional decomposition as 4G/EPC. For example, the 4G MME *could* be split into several functional blocks. For the scope of 5G-ENSURE, this has been deemed not critical. 5G-ENSURE focuses on security related enablers and it is clear that there will be authentication function in 5G, similar to the one of the current 4G MME.

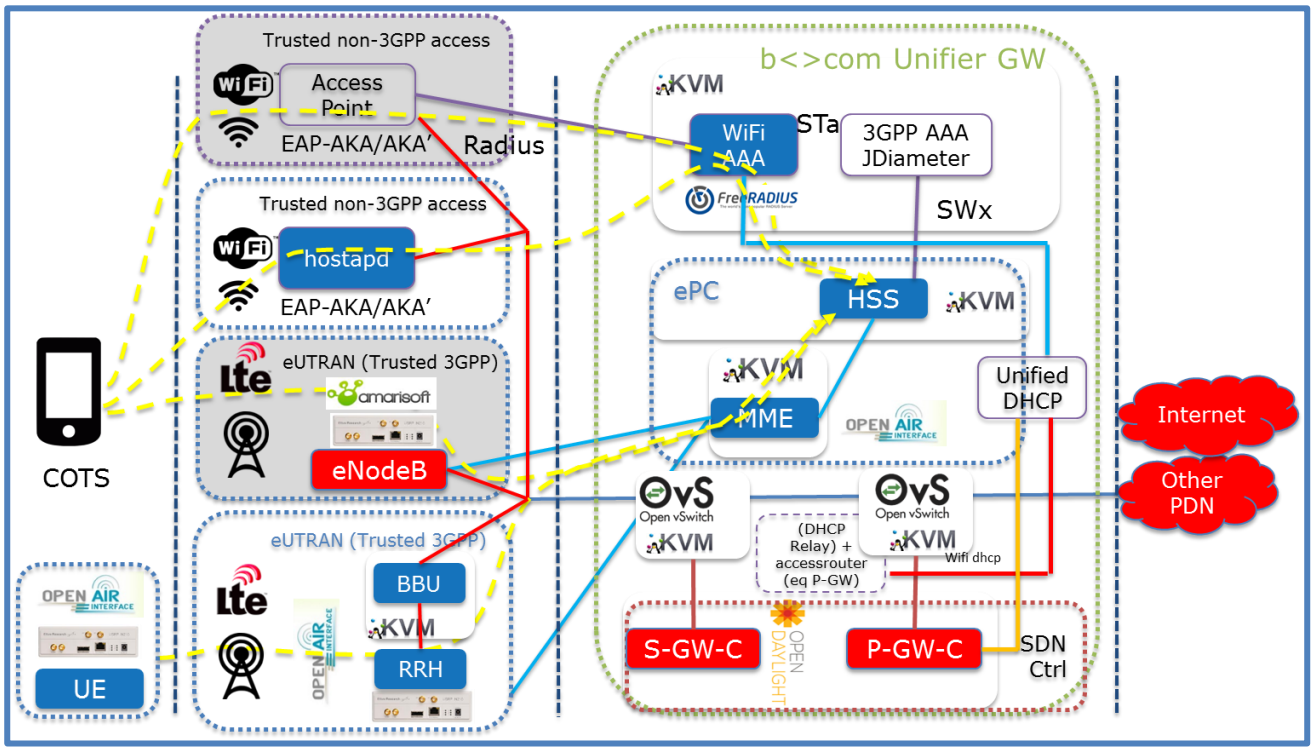

Figure 1: Unifier Gateway End to End architecture

From the previous schema, it is possible to identify that there are several radio access possibilities. It is currently possible to use COTS (Commercial off the shelf) UEs, eNodeBs and Wi-Fi access-points as black boxes in legacy 4G operation mode. However, from the project requirements perspective, as identified in

section 2.1, a major point of interest for such a solution is the ability to modify Network Functions and integrate security enablers in the framework.

Notice: Within the scope Release 1, each whitebox component modified by a partner is considered as an enabler component. This includes, for instance, Wi-Fi AP, AAA, DHCP, SDN controller, …

The Unifier Gateway architecture, as depicted on **Figure 1**, is a full end-to-end chain (RAN + ePC) based on Open Air Interface [10] which is an Open Source Framework and will be available for Release 2. Enablers which are not based on OAI framework, will have to comply with whitebox testbed approach.

Regarding the User Plane management, the P-GW and S-GW functions will be available for R2 based on OpenDayLight controller and the OpenVSwitch virtual switch. These components are also Open Source and would allow integrating 5G security based software in which it could also be possible to integrate the security enablers coming from the 5G-ENSURE project.

Figure 2 illustrates a detailed architecture of VTT's test site integrated into the overall testbed architecture of the project. The test site is currently largely based on LTE technology but 5G services and features will be developed upon that basis. Radio access network is based on Nokia's macrocell and small cell eNodeBs. OpenEPC [12] can be used to run core network functionality. The EPC runs on OpenStack/KVM virtualisation platform. All main EPC functionalities are included and multiple mobile devices (User Equipment, UE) can be connected to the network over real radio interfaces. In addition to real eNodeBs connected to the EPC, OpenEPC comprises eNodeB and UE simulators. In addition to E-UTRAN radio access, WLAN networks can be connected in a trusted and an un-trusted manner to the EPC. Internet of Things (IoT) radio network supporting multiple radio access technologies is integrated as part of the test network. Note that current 5G-ENSURE enablers are not dependent on layers below PDCP or NAS.

Because each of the EPC functionalities runs on a virtual machine, the test site enables deploying multiple instances of the EPC, where different functionalities and features regarding studied security enhancements can be connected between different test sites of the overall testbed architecture. Note that the purpose of deploying several EPC instances is to provide a flexible testing environment. It is *not* to be understood as means for validating the 5G network slicing concept, though for future use, it may serve also such purposes.

The EPC is connected to a service core network with a large variety of different services such as virtualisation platforms, Content Delivery Network (CDN) test network and IoT cloud system. A VPN tunnel router allows connecting the environment to other tests sites of the 5G-ENSURE project. VPN router resides in the service core network and it is connected over a 10 Gbps connection towards the Internet.

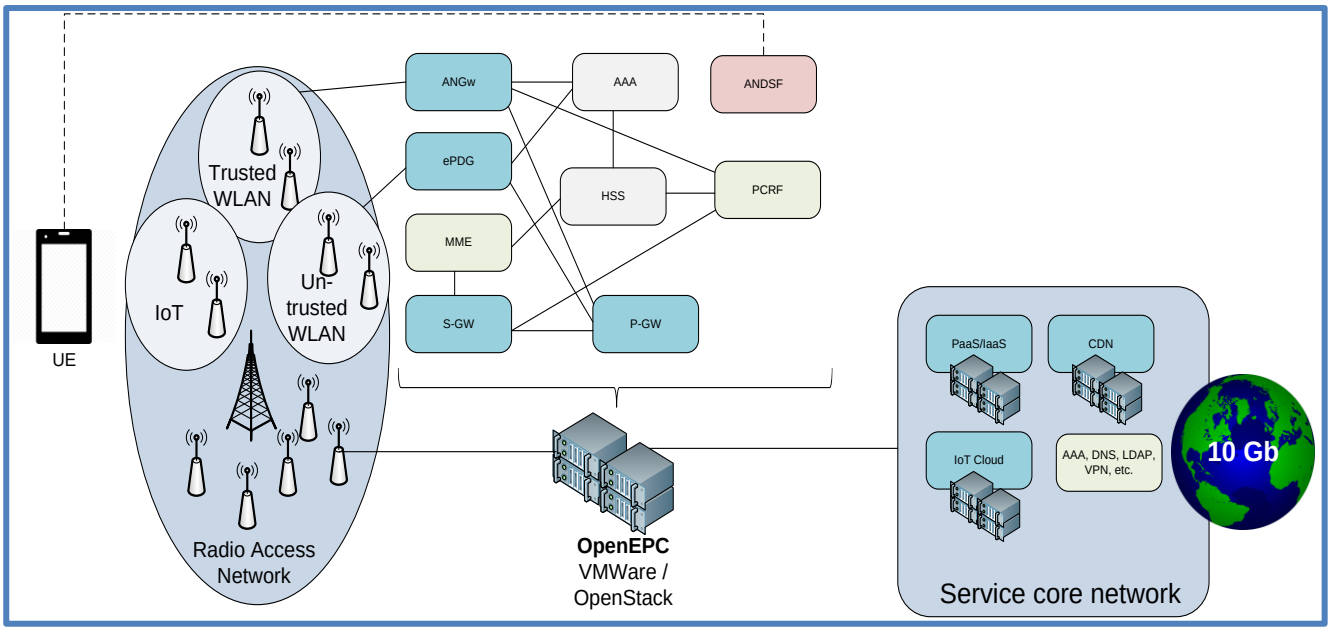

Figure 2: VTT's test site based on OpenEPC

With this two testbed nodes (b<>com and VTT), 5G-ENSURE has the ability:

- to test different implementations (RAN, ePC, etc),
- to provide multi-operator environment,
- to allow the Enabler Owners to evaluate the efficiency of their enablers on multi-domain environment.

2.3 Testbed detailed enabler's roadmap

The following roadmap is based on information collected inside D3.1 [2] and D3.2 [3] documents and the different contributions of Enabler Owners since the beginning of the project (see section 2.1).

As input, we consider the list of enablers available for the release R1 of 5G-ENSURE project:

Table 1: 5G-ENSURE Technical Roadmap for R1 (source D3.2 [3])

Category	5G-ENSURE security enablers	Features planned for 1st sw release (R1)
AAA	*Basic AAA enabler*	*A pre-study is required in the time frame of R1, however, due to lack of resources, an implementation is not feasible for the same release. Prototyping can potentially be done for R2.*
	Internet of things (IoT)	Group authentication
	Fine-grained Authorization	Basic Authorization in Satellite systems
		Basic distributed authorization Enforcement for RCDs
	Federative authentication and identification enabler	*none*
Privacy	Privacy Enhanced Identity Protection	Encryption of Long Term Identifiers **(**IMSI public-key based encryption**)**
	End-to-end encryption	*none*

Category	5G-ENSURE security enablers	Features planned for 1st sw release (R1)
	Device identifier(s) privacy	Enhanced privacy for network attachment protocols
	SIM-based anonymization	*none*
	Privacy policy analysis	*none*
Trust	Trust Builder[1]	5G Asset model
		5G Threat knowledgebase v1
	Trust Metric Enabler	Trust metric based network domain security policy management
	VNF Certification	VNF Trustworthiness Evaluation
Security Monitoring	System Security State Repository	Deployment model ontology
	Security Monitor for 5G Micro-Segments	Complex Event Processing Framework for Security Monitoring and Inferencing
	Satellite Network Monitoring	Pseudo real-time monitoring
		Threat detection
	Generic Collector Interface	Log and Event Processing
	Proactive Security Analysis and Remediation	5G specific vulnerability schema
		5G specific vulnerability schema implementation
		PulSAR interface with Generic Collector
		first study of a scenario based threat management
Network management & virtualization isolation	*Anti-Fingerprinting*	*Controller-Switch-Interaction Imitator* *This enabler has already been developed and evaluated. See the technical report publicly available at http://arxiv.org/abs/1512.06585, which has been accepted for publication at the IEEE Transactions on Forensics and Information Security). Its open specification is not included in this deliverable, since no further evaluation of its features is planned in R1.*
	Access Control Mechanisms	Southbound Reference Monitor
	Component-Interaction Audits	Basic OpenFlow Compliance Checker
	Bootstrapping Trust	Integrity Attestation of Virtual Network Components
	Micro Segmentation	Dynamic Arrangement of Micro-Segments

[1] The features planned for this enabler in R1 have changed. Please see the status section of the Trust Builder specification for more information in D3.2 [3].

Hereafter are reminded the deliverables and milestones originated from the 5G_ENSURE DoW and that will drive the work on testbed:

Deliverables:

- **D3.3 and D3.7 5G-PPP security enablers sw release:** Components delivered for integration in testbed (M11, M22),
- **D3.4.and D3.8 5G-PPP security enablers documentation**: Installation and Administration Guides; User and Programmers Guides; Unit Testing Plan (M11,M22),
- **D4.1 5G security testbed architecture**: Description of the testbed including nodes and their interconnection as well as the list of network elements (M6 delayed to M8),
- **D4.2 and D4.3 Test plan**: Description of how to evaluate the selected security enablers (M12, M18).
- **D4.4 Evaluation of the security enablers**: Results of the testbed runs (M24),
- **D4.5 Testbed extension and operation plan including business model** (M24).

Milestones and expected result:

- **MS4.1** Testbed up and running (M9),
- **MS4.2** The first set of enablers implemented with evaluation plans (M14),
- **MS4.3** Enablers evaluated and roadmap for the 5G security testbed (M24).

The testbed strategy of roadmap integration is based on 'easiest first' criteria. It means we will integrate the Release R1 enablers in 2 phases:

- The first phase (R1) is composed with a subset of enablers which are relatively easy to integrate based on the enabler hosting requirements (See Annex B). During this integration phase, we consider that we are able to integrate 2 enablers per week.
- The second phase (R1.1) is composed with the last part of enablers evaluated as more complex during the evaluation (see Annex B). During this integration phase, we expect that we are able to integrate one enabler per week.

The R1 integration phase will start in M12 (Oct'2016), with a duration of 4 weeks planned. The R1-phase will be based on the integration of following Release R1 enablers:

- Internet of things (IoT)
- Device identifier(s) privacy
- Fine-grained Authorization
- Satellite Network Monitoring
- Generic Collector Interface
- Component-Interaction Audits
- Bootstrapping Trust
- Access Control Mechanism

The R1.1 integration phase will start in M13, with a duration of 8 weeks planed (+ 2 extra weeks to cover the end of the year celebrations). The R1.1-integration-phase will be based on the integration of following Release R1 enablers:

- Micro Segmentation

- Security Monitor for 5G Micro-Segments
- Proactive Security Analysis and Remediation
- Privacy Enhanced Identity Protection
- Trust Builder2
- Trust Metric Enabler
- VNF Certification
- System Security State Repository

Note 1: The integration of an enabler could start as soon as it is packaged in terms of software and corresponding documentation are available inside D3.3 "5G-PPP security enablers sw release (v1.0)" and D3.4 "5G-PPP security enablers documentation (v1.0)". Prior to the enabler delivery, it is the Enabler Owner's responsibility to pass and succeed the **sanity checks** (unitary tests defined in D3.3 for each enabler). **Enabler is considered as integrated** on the testbed when the Enabler Owner has successfully passed the sanity checks with the enabler instance running on the testbed.

Note 2: The test plan of an enabler could start as soon as it is integrated inside the testbed, and tests are described inside D4.2 "Test plan" (Provided that it has no dependency with other enablers not already integrated).

The integration for all Release R1 enablers is estimated to be around three and a half months. This estimation is performed without any optimization and parallelization of integration tasks. Based on this roadmap, the integration phase of Release R1 enablers should be done by M16.

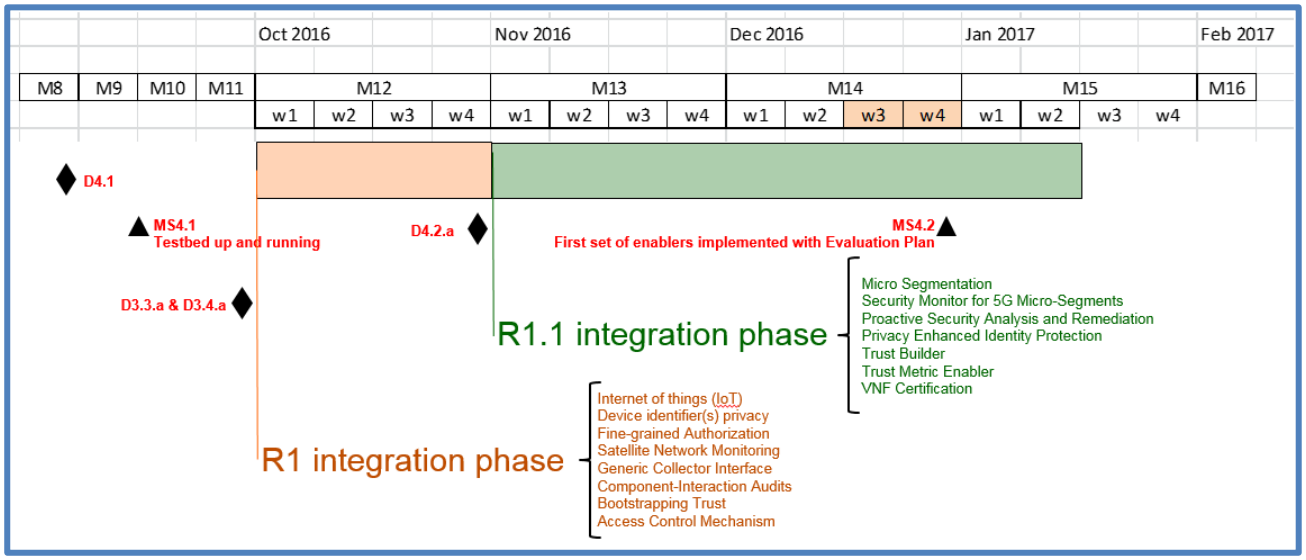

Figure 3: Testbed integration roadmap

[2] The features planned for this enabler in R1 have changed. Please see the status section of the Trust Builder specification for more information in D3.2.

3 Testbed hosting framework b<>secure

3.1 Overview

This section describes the tools available on the framework on which the testbed will be instantiated in b<>com premises. **B<>secure** is a multi-tenancy environment isolated from corporate resources. It is dedicated to Network & Security R&D domain for the instantiation of operational platforms. B<>secure could be seen as a framework capable of instantiating for each tenant an independent 5G-capable infrastructure. For each tenant it should be possible to apply the NFV/SDN [11] principles as defined by the ETSI [13], and the slicing and micro-segmentation security concepts [9] [14] that will be delivered by the corresponding security enablers.

The framework is provided with orchestration tools allowing for the instantiation and management of the configuration of the instances running in the environment. It will provide a 5G-ENSURE security enablers' catalogue service and a test management tool.

The framework is connected to the corporate RAN (either Wi-Fi or LTE advanced) allowing for the booking of radio resources during test sessions. It is also connected to corporate tools that provide basic features like AAA and network services. Finally, it has access to permanent storage for test logs and session outputs created during the period of the project.

Figure 4 illustrates the high level architecture diagram:

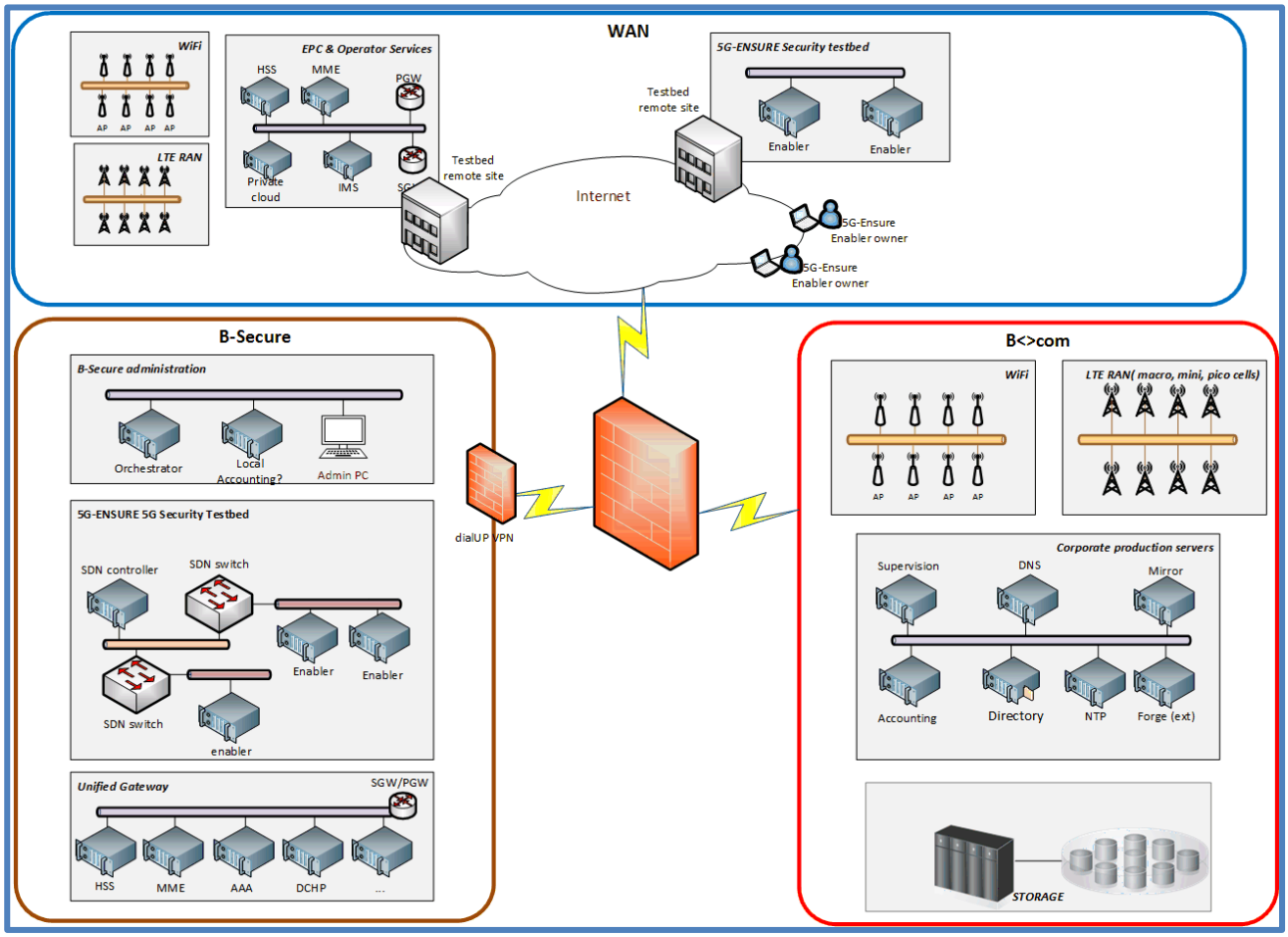

Figure 4: b<>secure HLD architecture

A tenant on the framework can be accessed remotely or interconnected with remote sites. This topic is described in the next chapter.

The next sections will cover a high level description of the framework building blocks.

3.2 Orchestration

The goal of the framework is to be able to easily handle the deployment of components and to ease up-scaling. The instantiation and configuration management tools apply to both virtual and bare-metal hosts. This allows handling of complex scenarios like telco infrastructures.

An important topic to highlight is the bare-metal host management within the framework: the hosts must be dedicated to the given tenant, even when they are running as hypervisor (for virtual machine hosting). This means that a given bare-metal host can only be assigned to one tenant at a given time. The reasons for do so are the following:

- ***Access to the Hypervisor***: It is foreseen that a security enabler's owner may need to access the hypervisor host in order to install enabler(s), capture traffic or collect logs. This is required in order to configure components like the hypervisor switch, monitor virtualization related information and perform traces.
- ***Resource competition***: Prevent competition on hypervisor resources from several tenant instances.
- ***Tenant physical isolation***: There are some threats linked to the virtualization. When running VMs from different tenants on the same hypervisor it's potentially possible to perform attacks from one tenant to another. Note that the proposed isolation is with regard to the experimentation framework, in order to allow hosting different testbeds within the same facilities. This isolation should not be assimilated to the one that should provide the slicing within 5G networks.

Some of the threats mentioned above can be the target of the implementation of some of the security enablers from the network management cluster within 5G-ENSURE. The environment will be adapted to allow performing the required tests in the perimeter of the 5G-ENSURE tenant.

The rest of the section will cover the security enablers' orchestration only. The deployment of hypervisors, network equipment and any other component that is not part of the tenant itself, is considered out of the scope of this document.

3.2.1 Enabler deployment

The orchestrator has been designed to perform the installation from scratch. This ensures that the right drivers will be loaded to satisfy the requirements on each host whether it is physical or virtual.

The main deployment steps are the following:

- Identify the host used to instantiate the enabler
- Define the operating system to install
- Provide a basic network configuration with the management network IP
- Deploy the basic Operating System installation, configure the management network and enable SSH.

Once the deployment of the base system is done, the system customization will be performed by the configuration management agent.

3.2.2 Configuration management

One of the key advantages of using a configuration management tool is to centralize the actions that are to be performed platform-wide during its life-cycle like:

- Keep all components up to date.
- Reconfigure components. For instance, if a change is operated on the engineering rules, it can be applied in one click to all the components.

Within the scope of the 5G-ENSURE testbed, the configuration management will allow, once the base OS installation is done, to:

- Install the missing system packages,
- Setup of tenant's engineering rules,
- Customize the experimentation network,
- Deploy the enabler from the catalogue,
- Configure the enabler.

Installing the missing system packages allows completing the system installation to fulfil its role on the platform. This can be for instance, the installation of the virtualization environment for a hypervisor, OpenVSwitch, or the X Window System for a desktop. It will also allow installing the packages required to apply the engineering rules, like LDAP or rsyslog.

Setup of tenant's engineering rules, in the case of 5G-ENSURE, will perform the following tasks:

- the LDAP agent will be configured to request the corporate directory for system login,
- the time service will be configured and synchronized with the same time server to grant time consistency across the platform components,
- a NAS volume will be mounted on the system to store collected data from test runs,
- rsyslog will be configured to collect some information for accounting purposes.

The previous engineering rules comply with the Testbed owners' Corporate Security Policy. They may be enhanced by other rules specific to the project.

During host instantiation, only the management network is set up. It is thus required to **customize experimentation network** on the host with regard to the enabler's needs. This may include setting up an OVS configuration in some of the cases.

Deploying the enabler from the catalogue and configuring it using the configuration management tool is an optional stage, although it is highly recommended. The automatic deployment of a security enabler needs a configuration-template phase based on collected owner information, in particular regarding the security enabler's dependencies. In order to build the enabler configuration template, collaboration with the Enabler Owner (See Annex B for more details).

3.3 Virtualization

Special attention is required in order to grant the proper operation of the network functions when they are virtualized (NFV) [11]. This is because some of the VNFs deal with real-time processing of the user plane data. It is well known that virtualization is not the most suitable solution to handle real-time processing constraints [15].

The testbed focuses on a single hypervisor: Kernel Virtual Machine (KVM). The choice is based on the enabler's hosting requirements collected information available in Annex B. In order to grant the best performances for the VNFs that will be hosted on the framework it will implement the recommendations for the KVM hypervisor [15].

The following schema shows a high level design about how network connectivity is designed on the hypervisor:

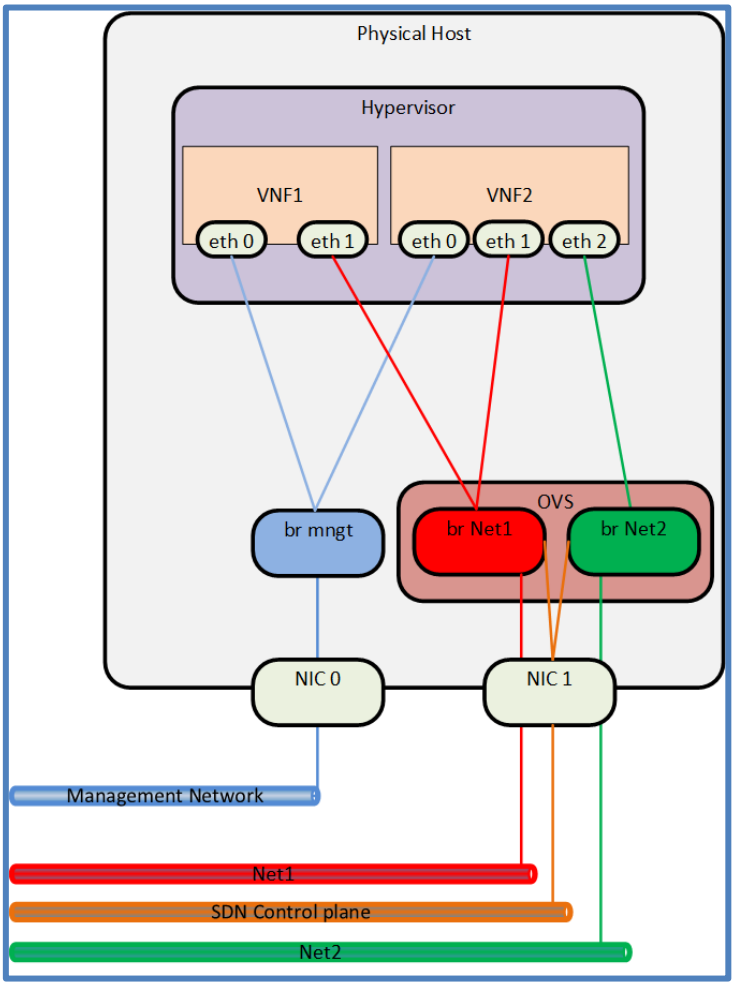

Figure 5: Hypervisor network design

As shown on the schema, there is an out-of-band management interface (depicted in blue in previous figure) providing resilient management using Linux bridges for all the components running on the hypervisor (the VNFs and the hypervisor itself).

Notice that some of the enablers provided by the 5G-ENSURE project, like those from monitoring and network management security clusters need to be deployed or require configuration on the bare-metal hosts. In order to cope with this requirement, the hypervisors are dedicated to the testbed and are accessible for testbed users.

The experimentation network will be handled by an OpenVSwitch instance running on the hypervisor. The OVS will be either SDN capable or behave as legacy L2/L3 switch depending on the requirements from the VNFs running on the hypervisor and the nature of the network that is connected to the VNF.

At the VM level, each connection to a given bridge on the experimentation OVS will be seen as a network interface by the VM.

3.4 Access control & Accounting

According to b<>com corporate directives, it is required to have personal access to the resources. In this way it is required to connect all provided resources (tenant instances, catalogue, storage, …) to the corporate directory in order to grant a fine grained access control across all the resources allocated by b<>com to the project. It is also required to keep track of the activities performed during the connection to the platform.

All partners of the 5G-ENSURE project will be able to create an account within the testbed which will allow them to access to resources available on the platform. The authentication will use a login and password scheme.

At the time of the writing, it is not yet identified whether it is required to have a more accurate access control to the resources (like per enabler or per security cluster). This can be implemented if required, but needs a clear definition from the partners about who can do what.

At the moment, three groups have been defined to be compliant with the default policies:

- **Manager**: In case some extra privileges are needed for some of the partners in order to manage the 5G-ENSURE testbed. For the moment, only b<>com staff will be able to perform platform administrative / management tasks.
- **Member**: By default all project partners are members of this group. It provides basic access control to connect to the VPN, access the catalogue …
- **Member with restricted access**: This group is meant for members that for some reason will not have full rights within the platform. This can be the case, for instance, of non-technical users that may access the testbed platform but won't be able to perform any change.

Within the perimeter of the enabler instantiation, the authentication and access control will be granted by an LDAP connection between the PAM (Pluggable Authentication Module) agent on the host and the corporate directory.

The accounting will be handled at two different levels:

- At corporate wide level, where wide accounting will allow to collect data from the network, the framework systems and the corporate services that are provided for the testbed.
- At testbed level, where accounting information from hosts deployed on the testbed will be collected.

Corporate accounting is out of the scope of this document. Concerning the testbed accounting, the following information will be collected at least:

- Login attempts: all login attempts, successful or not, will be collected.
- Connections: All successful connections will be logged (start time, end time).

3.5 Enabler catalogue

The testbed will provide a catalogue tool allowing for the creation of a repository with the 5G-ENSURE enablers that will run on the testbed. Such a tool allows having a centralized and homogenous environment to handle the enabler deployment on the testbed.

The entry point for the testbed team would be the release of enablers through software and accompanying documentation namely following deliverables to come:

- **D3.3 and D3.7 5G-PPP security enablers sw release:** Components delivered for integration in testbed (M11, M22).
- **D3.4 and D3.8 5G-PPP security enablers documentation**: Installation and Administration Guides; User and Programmers Guides; Unit Testing Plan (M11, M22).

It would be the basis for testbed team to make enablers integrated, tested and available through the catalogue in scope.

To minimize effort requested to get it done tools and formats to be used were investigated by the testbed team and defined as follow.

The tool that has been selected by the testbed is Artifactory [16] which provides the support for almost all formats of packaging in GNU/Linux environments. This tool covers all the packaging needs identified by the enabler's testbed hosting requirements (see Appendix B) criteria evaluation. Main Artifactory functionalities used for the project are:

- Create a 5G-ENSURE local repository.
- Host a local proxy for official repositories of different GNU/Linux distributions.

Within the scope of 5G-ENSURE, based on the Operating System chosen by the Enabler Owners for enabler delivery, and in order to minimize the complexity of the management of the enablers deployment and catalogue maintenance, the list of supported distributions in Release 1 are:

- Ubuntu packages using apt-get tool [17].

  Notice that Ubuntu distribution is a Debian fork. Both distributions share the same packaging system and the same repository structure

- CentOS packages using Yum tool. [18].

Other delivery methods associated with virtualized environments could also be envisioned to be supported:

- Vagrant images.
- Docker containers [19].

However, two important drawbacks for a specific delivery method for virtualized instances are:

- It will prevent instantiation on physical environments, which is one of the key features provided by b<>secure framework.
- It will get more complex to ensure that the engineering rules are properly applied to the instances and that installations are "clean".

The repository will be connected to the corporate LDAP server allowing users to authenticate using their testbed credentials. This will ensure that it is possible to apply the adequate access level restrictions to the

local repositories, either for adding new packages to the repository and to pull the packages from the instances deployed on the tenant.

The following schema illustrates the way the instances will be able to access the local catalogue in order to install the distribution packages and the project enablers.

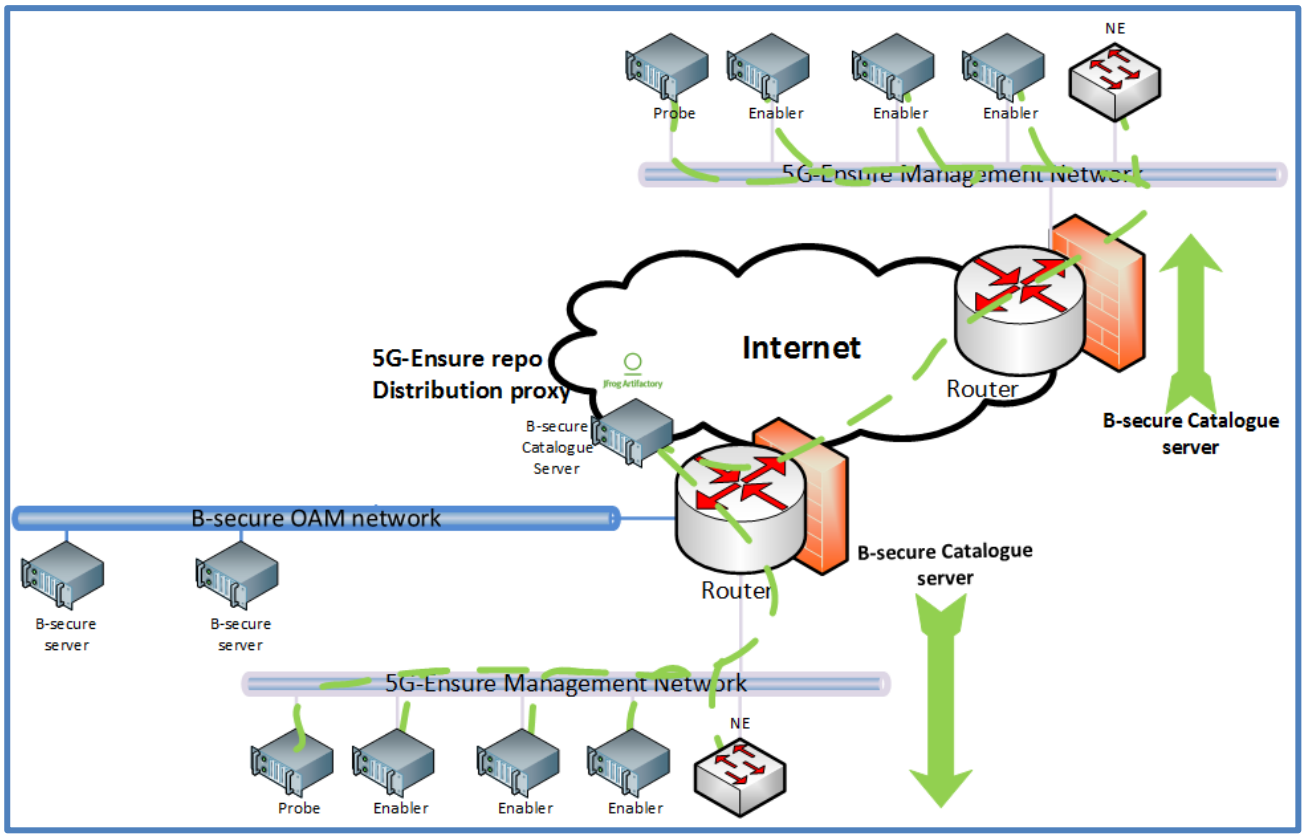

Figure 6: Access to the catalogue from the instances

The next schema shows how the catalogue gets populated.

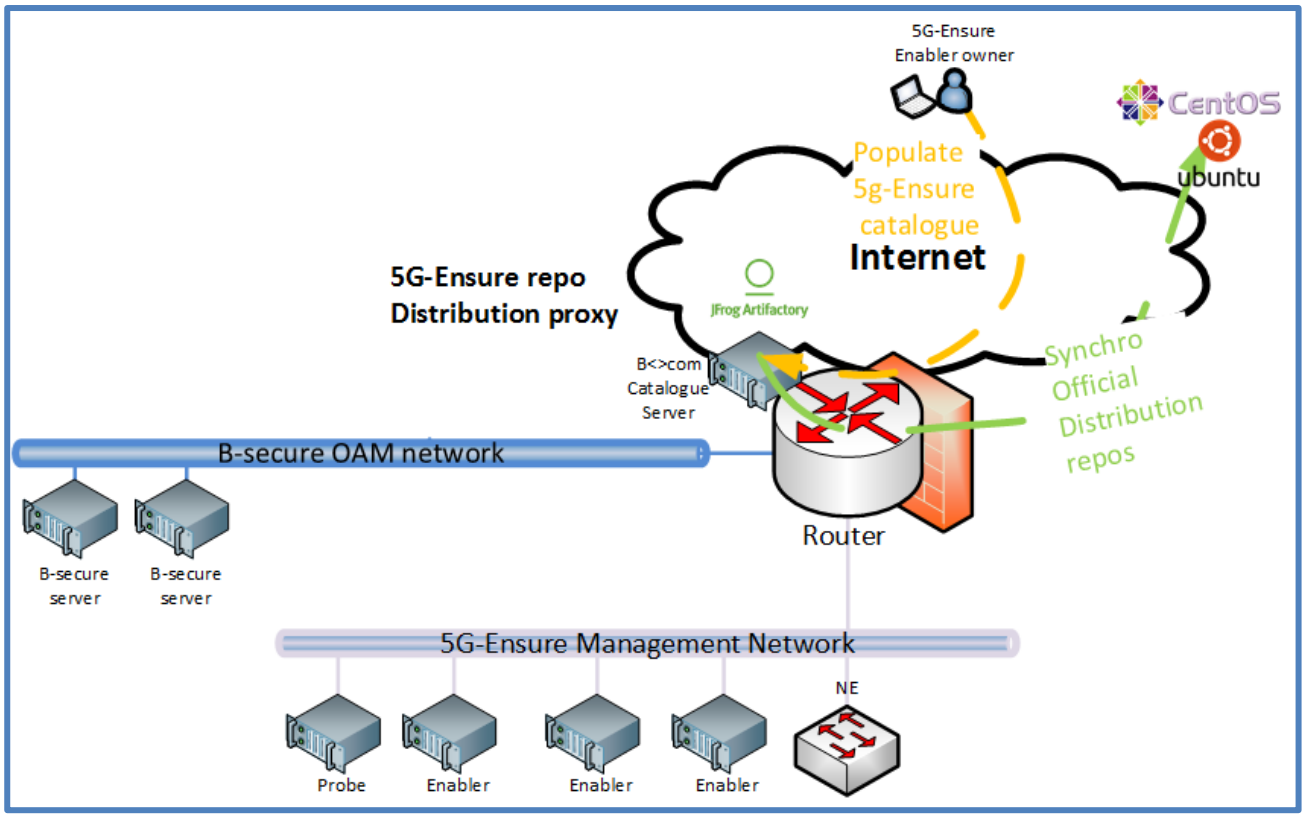

Figure 7: Catalogue population

Notice that it will be possible to populate the 5G-ENSURE catalogue without the dialup VPN.

The Operating System distribution local package repository (distribution mirror proxy) will be populated each time a new package is requested for installation on one of the instances. This provides a good compromise between installation performance (the package is only downloaded once) and storage optimization (only required packages are stored locally).

3.6 Test management

One of the main activities on which the testbed will be focused is the test plan runs. Thus a tool allowing to describe the test plan and to handle the collection of execution results will be available for Enablers' Owners and testers. Although the test plan falls under the scope of D4.2 and D4.3, the dataset management tool will be introduced in this deliverable.

At the time of the writing, b<>com has not yet deployed a dataset management tool for its corporate needs, but there is one that is compatible with the internal requirements and that can be deployed to cover the needs of 5G-ENSURE. This tool is **Testlink** [20], which is a free Open Source test management tool. It is a web based multi user tool.

Testlink provides the capacity to describe the test plan requirements and define the test cases for the needs of the test plan management in D4.2. For each it is possible to define the pre-requirements, the step by step test procedure, and the expected results. Once the test plan is available, it provides the capability to create the "test plan runs" (called builds on the tool) and allows collecting the results for each "run".

One important thing is that the tool is capable of exporting either the test plan and / or the "run" results in a variety of formats (word, excel, HTML…) which could be interesting for 5G-ENSURE as the test plan should be provided in D4.2 as a draft, an in D4.3 in its final version.

The Testlink user manual [21] and a screencast [22] will provide more detail about this tool.

The architecture retained for the deployment of this tool is depicted in **Figure 8**.

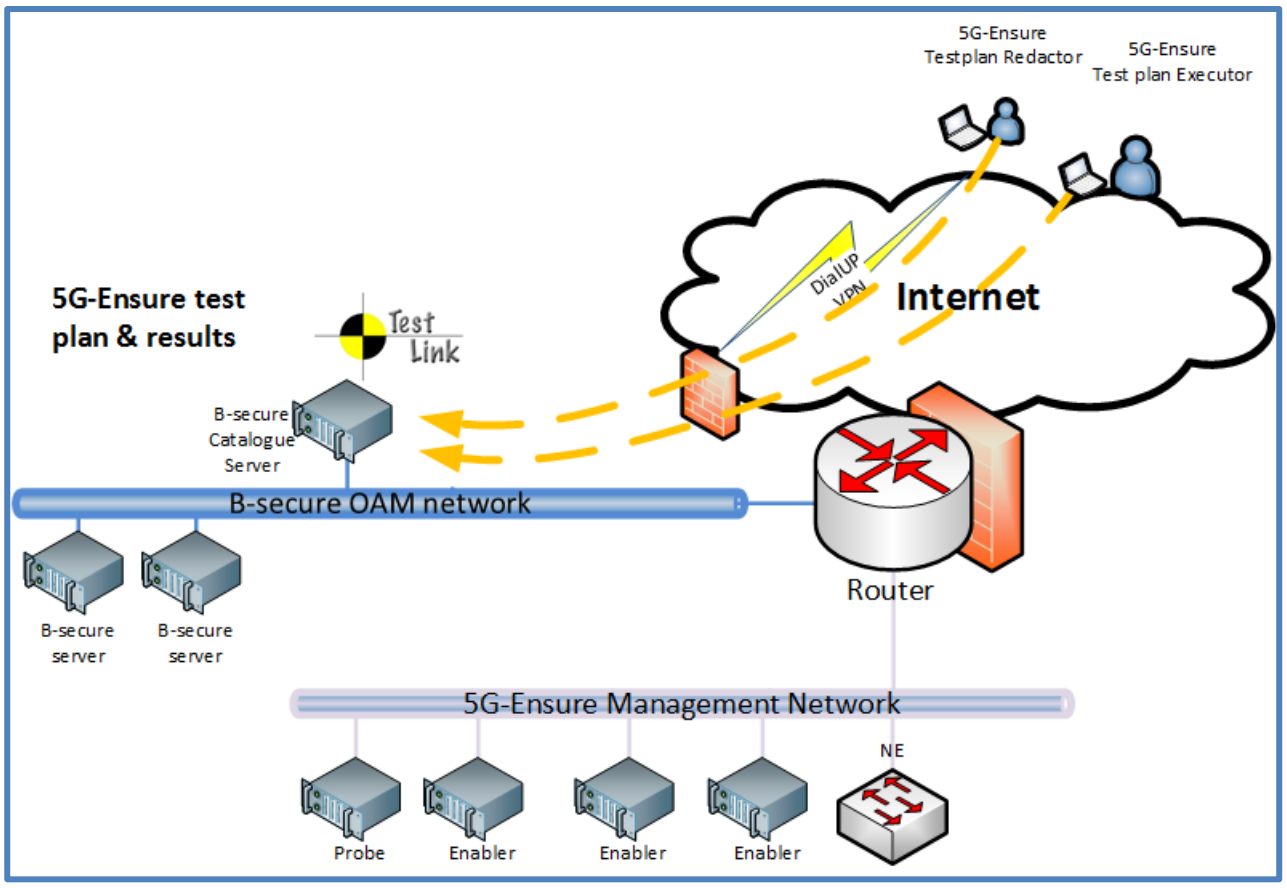

Figure 8: Test management architecture

As shown on the previous schema, the service will probably be hosted on the b<>secure framework environment, rather than being published as a corporate service. This means that it would only be possible to access the service through the dialup VPN.

3.7 Data archival

For the needs of 5G-ENSURE, b<>com will provide a persistent storage to record all the necessary data for the legal duration of the project. This storage will be accessible through the management network of the 5G-ENSURE tenant and will be visible as a NAS volume. In practice, it will most probably be mounted as NFS (Networked File System) on each enabler instance.

This storage will hold a variety of data linked to the testbed activities. It will typically host the test plan and the test results, but also all the instance profiles used for a given "test run" or even the traces and logs collected to analyse the results of a given test. The data to be collected and the way to do it will be defined and consigned on the test plan definition deliverables D4.2 and D4.3.

3.8 Network services

On any telco platform there are some basic network services that are important for its correct operation. The goal within the b<>secure framework is to deliver these services in an adequate manner to tenants without compromising the corporate servers. The strategy is to have a local framework server that provides the given services to all the framework tenants, which relays in the corporate servers. This way, it is possible to keep control locally on the way the services are delivered while relaying on production servers.

It will be possible to setup a dedicated server for a given tenant if the configuration requirements or the nature of the tests require it.

3.8.1 Network Time Service

Time synchronization across the testbed is a key factor in guaranteeing that the services will be delivered as expected. With regard to legacy environments, the virtualization of network functions may lead to unexpected behaviours regarding time synchronization. This service should be closely monitored during the early stages of the 5G deployments.

To get a proper network time reference for the tenants, the Network Time Server for the framework will run on a physical host and will be synchronized with the corporate Network Time Server(s).

All the bare-metal servers running on the tenant will be synchronized with the b<>secure Network Time Server. For virtual instances, the approach may differ depending on the hypervisor's virtualization environment. Several approaches are actually possible depending on the virtualization choice and the hardware capacities. In any case, it is important to monitor the time drift on the guest in order to prevent misbehaviours:

- Guest clock synchronization with host clock: it is rather easy to setup and, depending on the virtualization environment, provide accurate time with low drift. This is the case for KVM which provides kvm-clock tool allowing access to the constant TSC (Time Stamp Counter) of the host system. One important drawback of this solution is that, any error on the host clock will be mapped on the guest clocks.
- Using NTP service as the physical hosts: this solution provides a transparent network time management for all the hosts (virtual or physical) but may encounter drift issues when the guest is under CPU load (mainly).
- Another technique is to periodically run time synchronization command (i.e. though the CRON) to correct the drift. This approach does not look like to be resilient enough to deal with real-time services like those provided by telco platforms.

As a matter of fact, the two first approaches will be used simultaneously as advised in [23]:

- By default virtualization environment used is KVM running on the latest Ubuntu LTS release. This will ensure proper time synchronization between host and guest can be achieved via TSC.
- The NTP service will be configured to synchronize with the reference time server of the framework. It will correct the effects of clock skew as long as the clock runs no more than 0.05% faster or slower than the reference time source.

Below is the proposed NTP architecture:

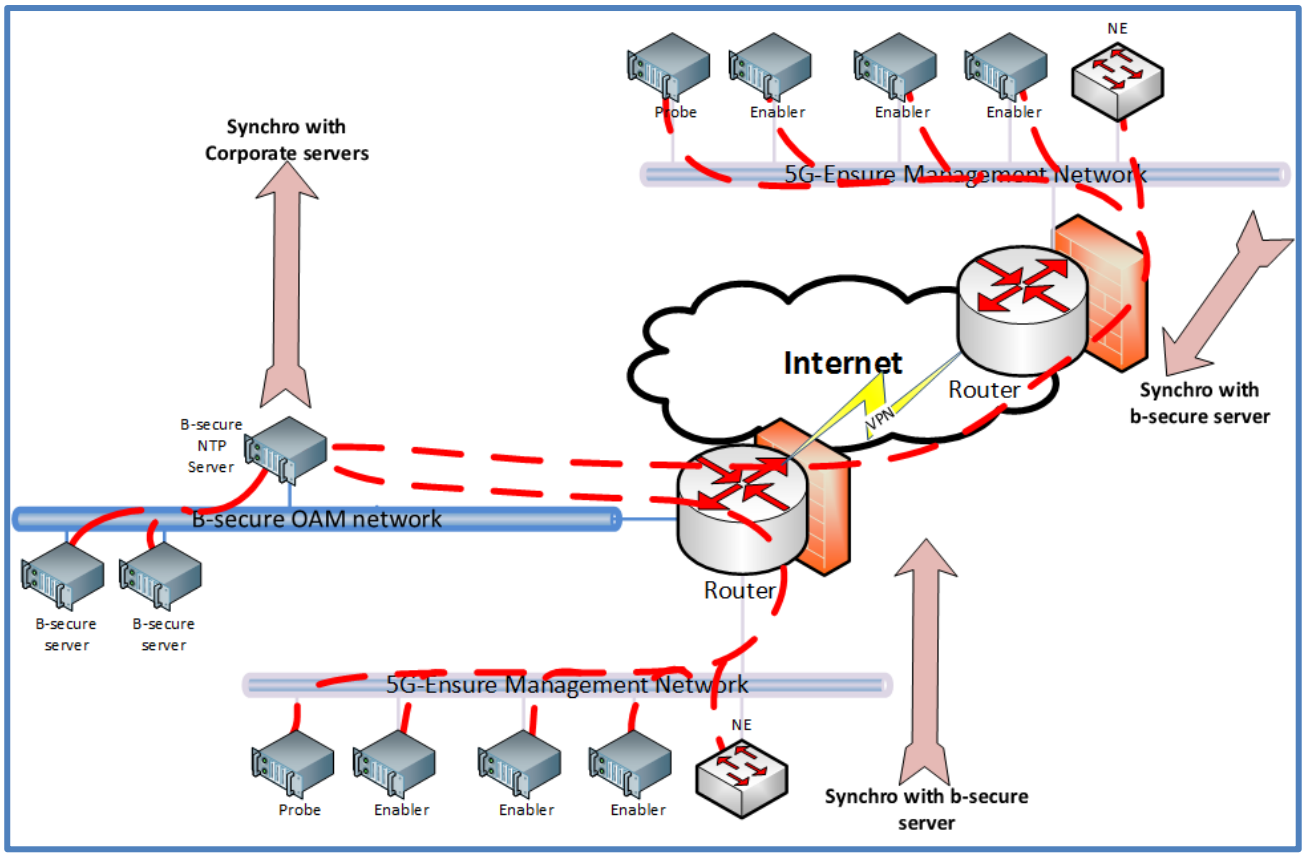

Figure 9: NTP service

3.8.2 Domain Name Service

DNS service is important as it allows dynamic discovery of network service allocation. In a context of an NFV approach this increases the flexibility and scalability of the deployments. However it becomes a mission critical service to ensure the proper operation of the platform.

The proposed architecture relies on the b<>com corporate DNS servers. As in the case of the NTP service, an intermediate server will be placed to support the b<>secure framework environment. In order to allow partners to modify DNS entries belonging to the testbed, it may be necessary to deploy a DNS server within the tenant instance to handle the local domain.

We foresee the need of several domains to address different needs:

- **b-secure.irt-b-com.org:** This domain is used by all the services provided within the b<>secure framework for all hosted projects.
- **lab.5g-ensure.eu:** This subdomain will be dedicated to all the testbed activities that will be exposed to the Internet. The subdomain will be operated by b<>com in delegation from the main project domain **5g-ensure.eu** which is operated by the 5G-ENSURE consortium.
- **lab.5g-ensure.local:** Internal domain used to operate the services within the tenant. All deployed tenant instances should be registered on this domain.

The following schemas illustrates two possible implementations of DNS hierarchy within the testbed. The first one is a centralized model in which all DNS provisioning for the framework and the tenants is handled by the b<>secure DNS server.

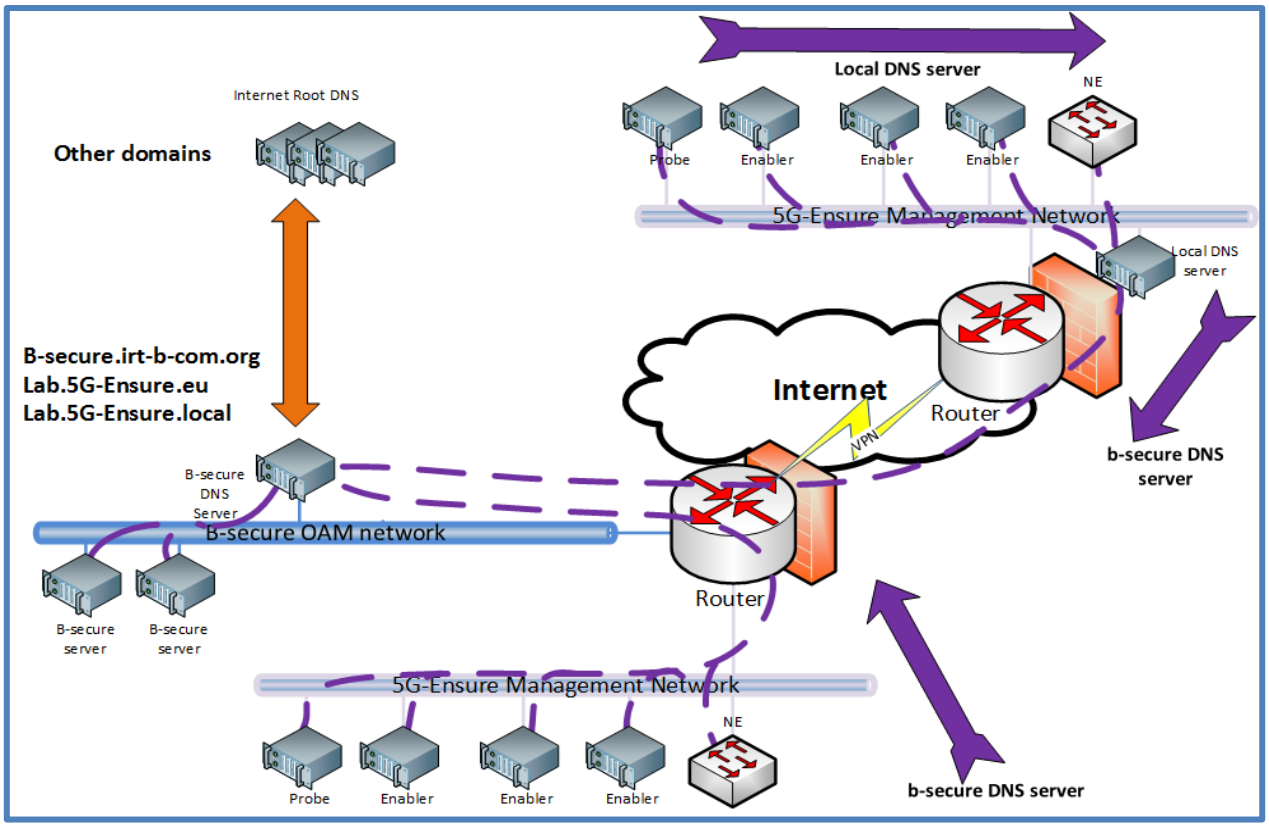

Figure 10: DNS centralized architecture

The main drawback is that, with this architecture, the 5G-ENSURE members would not be able to modify the records associated with the tenant local domain.

The other solution would be to instantiate a DNS server within the 5G-ENSURE tenant allowing tenant members to modify the records for local domains. The architecture is depicted in the following schema.

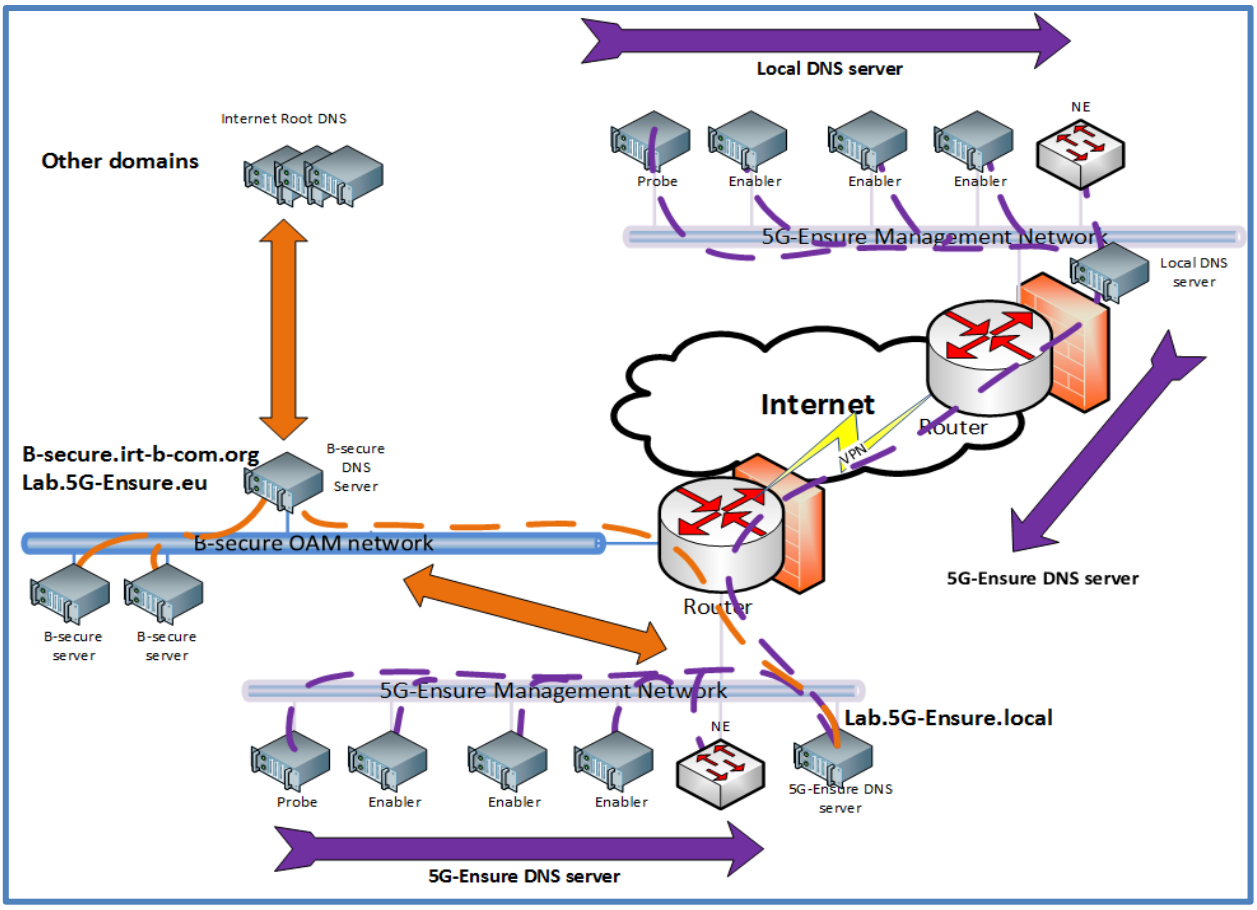

Figure 11: DNS decentralized architecture

The main drawback for this solution is that it would require implementing a synchronization method between the framework and the 5G-ENSURE DNS servers for the local domains as the orchestration tool actually adds the registers during the instantiation on the framework DNS server.

3.9 Helpdesk

The projects hosted on b<>secure framework should be able to request assistance in case something goes wrong on the framework or to perform new requests. As the goal of the framework is to host an operational environment (even in the case of testbeds), it is highly important to track a request/incident all along until its resolution. In order to do so, solutions like mailing lists are easy to setup but they are not efficient in terms of tracking.

The best approach is to use a ticket tracking tool. For the testbed operation purposes, the ticket tracking tool used will be GLPI, which provides an IT approach of issues management rather than a software development one. This should fit pretty well with the kind of issues handled within the testbed.

Users will be able to submit requests / incidents and they will be escalated according to their nature to the right team. Check section 5.2 to get the details of the process flow.

3.10 Hardware resources

The hardware resources dedicated to the project are expected to be dynamically allocated according to project needs between M9 and M24, the period during which the testbed should be operational. The allocation may start a bit earlier to allow the setup of the environment. The amount of resources allocated

at a given period will vary depending on the number of enablers available and the activities conducted on the testbed at the given time.

For instance, the complexity of enablers available in Release 2 is higher than the one expected in Release 1, thus it is expected to require more resources to host Release 2 enablers.

The renting period will be handled on a per month basis and the total budget should not exceed the budget provisioned for testbed hosting.

3.10.1 Computing nodes

There are two mainly two kinds of computing nodes that will be available to host the 5G-ENSURE enablers and other tenant instances:

SeaMicro server blades

The SeaMicro is a blade center equipped with 64 blades that targets HPC and Cloud based deployments. It is based on AMD technology and provides a diverse network connectivity to the external networks. It will host all the enablers and other building blocks that will run on the b<>com datacenter.

Each blade is composed of:

- 1 x AMD Opteron(tm) Processor 4365 EE with 8 x 2Ghz cores
- 64 GB DDR3 RAM
- 128 GB SAS Hard Disk (Extensible with SAN/NFS mounts)
- 8 x 1 Gbps NICs

Dell Precision Tower 5810

It is a powerful desktop server allowing to instantiate the enablers and other components that need to run in the labs. Most of the instances will run in the datacenter, but sometimes it is required to have the instance running on the lab for different reasons. In this case the instance will run on a lab desktop.

Each Dell Precision tower is composed of:

- 2 x Intel(R) Xeon(R) CPU E5-1620 v3 @ 3.50GHz with 4 x 3,5 GHz Cores
- 16 GB DDR3 RAM
- 1 TB SATA Hard Disk
- 5 x 1 Gbps NICs

Some other equipment like personal laptops may be used for the needs of demos.

3.10.2 Networking

The testbed resources are envisioned to be spread across multiple remote sites (at least b<>com & VTT at the moment). They could be also wide spread internally within each site (labs and datacenters for instance), so there must be a commitment on the allocated bandwidth between the entire execution environments.

The commitment for b<>com premises is:

- 1 Gbps internal connectivity between the labs and the datacenter. This connectivity is provided on top of the b<>com network infrastructure as a virtual wire.

- 100 Mbps average Internet connectivity within a 10 Gbps connectivity link. This access is provided by RENATER (the French National Research and Education Network), and should allow good interconnection with other NRENs from other countries through the GÉANT backbone.

As described in the introduction of the chapter, access will be leased on a monthly basis.

The following network equipments can be booked as tenant components:

- Alcatel-Lucent OS 6400: Standard of the shelf switch with layer 2 and layer 3 capabilities.
- Pica8 P-3295 SDN switch: OVS bare-metal implementation supporting OpenFlow v1.3 & v1.4.
- Some other SDN capable switches will be available when the testbed will be operational but they have not been chosen yet.

Special attention is accorded to the Radio Access Network (RAN). As shown in Figure 4, the HLD architecture diagram b<>com's RAN is a corporate resource shared across multiple projects and it is subject to resource planning, meaning that it may need to be booked depending on its nature.

The available resources are:

- WiFi AP supporting EAP-AKA authentication for mobile SIM based authentication.
- SDR based eNodeBs running on top of National Instruments USRP X310. They serve Macro (under construction) and Pico cells. They may run two different firmware images:
    - A 4G Legacy one based on Amarisoft on-the-shelf eNodeBs that is consider as a "blackbox".
    - A pre-5G modifiable based on OAI framework considered as a "whitebox".
- As for the User Entities, two different terminals are available:
    - COTS 4G smartphones that are seen as "blackboxes".
    - The OAI UE implementation running on top of an USRP (same solution as for eNodeB) considered a "whitebox".

3.10.3 Resilient storage

b<>com will provide storage to allow to preserve all testbed related data in a centralized and resilient way as committed in the DoW. As it is not possible at this stage of the project to get an order of magnitude of the amount of storage that will be needed, the proposal is to:

- Allocate 1 TB initially,
- Increment by 1 TB as storage needs increase.

The access to this centralized storage can be done in a variety of network protocols allowing to expose the storage space in the most suitable way.

The general idea is that dataset management related data will be stored on this shared space.

4 Testbed connectivity

The external testbed connectivity is a key aspect that has been identified since the beginning of the project. This includes aspects like Enabler Owner remote access, testbed site interconnection and Internet access for enabler instances running on the testbed.

As b<>com has been selected as the primary testbed site, it will act as the central point to allow the access to all resources allocated to the 5G-ENSURE testbed on all the sites.

4.1 Internet access

Internet access as described in this section is related to the capacity of the tenant instances to reach Internet services. All instances on the tenant may have Internet access (if requested by the Enabler Owner). For security reasons, this access will be limited by default to HTTP(s) service. If for some reasons there are special requirements for a given enabler or component, a tailored configuration may be implemented under request.

b<>secure framework has a public IPv4 pool allocated to. All b<>secure services exposed to the Internet use one of the IPs from this pool.

It is also possible to expose services provided by a given tenant (5G-ENSURE testbed in this case) through one or several of the public IPs allocated to b<>secure. However, at the time of this writing, it is not foreseen to provide any service within 5G-ENSURE testbed requiring to be reached from the Internet directly.

4.2 Partner's remote access

Project partners need to have secure access to the testbed in order to be able to deploy / configure their enablers, run testing activities and access to datasets.

A DialUP IPSec service will be hosted on the primary premise of the testbed and will grant secured access to the testbed resources.

Two methods will be provided to allow partners to establish a connection to the testbed based on the IPSec dialup VPN.

4.2.1 Client-less VPN connection

This method allows establishing the VPN tunnel from a web browser. It doesn't need to install a VPN client on the client side, but **it requires ability to execute Java applets**.

Once the user is logged in, it is possible to reach the enablers through their management interface using either embedded web tools or creating a tunnel that will allow usage of local tools. The preferred method will be to use the tunnels as they will allow partners to use their own tools and it is possible to upload/download files between the enabler instance and the local hosts.

One main drawback is that each management interface that needs to be accessible must be added manually to the VPN configuration.

4.2.2 With a VPN Client installation on the local computer

The VPN client is Cisco Any Connect [24] and it is available for Windows, Linux and MAC OS. The client is available for download on the VPN web portal once the user is logged in. The VPN connection will use HTTPS to encapsulate the IPSec tunnel, so most of the firewall configurations should allow the traffic.

One key advantage is that once the VPN connection is established all management networks can be accessed at once with any of the administration protocols (SSH, HTTPs,….) without any further configuration. Other advantage is that there is a direct connection between the client computer and the server, and thus, it is possible to upload/download data.

However, installing such client requires administrative privileges on the client computer which might be restricted for some of the partners.

4.3 Testbed site interconnection

In order to create an extensible testbed with all security enhancements studied and developed in the project, test sites from different partners are interconnected. Interconnections are carried out through VPN tunnels. Each VPN tunnel can be configured by taking the capabilities of partners' VPN routers into account. However, parameters listed in Table 2 are used as preferred configuration.

Table 2: VPN tunnel configuration

Parameter name	Parameter value
Internet Key Exchange (IKE)	*Authentication Method:* Preshare
	Coding Algorithm (encryption): AES 128
	Messages Digest Algorithm (authentication): SHA1
	DH group: 2
	Lifetime: 28800
	Nat-T : No
IPSec	*Protocol :* ESP
	Perfect Forward Secrecy (PFS) Yes
	Group : 2
	Coding Algorithm (encryption) : AES 128
	Messages Digest Algorithm (authentication) : SHA1
	Lifetime : 3600
	Replay detection : No
	Auto key keepalive: No
Authentication key	A secure authentication key shared by phone
IP network and address information	- Subnets of both sites - IP Gateway IP addresses of both sites

Quality of Service (QoS) measurements were carried out in order to validate that the performance of VPN tunnels is sufficient for the targeted use scenarios. An IPerf [25] traffic generator of version 3.1 is used to

measure the maximum throughput capacity and jitter behaviour of VPN tunnels with differently sized Maximum Transmission Units (MTUs) and packet sizes. TCP traffic will be used to measure throughput performance while UDP traffic is used to measure mainly jitter. The UDP measurements indicate the baseline jitter behaviour without seeking to maximize maximum throughput results. The measurements are performed in both directions. Instead of measuring one-way delays, Ping is used to measure Round-Trip Time (RTT). One-way delay measurements would need end-hosts' clocks to be synchronized accurately. With Network Time Protocol (NTP), a synchronisation accuracy of 10-20 ms could be achieved in optimal conditions. However, delay behaviour in fixed networks is typically symmetrical in both directions (assuming that the route is the same), which makes RTT measurements sufficient for the baseline performance measurements needed in this case. In order to get also statistically valid results about the performance, measurements are carried out during a 24 h period with an interval of 30 minutes. To automate the performance measurements, shell scripts, shown in Annex A, have been prepared to carry out the measurements. In addition, Perl scripts were prepared to parse the measurement results log files.

The performance evaluation of a VPN tunnel between b<>com and VTT has already been carried out. **Figure 12** shows the performance in relation to different MTU and TCP segment sizes. In measurements with different TCP segment sizes, the MTUs of both end hosts connected over the VPN tunnel were set to 1500 B. In measurements with different MTUs, the MTU size was changed on b<>com's side only. However, as the measurements were performed to both directions, either the sender or the receiver side had a changing MTU in these measurements, depending on the measurement direction. Overall, MTU or TCP segment sizes were not observed to affect the throughput performance significantly until the MTU or TCP segment size was decreased down to 100 B. The maximum TCP throughput capacity is about 97 Mb/s. However, in the direction from VTT to B<>Com, the performance stayed stable regardless of the MTU in the receiver side (B<>Com).

In the measurement results shown in **Figure 12**, all individual links between the end-hosts were 1Gb/s or higher. **Figure 13** shows results from same measurements, however, now with a link speed of 100Mb/s in one of the links between the measurement end-hosts. As can be seen, MTU and TCP segment size has now more drastic impact on overall TCP throughput. With the MTU of 100B, less than 20 Mb/s average throughput was measured from b<>com to VTT.

One of the key results from these measurements is that even though there is fragmentation of IP packets performed, which happens with MTU and TCP segment size of 1500 B due to VPN tunnel header overhead, its effect on performance is seen as negligible.

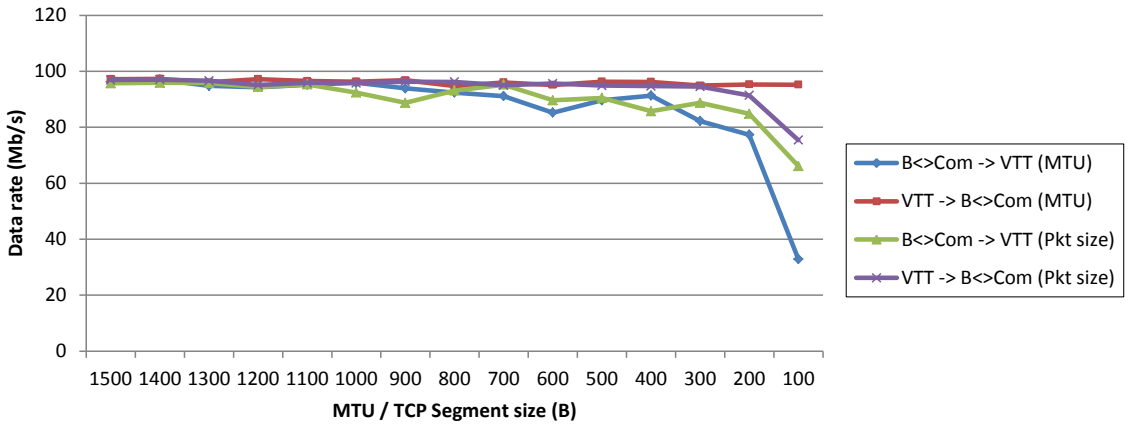

Figure 12: TCP throughput performance in relation to MTU and TCP segment sizes with ≥1Gb links

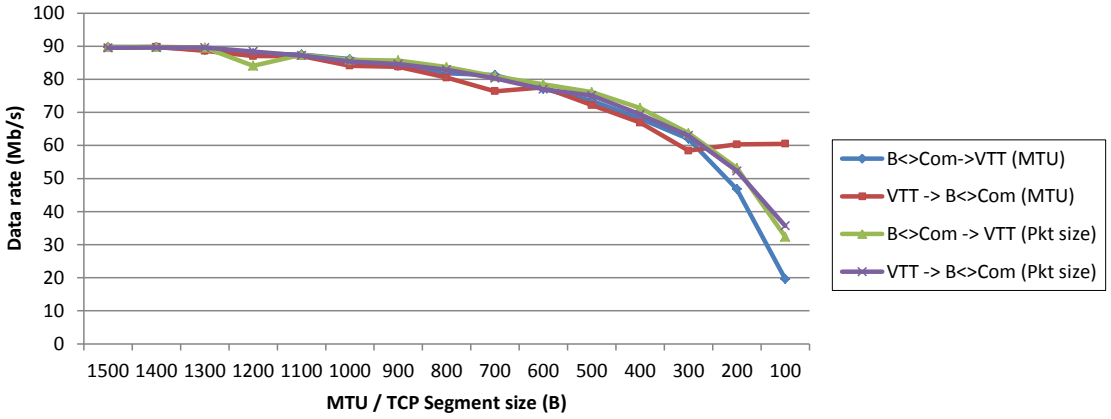

Figure 13: TCP throughput performance in relation to MTU and TCP segment sizes with a 100 Mb link

In order to see throughput performance variation between different times of a day, measurements were carried out for a 24 h period. Standard deviation in throughput results is only 4 Mb/s, which indicates that there is not much variation in results and throughput stays relatively stable.

Average RTTs between B<>Com and VTT were measured to be 65 ms, at maximum being not more than 72 ms. From a 24h measurement period, the standard deviation of RTT values is only 2.6 ms. Average jitters were measured to be 3.9 ms. At minimum, jitters of 0.6 ms were obtained while maximum jitter values exceed 10 ms. Standard deviation of the jitters was 2.7 ms.

Although measurement campaign for baseline QoS performance of VPN tunnels would not reveal any particular performance problems, the performance can change over time. Thus, having a real-time and passive QoS monitoring tool is recommended to be used when making tests over the tunnels. VTT's Qosmet tool [26] is a light-weight and software-based passive QoS measurement tool, which allows real-time monitoring and analysis of various QoS parameters. The term passive means in this context that the Qosmet tool does not generate the traffic to be measured but it measures QoS from traffic of active applications. Through its integrated Quality of Experience (QoE) algorithms, Qosmet is capable of doing also real-time pseudo-subjective QoE evaluation for particular applications. When tests that need controlled network environment are being conducted over the testbed, Qosmet can be used on each end of VPN tunnels to monitor the QoS of tunnels. This enables finding reasons whether possible inconsistencies in test results are caused by the tunnel(s) or the test setup/applications.

4.4 Interconnection with other 5G-PPP platforms

Interconnections with other 5G-PPP platforms could be enabled by the VPN tunnelling capabilities of the 5G-ENSURE testbed. It has to be decided which features will be interconnected with other 5G-PPP projects.

This usage of the testbed will be qualified within the D4.5 "Testbed extension and operation plan including business model".

5 Testbed operational procedures

This section covers the operation procedures that have been already identified at this stage of the project. Other procedures may appear during the project depending on the testbed requirements evolution.

5.1 Testbed user registration

All provided services (at least those provided by b<>com) implement personal access control. So, the first step to perform in order to gain access to the testbed resources is to create a testbed personal account. The procedure to obtain a testbed account is the following:

- Write a mail to the testbed mailing list and request a new account creation. Include the following personal information :
  - ***First name***
  - ***Last name***
  - ***Mail address***
  - ***Company name***
- The Testbed Support team will handle the request and, if accepted, will provide a login / password to access the resources.
  - By default, all registered testbed users will have the same access rights as members of the project.

  Notice: *The provided password is temporal. The first time the user will connect to the VPN, it will be requested to change it*.

Once the user has the user account, it will be possible to connect to all the available services with the account (VPN, dataset management, storage, virtual machines….). At the time of the writing, it is not foreseen to have different access levels within the testbed, but it will be possible to implement a per security cluster or per enabler access basis. This could be required to limit the access rights to enabler hosts or to the datasets of the different enablers.

It is also possible to have administrative or restricted access profiles. In order to implement this kind of access control inside the testbed, it would be necessary to have a clear request coming from the enabler Owners.

5.1.1 Password management

A password management tool is not yet available for external b<>com users. In case a user loses its password, it will be required to contact the Testbed Operator team in order to reset the password.

5.2 Testbed support

An operational platform requires a support service in order to operate properly. Usually the support is composed of several layers or stratums that range from general issues/request troubleshooting, to specialized troubleshooting. Based on the existing roles, the following layers could be adopted:

- Layer 1: Handled by the Testbed Support group which is in charge of receiving the initial request and either resolve the issue or address it to appropriate group on the upper layers.
- Layer 2: Requests addressed to this layer will be managed by the Testbed Operator when it deals with an issue / request about the testbed execution environment or by the Enabler Owner if it can't be solved and deals with the behaviour of enabler itself of one of the enablers running on the testbed. Nevertheless support would stay limited to the features provided by the enabler in the release considered (e.g. enabler R1 features).

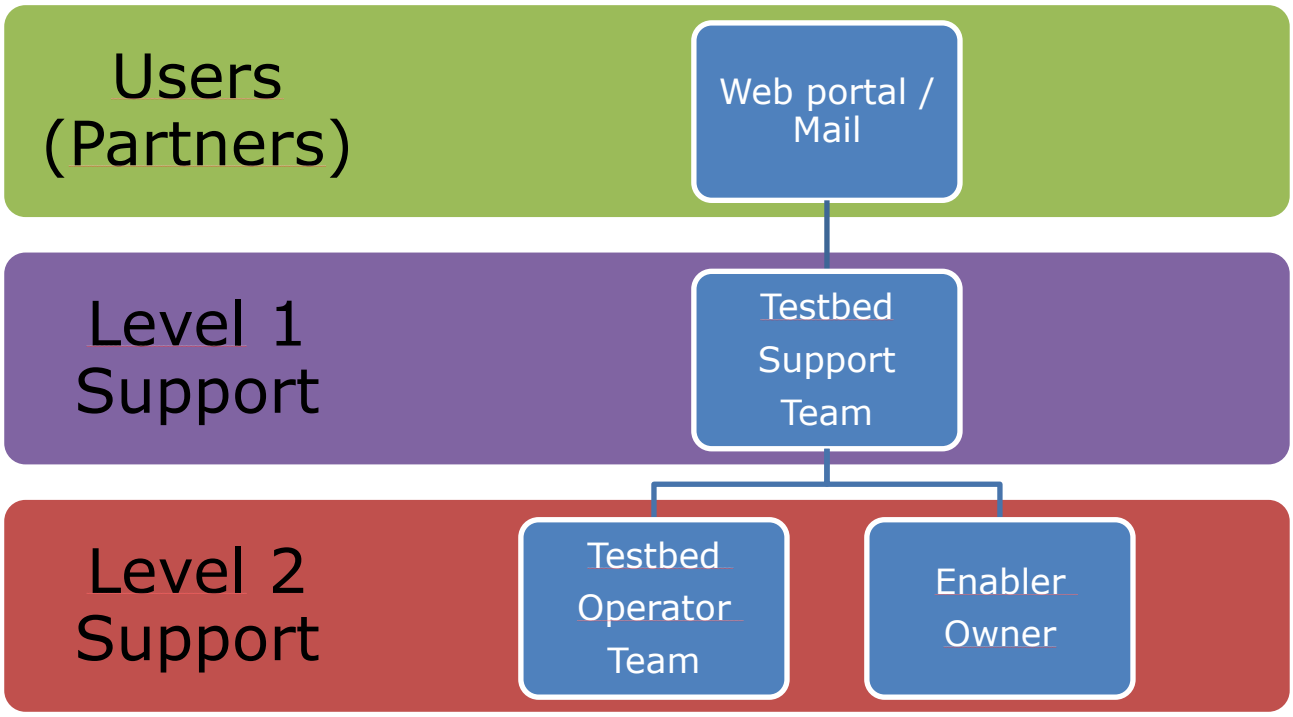

Figure 14: Testbed support organization

As defined in section 3.9 a helpdesk tool will be available to help with the issue management and tracking. A basic support process flow is required in order to grant the proper management on the request.

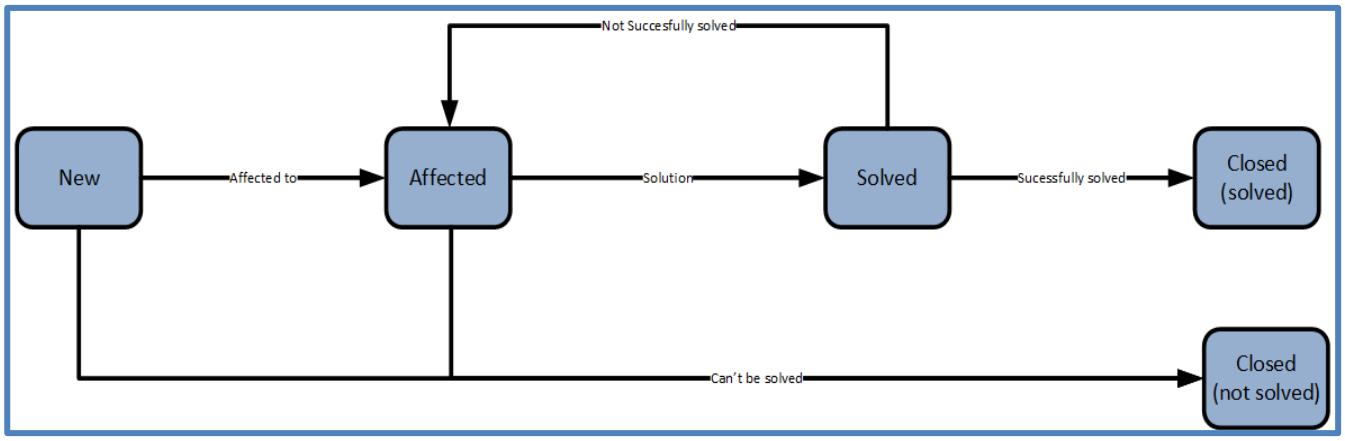

Figure 15: Support basic process flow

The previous figure provides the support basic process flow which can be described as follows:

- A testbed user opens a ticket for a new request / incident. The tickets is created and its status is **new**.
- The ticket will be evaluated by the Testbed Support team (layer 1 support) and affected to the appropriate team. The ticket status will become **affected**.
- Once a solution is identified and applied, the ticket will be **solved**.
- If the solution satisfies the testbed user's expectations, the ticket becomes **closed**. Other ways it will come back on status affected.

While the ticket is open, it is possible to:

- Request complementary information to the testbed user to better qualify the ticket.
- To close the ticket because there is not a feasible solution or because the demand is simply out of scope.

As any other tool proposed for the 5G-ENSURE testbed, the ticketing system requires personal testbed credentials to access the helpdesk tool.

All requests coming from testbed users should be performed via the helpdesk. Only external requests like account creation or new site interconnection are supposed to be addressed by mail. The only exception is the password reset request until a tool will be available for external users.

5.3 Remote access to the testbed

Once the user has been registered it is possible to access the testbed through the DialUP VPN. The first connection should be done through the VPN web portal.

Notice: ***Java Runtime Environment needs to be supported by the browser in order for the VPN portal to work properly.***

Here under are the steps to connect to the testbed:

- Before starting the connection, it is required to add the VPN portal URL to the trusted hosts on the Java configuration panel.
    - Open Java configuration tool and go to “Security” tab:

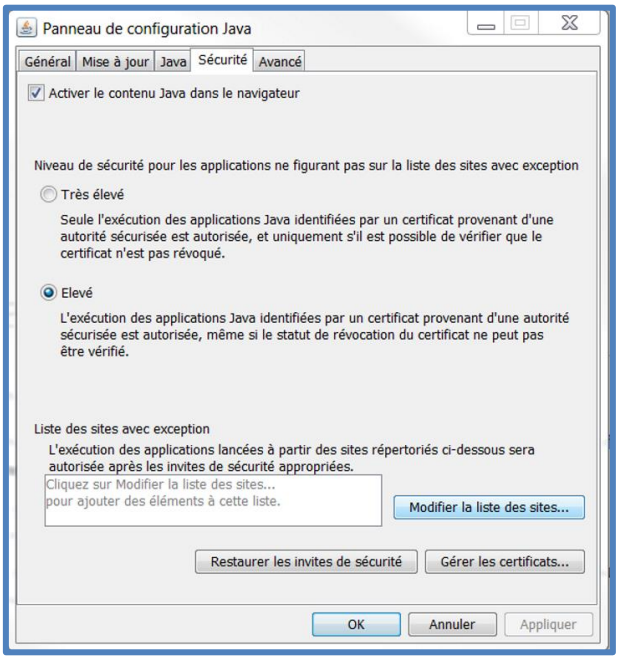

Figure 16: VPN web portal Java configuration

    - Add the VPN portal URL (**https://vpn.b-secure.irt-b-com.org**) to the trusted sites (requires admin grants):

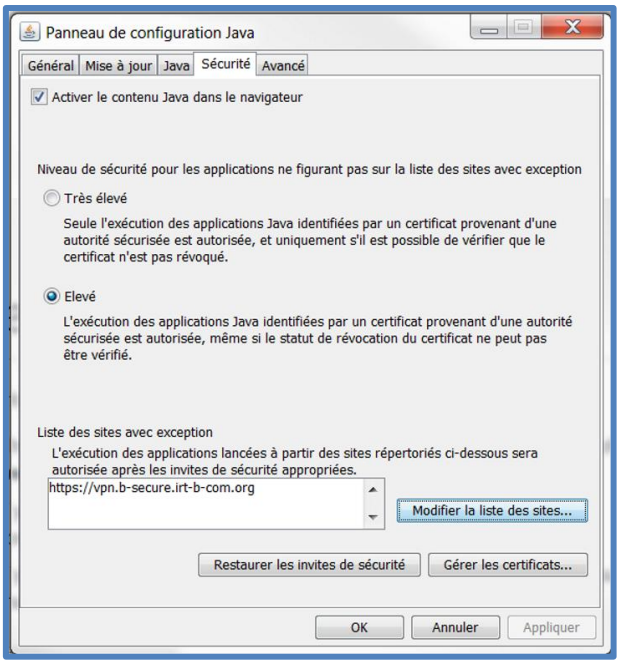

Figure 17: VPN web portal add trusted server

- Connect with the web browser to https://vpn.b-secure.irt-b-com.org. Something similar to the following should appear:

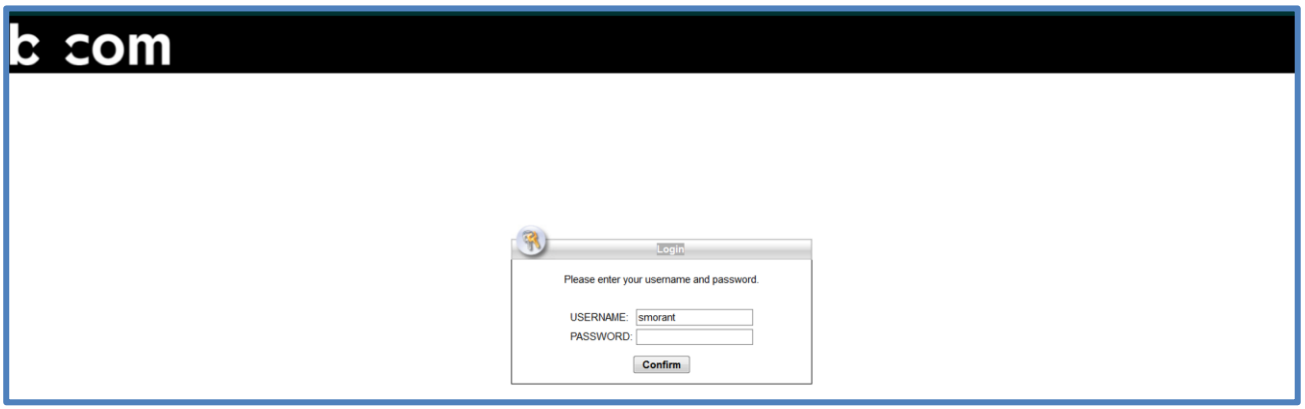

Figure 18: VPN web portal login window

- Type the user credentials and click on “confirm” button.
- The first time user logs in through the VPN, it will be requested to change its temporary password and choose a personal one.

Notice: ***it is requested to provide passwords meeting some minimum security requirements (ie length, mixing capital, low case letters and special characters***).

- Once the connection is established, a welcome banner will popup. Click on “Continue” button:

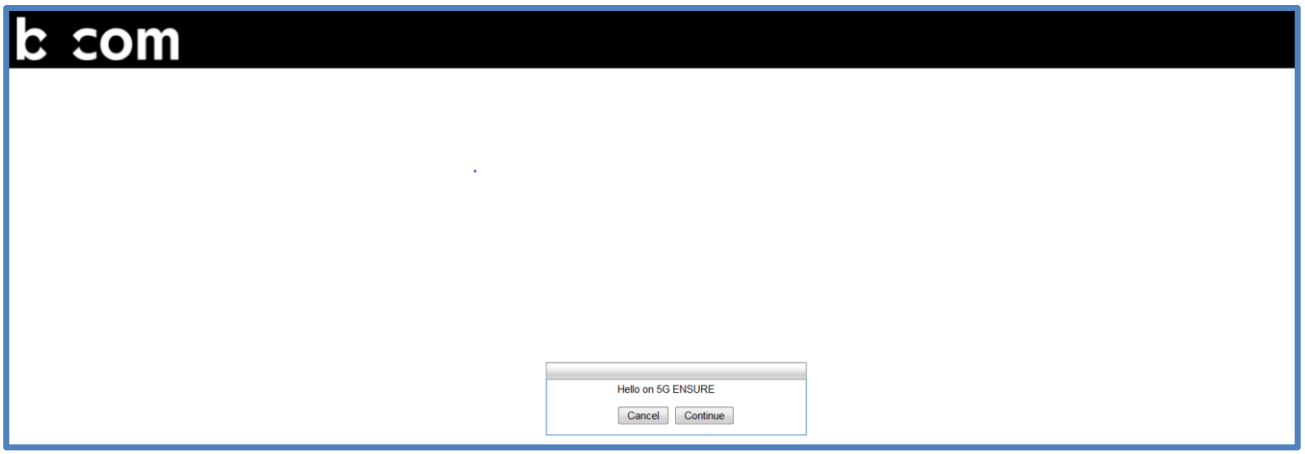

Figure 19: VPN web portal welcome banner

- By default the following menus will be proposed for 5G-ENSURE:

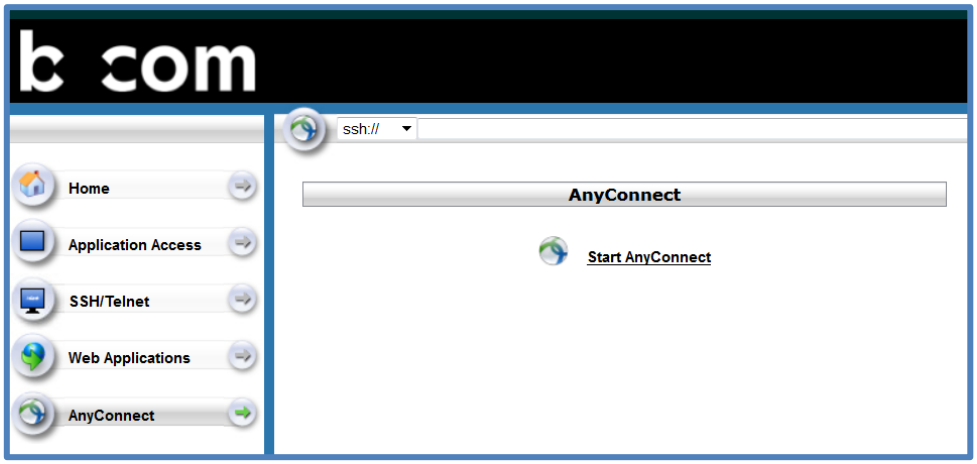

Figure 20: VPN web portal default page

  - Home: Will actually show the resume with all accessible connections.
  - Application access: Provides the access to enabler management interface by means of a per application tunnel.
  - SSH/Telnet: Allows to connect with SSH protocol to the different hosts present on the testbed platform, or at least hosted on b<>com premises.
  - Web Applications: Will allow to connect to the hosts present on the testbed by the means of HTTP(s) protocol.
  - AnyConnect: Allows to download the Cisco Any Connect client to local computer
- In order to test the compliance of the web browser access method with the local computer configuration, there is a bookmark allowing access a test virtual machine.
  - Go to SSH connections and click on “5G-ENSURE test server SSH”.

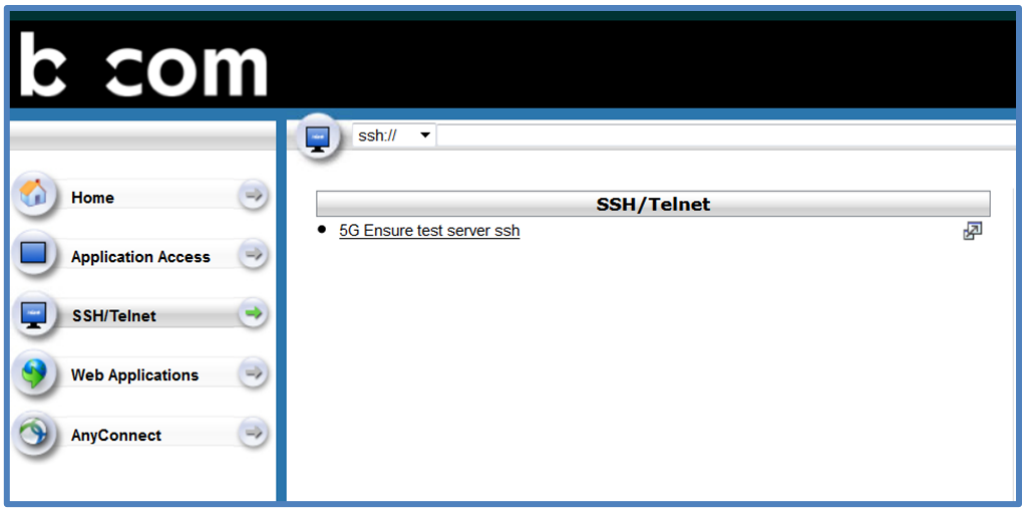

Figure 21: VPN web portal SSH

- If you obtain a message like the following one, it is pretty much possible JAVA is not properly installed or configured to run with the chosen browser:

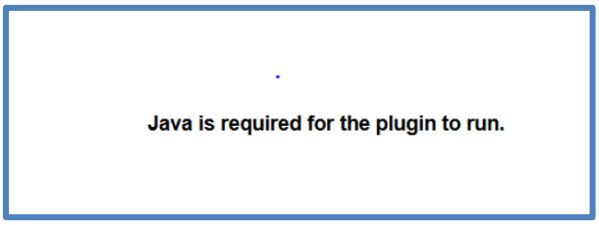

Figure 22: VPN web portal Java setup error

- Afterwards it is highly possible that the JAVA plugin requires activation:

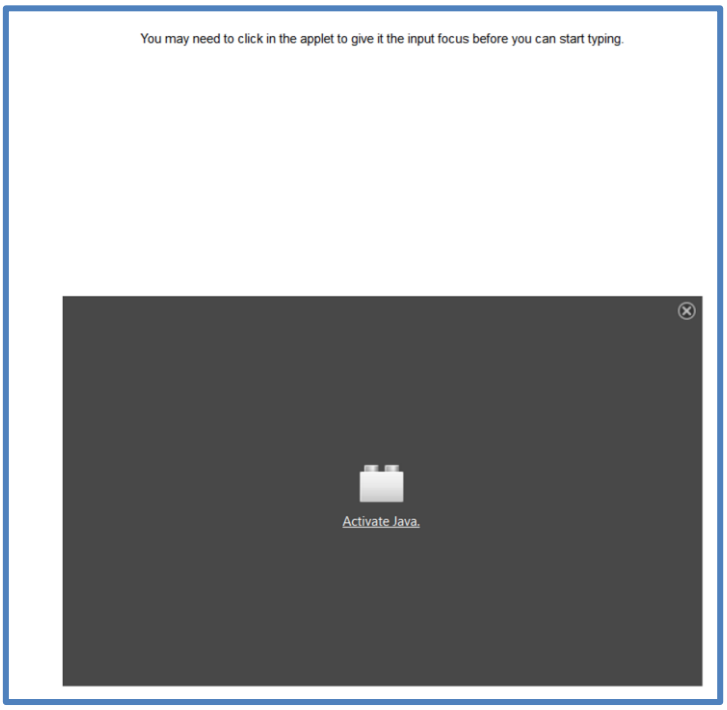

Figure 23: VPN web portal plugin activation

- Press continue when the following security alerts will pop-up, press “continue” or “execute” buttons:

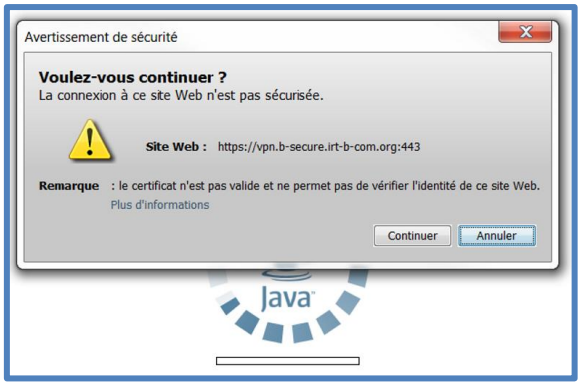

Figure 24: VPN web portal security alert

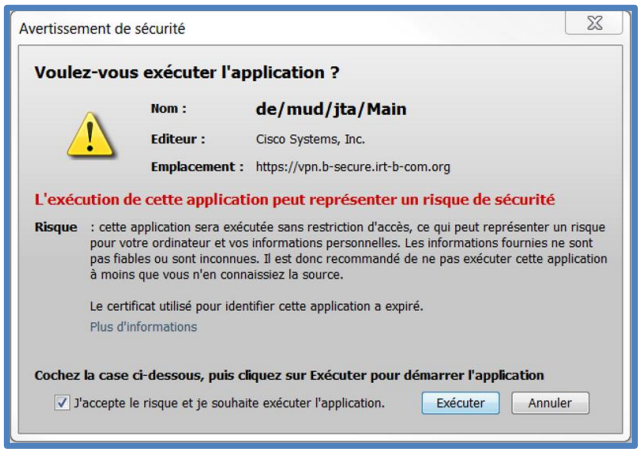

Figure 25: VPN web portal JAVA execution alert

- Once done a pop-up requesting the user credential will pop-up. Use the personal credentials to log in:

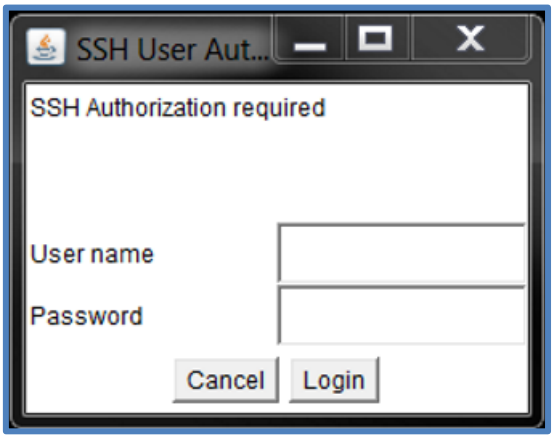

Figure 26: VPN web portal SSH login window

- If everything went right the terminal should be accessible through the web browser.

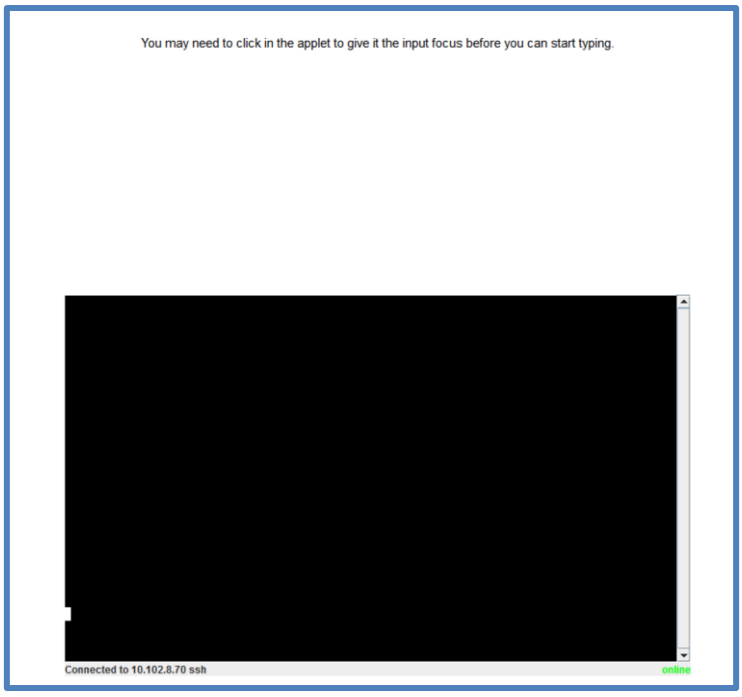

Figure 27: VPN web portal SSH console

The previous example can be applied to other management protocols like web browsers.

Another clientless access method is the application tunnel, called application access on the web portal:

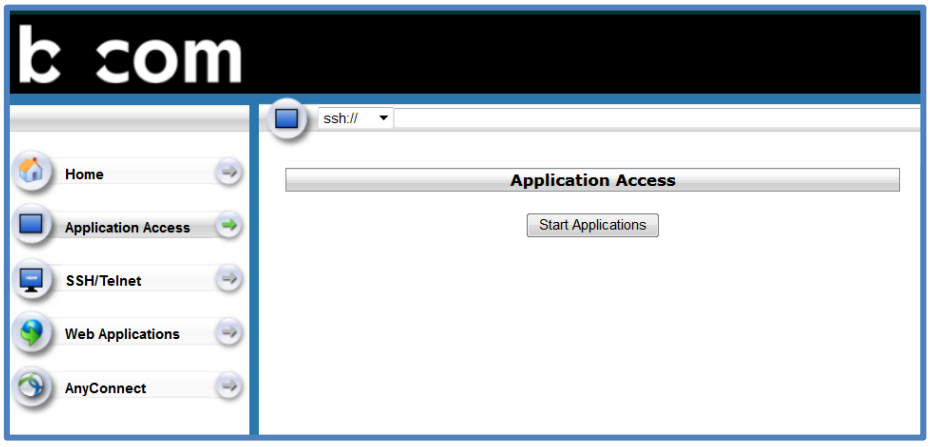

Figure 28: VPN web portal Application access

- When clicking on the start applications, the VPN server will create the VPN tunnels defined on the VPN server and they will be displayed on a popup:

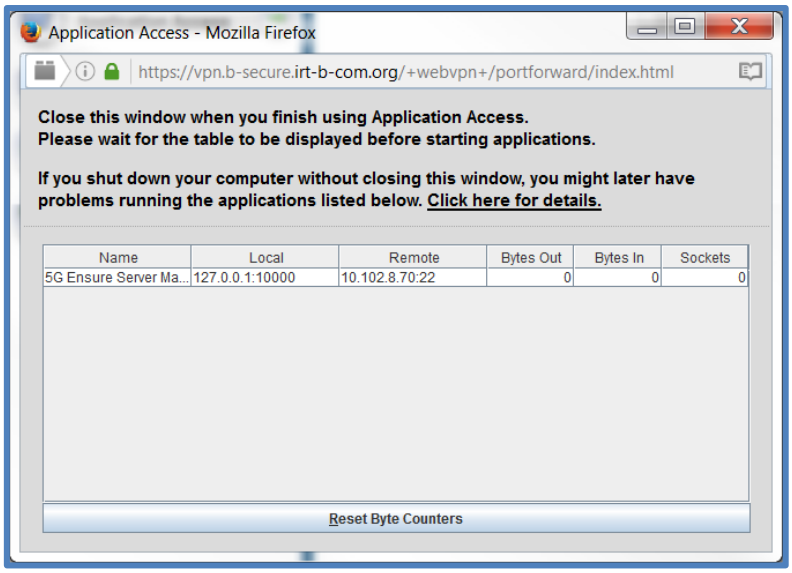

Figure 29: VPN web portal application tunnel status

- In the example, an SSH tunnel connection will be tunnelled through the 127.0.0.1:10000. In order to gain access to the enabler SSH console configure the local SSH client with the connection information:

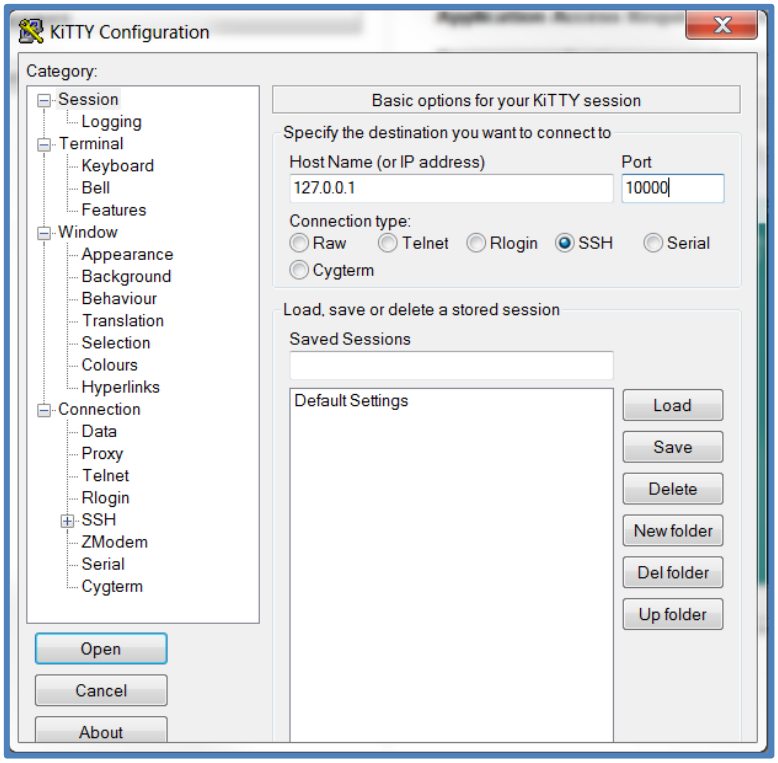

Figure 30: VPN web portal SSH remote access through application tunnel

- When clicking on the connect button, the remote terminal window should appear. Use personal testbed credentials to log in:

Figure 31: VPN web portal VPN SSH remote console

If only clientless VPN access method will be used, then this is all. On the other hand, if willing to use the Cisco AnyConnect VPN client to establish the VPN connections, follow the next steps.

- Go to the VPN web portal "AnyConnect" entry and click "Start AnyConnect":

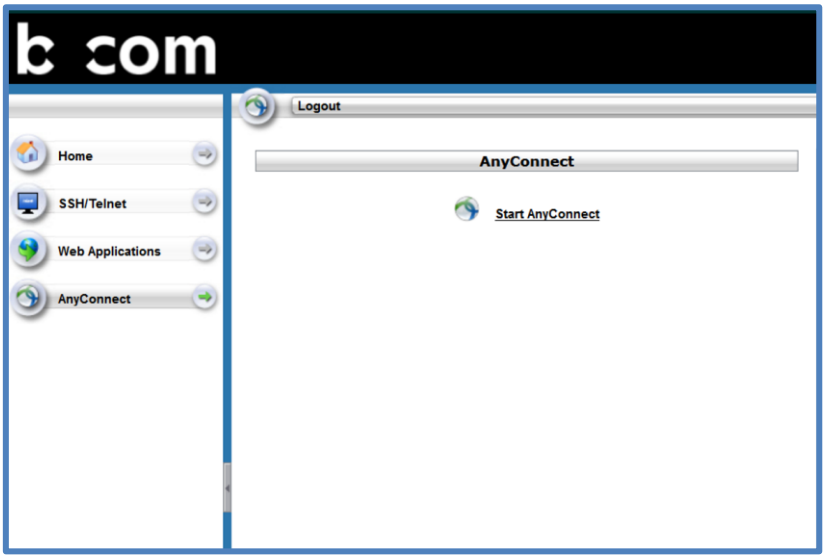

Figure 32: VPN web portal AnyConnect

- A security pop-up will request execution authorization for the JAVA applet. Press "Execute" button:

Figure 33: VPN web portal AnyConnect JAVA wizard execution alert

- The applet will attempt to download and install the Cisco AnyConnect client but it is pretty much possible that the action fail depending on the environment. If so, it will be proposed to manually download and install the client. Press on the proposed link ("Windows Desktop" in the example) and proceed with the client install:

  Notice: Admin rights are required on the local computer in order to successfully complete this step.

Figure 34: VPN web portal AnyConnect wizard

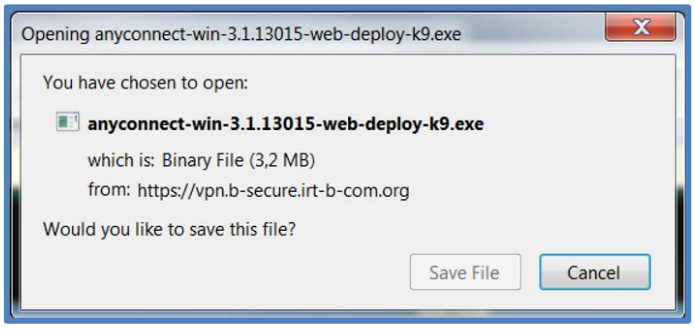

Figure 35: VPN web portal AnyConnect download popup

- Once the VPN client is installed, launch it and provide the VPN server URL. Then click on “Connect” button:

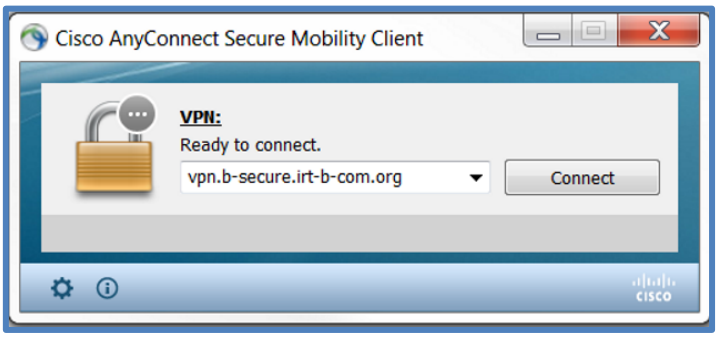

Figure 36: DialUP VPN server configuration

- Provide the testbed personal credentials when prompted:

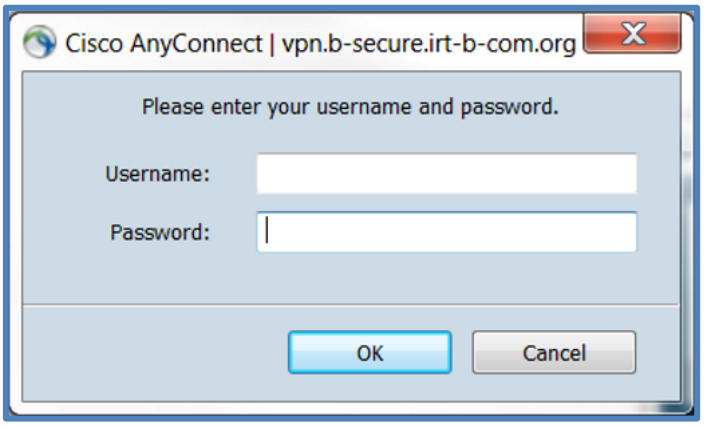

Figure 37: DialUP VPN login window

- If everything went right a pop-up like this will appear:

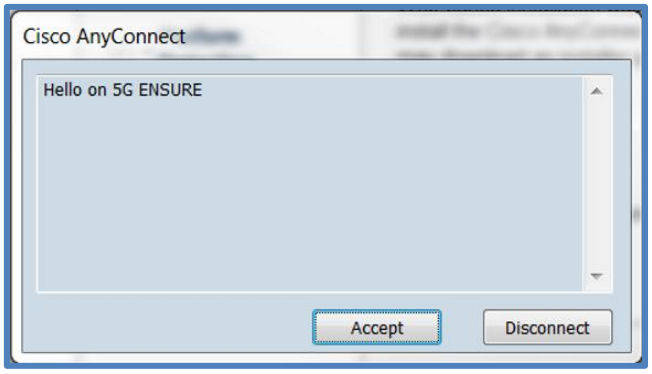

Figure 38: DialUP VPN AnyConnect welcome banner

- At this stage it should possible to connect to any of the hosts running on the testbed using the applications from the local computer.

If for some reason the procedure does not work as expected, report the incident following the operational procedure described in section 5.2.

5.4 Remote site interconnection

As defined on previous chapters, the testbed is expected to grow during the project lifetime. A number of partner's remote sites or other 5G-PPP testbed platforms should be interconnected with the primary testbed location in a secured way. The initial setup includes the interconnection of VTT and b<>com testbeds. This interconnection is already operational and has been qualified as reported in chapter 4.3.

As already defined, the preferred method to connect a new site is a site-to-site IPSec connection. In order to commit on the parameters for the IKE and the IPSec tunnel establishment, an interconnection setup form has been defined and it's available under request. The main configuration information is depicted in **Table 2**.

The form is prefilled with a standard configuration that should be supported by most of the security equipment. The proposed configuration has been validated between the main testbed providers (b<>com and VTT).

Once the form is completed, it should be addressed to the testbed mailing list in order to build the VPN connectivity between the sites.

Once the VPN is established, a characterization of the interconnection capacities must be driven based on the protocol established on the section 4.3. This action will qualify the quality of the link and will allow determining the type of tests that could be driven through the interconnection.

5.5 Testbed enabler delivery

A catalogue tool (Artifactory [16]) will be provided by the testbed owner to store the packaged enablers. Enabler's packaging is an operational requirement for the enablers to be deployed on the testbed. The access to the catalogue will require the user's credentials. There are available screencasts [17] [18] [19] and a complete user guide [27] describing the generic procedures of the catalogue usage. Here is the outline of the testbed enabler delivery procedure:

- Connect to the catalogue repository using a web browser. A web page looking like the following should come up:

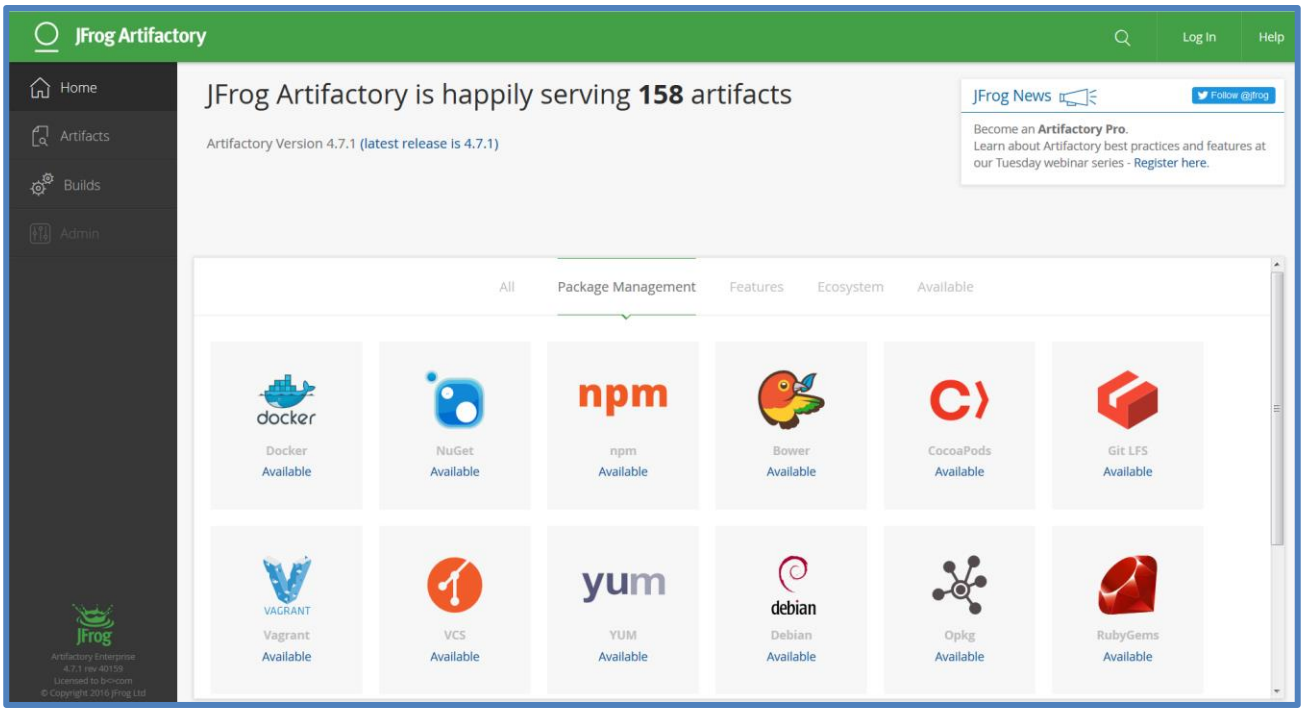

Figure 39: Catalogue home page

- Use the credentials to get the access to the repository.
- Among the provided 5G-ENSURE local repositories, choose the one corresponding to the enabler's bundle (rpm, deb package, Docker container) and upload the new package. The repositories will look something like :
    - Local-5G-Ensure-deb (Screencast available at [17])
    - Local-5G-Ensure-yum (Screencast available at [18])
    - Local-5G-Ensure-docker (Screencast available at [19])
- It is also possible to perform the action from the command line. In order to get the right format, use the "**set me up**" button to get the configuration setup for the current repository. Output will look like:

To deploy a Debian package into Artifactory you can either use the deploy option in the Artifact's module or upload with cURL using matrix parameters. The required parameters are package name, distribution, component, and architecture in the following way:

curl -u<USERNAME>:<PASSWORD> -XPUT "https://repository.lab.5Gensure.eu/local-5g-Ensure-deb/pool/<DEBIAN_PACKAGE_NAME>;deb.distribution=<DISTRIBUTION>;deb.component=<COMPONENT>;deb.architecture=<ARCHITECTURE>" -T <PATH_TO_FILE>

You can specify multiple layouts by adding semicolon separated multiple parameters, like so:

curl -u<USERNAME>:<PASSWORD> -XPUT "https://repository.lab.5Gensure.eu/local-5g-Ensure-deb/pool/<DEBIAN_PACKAGE_NAME>;deb.distribution=<DISTRIBUTION>;deb.distribution=<DISTRIBUTION>;deb.component=<COMPONENT>;deb.component=<COMPONENT>;deb.architecture=<ARCHITECTURE>;deb.architecture=<ARCHITECTURE>" -T <PATH_TO_FILE>

To add an architecture independent layout use deb.architecture=all. This will cause your package to appear in the Packages index of all the architectures under the same Distribution and Component, as well as under a new index branch called binary-all which holds all Debian packages that are marked as "all".

Figure 40: Catalogue update configuration

- Perform the required actions (i.e. index regeneration for Ubuntu repository packages) to update the project metadata.

At this point, it should be possible to install / update the enabler from host system by using repository management tool, or deploy a new Docker container.

For the instances deployed on the testbed, the configuration management tool will configure their repository to point to the catalogue. However if for some reason a manual configuration is required, use the "**set me up**" button to get the configuration that needs to be applied to use the current repository. For instance, for the current example, the output configuration would be:

To use Artifactory repository to install Debian package you need to add it to your source.list file. You can do that using the following command:

sudo sh -c "echo 'deb https://repository.lab.5gensure.eu/local-5g-Ensure-deb <DISTRIBUTION> <COMPONENT>' >> /etc/apt/sources.list"

For accessing Artifactory using credentials you can specify it in the source.list file like so:

https://<USERNAME>:<PASSWORD>@repository.lab.5gensure.eu/local-5g-Ensure-deb <DISTRIBUTION> <COMPONENTS>

Your apt-get client will use the specified Artifactory repositories to install the package

apt-get install <PACKAGE>

Figure 41: Catalogue repository configuration

5.6 Deploying an enabler on the testbed

There are two identified methods to deploy an enabler on the testbed:

- On a local instance on the testbed sites, meaning that the enabler is on the 5G-ENSURE repository and that it can be instantiated on the provided testbed environment (orchestration tools and so on).
- On Software as a Service mode, meaning that the Enabler Owner will deploy the component on its premises and will perform a site interconnection with the testbed.

The former solution is the preferred one as it will allows a better end-to-end integration of the enabler and ensures a higher degree of compliancy and uniformity among the enablers.

The latter solution is expected to be used only in the case when it is not possible to perform a local instantiation. As such security enablers delivered in SaaS mode should be the exception and would require to be duly justified.

SaaS mode will not be deeply covered by the document as it is believed that a tailored solution will be required for each enabler running in this mode.

5.6.1 Enabler local instantiation in the testbed

In order to provide the required degree of conformity for a telco grade platform, the deployment of the testbed instances will be handled by the Testbed Operator who will ensure that the required engineering rules are applied to all the instances running on the testbed.

Once the host instance is deployed, it is up to the Enabler Owner to define the way to install, update, and configure the enabler instance.

Below is the deployment procedure for enablers running on standalone instances:

- The Enabler Owner delivers the enabler to the 5G-ENSURE repository in one of the supported bundle formats.
- The Enabler Owner requests the deployment of the enabler through the helpdesk. The information required are the following:

  - Instance flavour
  - Network architecture
  - Connection to other components
- The request will be validated and completed with some complementary information:
  - IP address
  - Host name
  - Service name
  - Hypervisor hosting the NFV (if virtualized)
- The Testbed Operator team will create the instance on the testbed based on the provided requirements and will apply the engineering rules.
- If the enabler is provided with a configuration template, it will be applied to newly created instance to complete the deployment. Other way, the Enabler Owner will be capable of deploying the enabler application and complete the configuration of the instance.

In some cases, the enabler will interface with a shared component of the platform like MME, HSS or eNodeB. As the choice of the integration framework and the way to deal with is not yet defined (not in the scope of Release 1), it is not possible to define the operational procedure to perform the deployment, but most probably, these network functions will be handled by the Testbed Operator team.

5.6.2 Enabler in SaaS mode

Setting up an enabler in SaaS mode implies a complex configuration with many parameters, too specific to describe with an operational procedure in this document.

Anyhow, the initial step would be to contact the Testbed Operator team following the support procedure and define the requirements for the given enabler. A site-to-site VPN interconnection will be required in order to allow the interconnection between the testbed main facility and the remote site hosting the enabler.

5.7 Test management

Deliverables D4.2 and D4.3 will cover the test plan and the associated testbed activities later on. Here, a high level description of the operational procedures allowing the testbed user to book the testbed resources and deal with the dataset management will be provided.

5.7.1 Testbed resources booking

With regard to the way the testbed has been defined, it appears that resources management will be required in order to grant that operations conducted on the testbed can be driven at a given period of time. The main idea is to be able to check the availability and book all the resources required.

In order to do so, the Testbed Support team will handle a **laboratory notebook** allowing to track the testbed usage and manage the testbed resources booking.

The laboratory notebook will be based on an online tool and the testbed users will be able to request the reservation of a part or the entire testbed. As the data archival is highly important, there should not be parallel testing sessions on the same component to simplify the data archival and analysis, at least during the test plan execution.

5.7.2 Dataset management (data generation / collection, test management tool)

Another important aspect is to cover the dataset management. As already mentioned, the test plan will be initially defined on the deliverable D4.2 and updated on D4.3. Here, there will be introduced the procedures

that will be implemented on the testbed in order to support the test plan definition, execution and the data archival.

B<>com will propose an online tool to drive the tests. This tool is not available yet and the test plan is not implemented, so the procedure can only be described at high level. The tool's user guide [21] and some screen casts [22] are available to provide a general approach of the tool.

To begin with, a test plan will be created on the tool, the testbed users that will do part of the test plan editing will be added to test plan as Editors.

Once the test plan is available, the testbed users participating to the test plan execution will be added to the test tool. They will be affected to the set of tests that will be run by them. The users will get the role of test Executor.

Afterwards, the testbed resources have to be booked on the testbed booking tool by the test Executors, for the time period where the tests will be performed. During the execution of the test plan, it is strictly forbidden to run other tests in parallel on the same components of the testbed. It is a major constraint to properly identify the components impacted by a given test.

For a given test, prepare the traces, probes, etc that will allow to validate or troubleshoot the test result.

Run the test and collect the traces. There will be a resilient storage available for the project. The way to access it may vary, depending on the enabler. However, all the data collected should be stored on this storage. The way to store the data should be structured like:

- **<test_plan> / <test_build> / <test_case>**

Report the test result to the dataset management tool.

Return to the original "clean" configuration without residual changes to the components (required by the test).

The platform configuration should not be modified during the duration of a test plan execution unless explicitly mentioned by one of the tests (i.e. activate and deactivate a function to check impact). The goal is to avoid obtaining inconsistent results.

It is not currently planned to connect this tool with an automatic test tool like **Jenkins** and, due to the nature of the tests that will be conducted, it does not look necessary to couple it to a bug tracker. However both integrations may be possible.

6 Conclusion

This document introduces the 5G security testbed architecture that complies with 5G-ENSURE project's enablers requirements (first and second release as reported in the Technical Roadmap D3.1 [2]). While the 5G-ENSURE architecture is still work-in-progress, as far as the working hypothesis is considered, no conflicts have been identified on architectural level. The overarching goal of the testbed is to allow to host 5G-ENSURE security enablers and get them tested against their claims to address 5G security needs identified through use cases in D2.1 [8].

The testbed is under development and has been designed to meet enabler requirements based on information available. The testbed will be released by M9 (i.e. July'16) (MS4.1 Testbed up and running).

The 5G-ENSURE enablers are now under development and are expected to be delivered on due time by M11 (i.e. September 2016) in the context of the next deliverables to come (namely D3.3 "Security enablers sw release (v1.0)" and D3.4 "Security enablers documentation (v1.0)", D4.2 "Test plan" (M12 for the first release). Based on this delivery in due time, all these enablers (Release R1) are expected to be integrated in the testbed between M13 and M16, and ready for test evaluation.

The work reported in this document was performed within WP4 and in close collaboration with other technical work packages and exploited the deliverables already produced by the project, especially D2.1 [8] on Use Cases, D3.1 [2] on Technical Roadmap, D3.2 [3] on enablers Open Specifications and the deliverables under development, especially D2.3 Risk assessment, mitigation and requirements (draft M9), early version of D2.4 Security architecture (draft M12).

The testbed described here is planned to be evolved and enriched based on lessons learnt from enablers R1 integration but also information that will come from next deliverables.

This testbed architecture is also expected to be shared with 5G-PPP community at large starting first and foremost with 5G-PPP Security community through 5G-PPP Security WG activities performed.

References

[1] 5G-PPP, “5G-PPP 5G Architecture White Paper,” [Online]. Available: https://5g-ppp.eu/wp-content/uploads/2014/02/5G-PPP-5G-Architecture-WP-For-public-consultation.pdf.

[2] 5G-ENSURE, “5G-ENSURE D3.1 5G-PPP Security Enablers Technical Roadmap early vision,” [Online]. Available: http://5gensure.eu/sites/default/files/Deliverables/5G-ENSURE_D3.1-5G-PPPSecurityEnablersTechnicalRoadmap_early_vision.pdf.

[3] 5G-ENSURE, “5G-ENSURE D3.2 5G-PPP security enablers open specifications (v1.0),” [Online]. Available: http://5gensure.eu/sites/default/files/Deliverables/5G-ENSURE_D3.2-5G-PPPSecurityEnablersOpenSpecifications_v1.0.pdf.

[4] Wikipedia, “Wikipedia - Orchestration,” [Online]. Available: https://en.wikipedia.org/wiki/Orchestration_(computing).

[5] Wikipedia, “Wikipedia - Blackbox definition,” [Online]. Available: https://en.wikipedia.org/wiki/Black_box.

[6] Business_Dictionary, “Business Dictionary - White box,” [Online]. Available: http://www.businessdictionary.com/definition/white-box.html.

[7] IETF, “IETF RFC 7498 Problem Statement for Service Function Chaining,” [Online]. Available: https://tools.ietf.org/html/rfc7498.

[8] 5G-ENSURE, “5G-ENSURE D2.1 Uses Cases,” [Online]. Available: http://5gensure.eu/sites/default/files/Deliverables/5G-ENSURE_D2.1-UseCases.pdf.

[9] 5GNorma, “5G Norma D2.1 Use cases, scenarios and requirements,” [Online]. Available: https://5gnorma.5g-ppp.eu/wp-content/uploads/2015/11/5G-NORMA_D2.1.pdf.

[10] “Open Air Interface,” [Online]. Available: http://www.openairinterface.org/.

[11] A. Diez, “Understanding NFV Management and Orchestration,” 2015.

[12] “OpenEPC Home Page,” [Online]. Available: http://www.openepc.com.

[13] “ETSI NFV home page,” [Online]. Available: http://www.etsi.org/technologies-clusters/technologies/nfv.

[14] 5GNorma, “5G Norma D3.1 Functional network architecture and security requirements,” [Online]. Available: https://5gnorma.5g-ppp.eu/wp-content/uploads/2016/01/5G_NORMA_D3.1.pdf.

[15] “KVM4FV,” [Online]. Available: http://artifacts.opnfv.org/kvmfornfv/docs/all/all.pdf.

[16] “Artifactory home page,” [Online]. Available: https://www.jfrog.com/confluence/display/RTF/Welcome+to+Artifactory.

[17] “Artifactory as a Debian repository,” [Online]. Available: https://www.jfrog.com/video/setting-up-artifactory-4-as-a-debian-repository-in-minutes/.

[18] “Artifactory as a YUM repository,” [Online]. Available: https://www.jfrog.com/video/artifactory-yum-repository/.

[19] “Artifactory as a Docker registry,” [Online]. Available: https://www.jfrog.com/video/install-artifactory-docker-registry-one-minute-less/.

[20] “TestLink home page,” [Online]. Available: http://testlink.org/.

[21] “Testlink user manual,” [Online]. Available: https://wiki.openoffice.org/w/images/1/1b/Testlink_user_manual.pdf.

[22] “TestLink Screencast,” [Online]. Available: https://www.youtube.com/watch?v=6s48WGuX2WE.

[23] “RHEL Virtualisation KVM timing management,” [Online]. Available: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Virtualization_Deployment_and_Administration_Guide/chap-KVM_guest_timing_management.html.

[24] “Cisco Anyconnect,” [Online]. Available: http://www.cisco.com/c/en/us/support/security/anyconnect-secure-mobility-client/tsd-products-support-series-home.html.

[25] “IPerf home page,” [Online]. Available: https://iperf.fr/.

[26] VTT, “QoSMet home page,” [Online]. Available: http://www.vttresearch.com/qosmet.

[27] “Artifactury user manual,” [Online]. Available: https://www.jfrog.com/confluence/display/RTF/Welcome+to+Artifactory.

[28] Ansible, “Ansible home page,” [Online]. Available: https://www.ansible.com/.

A Annex : Interconnection characterization scripts

Table 3. Shell script for VPN tunnel TCP and UDP throughput and RTT performance measurement

```
#!/bin/bash

#IP address of the receiving side (non-master)
ADDR=[IP_ADDRESS]
#Duration of measurement runs
TIME=30
#Start bitrate for UDP measurement
UDP_BITRATE=30
#End bitrate fo UDP measurement
UDP_BITRATE_END=150
M=M

#Measure 24h every 30 minutes
for i in {1..48}
do
  #TCP measurement with 4 parallel streams and omit results from first second
  echo Starting TCP measurement$'\n'
  echo *****START $(date)**********$'\n' >> TCP-Results.txt
  iperf3 -c $ADDR -w 64M -P 4 -t $TIME -O 1 --logfile TCP-Results.txt
  echo *****END $(date)**********$'\n' >> TCP-Results.txt

  #UDP measurement
  echo *****START $(date)**********$'\n' >> UDP-Results.txt
  while [ $UDP_BITRATE -le $UDP_BITRATE_END ]
  do
    echo Starting UDP measurement with bitrate $UDP_BITRATE Mb/s$'\n'
    iperf3 -A 0 -c $ADDR -u -i 1 -b $UDP_BITRATE$M -t $TIME --logfile UDP-Results.txt
    UDP_BITRATE=$(($UDP_BITRATE + 10))
  done
  UDP_BITRATE=30
  echo *****END $(date)**********$'\n' >> UDP-Results.txt

  #Measure RTT
  echo *****STAR $(date)**********$'\n' >> Ping_output.txt
  echo Starting Ping measurement$'\n'
  ping $ADDR -c 10  >> Ping_output.txt
  echo *****END $(date)**********$'\n' >> Ping_output.txt

  #Wait until the next measurement round
  echo Waits until the next measurement round$'\n'
  sleep 1370
done

echo Measurements performed$'\n'
```

Table 4. Shell script for VPN tunnel performance measurement with differently sized MTUs

```
#!/bin/bash

#IP address of the receiving side (non-master)
ADDR=10.10.100.2
```

```
#Duration of measurement runs
TIME=30
#MTU start value
MTU=1500
#MTU end value
MTU_END=100
INTERFACE="eth0"

#Measure with different MTU sizes
while [ $MTU -ge $MTU_END ]
do
  #Set MTU
  echo Setting MTU to $MTU$'\n'
  ifconfig $INTERFACE mtu $MTU
  #TCP measurement with 4 parallel streams and omit results from first second
  echo Starting TCP measurement$'\n'
  echo *****START $(date)**********$'\n' >> TCP-MTU-Results.txt
  iperf3 -c $ADDR -w 64M -P 4 -t $TIME -O 5 --logfile TCP-MTU-Results.txt
  echo *****END $(date)**********$'\n' >> TCP-MTU-Results.txt

  sleep 5

  #Make measurement in reverse direction
  echo *****START $(date) R**********$'\n' >> TCP-MTU-Results.txt
  iperf3 -c $ADDR -w 64M -P 4 -t $TIME -O 5 -R --logfile TCP-MTU-Results.txt
  echo *****END $(date) R**********$'\n' >> TCP-MTU-Results.txt

  MTU=$(($MTU - 100))

  #Wait until the next measurement round
  echo Waits 5 sec until the next measurement round$'\n'
  sleep 5

done

MTU=1500
ifconfig $INTERFACE mtu $MTU

while [ $MTU -ge $MTU_END ]
do
  #Set MTU
  echo Setting max segment size to $MTU$'\n'
  #TCP measurement with 4 parallel streams and omit results from first second
  echo Starting TCP measurement$'\n'
  echo *****START $(date)**********$'\n' >> TCP-PKTSIZE-Results.txt
  iperf3 -c $ADDR -w 64M -P 4 -t $TIME -O 5 -M $MTU --logfile TCP-PKTSIZE-Results.txt
  echo *****END $(date)**********$'\n' >> TCP-PKTSIZE-Results.txt

  sleep 5

  #Make measurement in reverse direction
  echo *****START $(date) R**********$'\n' >> TCP-PKTSIZE-Results.txt
  iperf3 -c $ADDR -w 64M -P 4 -t $TIME -O 5 -R -M $MTU --logfile TCP-PKTSIZE-Results.txt
  echo *****END $(date) R**********$'\n' >> TCP-PKTSIZE-Results.txt

  MTU=$(($MTU - 100))
```

```
  #Wait until the next measurement round
  echo Waits 5 sec until the next measurement round$'\n'
  sleep 5
done

echo Measurements performed$'\n'
```

B Annex : Enablers hosting requirements

In this appendix there is the collection of enabler requirements regarding the hosting on the testbed. For a shake of readability, common requirements are listed hereunder while the rest of the data is depicted in the **Table 5**:

- NTP server needs to be in stratum ≤ 4.
- All the enablers will be instantiated on the test (not in SaaS mode)
- All the enablers can be virtualised
- All the enabler will use Ansible [28] as configuration management tool
- Network bandwidth requirements is less than 100Mbps for all the enablers

In the testbed, the following virtualisation flavours have been defined to match enabler's capacity requirements:

- vSmall (1 CPU, 2 GB RAM, 20 GB HDD)
- vMedium (2 CPU, 4 GB RAM, 40 GB HDD)
- vLarge (4 CPU, 8 GB RAM, 80 GB HDD)

Table 5: Enablers hosting requirements

		Delivery	Enabler Instance requirements				Virtualisation	NFV/SDN enabler's constraints			Integrationto testbed		Interconnection		
	Enabler Name	Bundle	Flavor	Software	Configuration	Instances	Hypervisor	Protocol	Version	Controller	VPN client	Admin network	To other security enablers	To other components	Other constraints
AAA	*Internet of things - IOT*	Package	2 x vMedium (MME, HSS) 5 x vSmall (UEs/eNodeBs)	OS: Ubuntu LTS Pwk : OAI FS : ext4 Kernel : Low latency	OAI MME, HSS OAISIM	7	KVM	N/A	N/A	N/A	Yes for Linux and OSX 10.11	Not Defined	None	UE, ENODEB, MME, HSS	None
	Fine-grained Authorization	Package	vMedium	OS : Ubuntu LTS JDK7 Tomcat 7 FS : ext4	None	1	KVM	N/A	N/A	N/A	Yes, TBC by TASE iT	Not Defined	None (in principle)	AAA	None
		Package	vMedium	OS : Ubuntu LTS JDK7 Tomcat 7 FS : ext4	None	1	KVM	N/A	N/A	N/A	Yes	Not Defined	None (in principle)		None
Privacy	*Privacy Enhanced Identity Protection*	Package	1 x vSmall (UE) 1 x vMedium (Core)	OS: Ubuntu LTS FS : ext4	UE : Wifi dongle WPA supplicant card reader accessed via PCSC Testing SIM card Core : Host Apd HLR_AU_GW HSS	2	KVM	N/A	N/A	N/A	Yes	Yes	No	None in R1 In R2: eNodeB, WIFI AP, UE, AAA, HSS	No
	Device identifier(s) privacy	Package	vSmall	OS: Ubuntu LTS FS : ext4	UE : Wifi dongle WPA supplicant	1	KVM	N/A	N/A	N/A	Yes	Yes (already defined)	No	This enblers requires several WIFI access networks with different simulated DHCP setups	No
Trust	*Trust Builder*	Package	vLarge	OS : Ubuntu LTS FS : ext4 Docker	One VM with docker support 2 Docker containers - Web application - TripleStore	1	KVM + Docker	N/A	N/A	N/A	N/A	Not Defined	None	None	None
	Trust Metric Enabler	Package	vSmall	OS: Ubuntu LTS FS : ext4	None	1	KVM	N/A	N/A	N/A	Yes	Not defined	Micro segment enabler and security monitor	UE, Trust Model, Network statistics	None
	VNF Certification	Package + Docker	vLarge	OS: Ubuntu LTS FS : ext4 JDK8 Eclipse(Mars) Docker	1 VM with docker support 1 docker container: - DtWC Repository	1	KVM + Docker	N/A	N/A	N/A	Yes	Not Defined	None	SDN controller, repository	None
Security Monitoring	*Security Monitor for 5G Micro-Segments*	Docker	vLarge	OS: Ubuntu LTS FS : ext4	None	1	KVM	N/A	N/A	N/A	Yes	Not Defined	Micro segment enabler	SDN Controller	None
	Generic Collector Interface	Package	vMedium	OS: Ubuntu LTS FS : ext4	None	1	KVM	N/A	N/A	N/A	Yes	Not Defined	No	R1 None / R2 all components	None
	PulSAR: Proactive Security Analysis and Remediation	Docker	vLarge	OS: Ubuntu LTS FS : ext4 Docker	One VM with docker support 2 Docker containers	1	KVM + Docker	N/A	N/A	N/A	No	Not Defined	Security Monitoring	SDN Controllers NFV Orchestrator	None
Network Management	*Access Control Mechanisms*	Package	1 x vLarge (Controller) 3 - 10 x vSmall (SDN SW)	OS: Ubuntu LTS FS : ext4	One host dedicated to build a SDN network. 4 to 10 instances	4-10	KVM	Open Flow	v1.3 & v1.4	Onos (Modified)	Yes	Yes	Modified SDN controller interacts with data plane components via OpenFlow	SDN Controller At least one software switch and at least one OpenFlow hardware switch	The requirements are similar to the enabler "Component-interaction Audits".
	Component-Interaction Audits	Package	1 x vLarge (Controller) 3 - 10 x vSmall (SDN SW)	OS: Ubuntu LTS FS : ext4	The enviroment will be mutualized with the previous Enabler	enviroment mutualized with previous Enabler	KVM	Open Flow	v1.3 & v1.4	Onos (Modified)	Yes	Yes	Yes. Enabler communicates over a UDP socket with other network coponents (SDN controller, software switch)	SDN Controller + OVS. At least one software switch (Modified OVS version)	The requirements are similar to the enabler "Access Control Mechanisms".
	Bootstrapping Trust	Package	vMedium	OS: Ubuntu LTS FS : ext4	1 controller 2 Endpoints with traffic generators & OVS SDN Switches (same host)	3	KVM	Open Flow	v1.3 & v1.4	Onos	Yes	Yes	No	SDN Controller + SDN Switch. Customised version of OVS	No
	Micro Segmentation	Package	vLarge	OS: Ubuntu LTS FS : ext4 Docker OpenVirtex	1 controller 2 Endpoints with traffic generators & OVS SDN Switches	1-3	KVM	Open Flow	v1.3	RYU	Yes	Yes	Trust metric enabler / Security monitoring enabler	SDN Controller + SDN Switch, R2 (MME, HSS, AAA)	No