Deliverable D3.4
5G-PPP Security Enablers Documentation (v1.1)
Satellite Network Monitoring

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	30.09.2016	
Dissemination Level:	Public	
Lead beneficiary	NEC	Felix Klaedtke, felix.klaedtke@neclab.eu
Authors	Thales Alenia Space: Gorka Lendrino	

Document Version	Date	Change(s)	Author(s)
0.1	28.06.2016	Created template	Felix Klaedtke
1.0 draft	01.08.2016	First draft	Gorka Lendrino
1.0	31.08.2016	First issue	Gorka Lendrino
1.1	30.09.2016	Completed enabler installation	Gorka Lendrino

Foreword

5G-ENSURE belongs to the first group of EU-funded projects which collaboratively develop 5G under the umbrella of the 5G Infrastructure Public Private Partnership (5G-PPP) in the Horizon 2020 Programme. The overall goal of 5G-ENSURE is to deliver strategic impact across technology and business enablement, standardisation and vision for a secure, resilient and viable 5G network. The project covers research & innovation - from technical solutions (5G security architecture and testbed with 5G security enablers) to market validation and stakeholders' engagement - spanning various application domains.

5G security requirements collected in WP2 will be meet investigating and developing a number of 5G security and privacy enablers. An early vision of the proposed enablers has been provided in D3.1 together with a technical roadmap for R1. Subsequently, enablers planned to be software released for R1 have provide an open specification in D3.2.

This enabler's manual is part of deliverable D3.4 and will be accompanied by its SW delivery (D3.3).

Disclaimer

Copyright notice

Contents

1 Introduction

The main goal of this security enabler is to provide pseudo real-time monitoring and threat detection in 5G integrated satellite and terrestrial systems.

“Satellite Network Monitoring” enabler features are in various states of research at the writing time of this document and there is no planned software delivery for all the features. Nevertheless, in the scope of R1 some client-side features are currently being developed as a preliminary open specification and as a proof of concept.

In this SW release, several indicators (including security metrics) will be collected from the listed 5G integrated satellite and terrestrial systems and will be periodically delivered to the monitoring system. The “Generic Collector Interface” enabler will be analysed (R2) in order to be used as a building block. Later, an active security analysis will be used to detect specific threats and vulnerabilities in the satellite network and response to the threats identified.

The manual is organized as follows. In Section 2, we describe enabler is installed, configured, and administrated, and in Section 3, we describe how to use the enabler. In Section 4, we provide the tests needed to cover all the features of the enabler. In Section 5, we thank people for contributions. Finally, abbreviations and references are listed in Section 6 and Section 7.

2 Installation and Administration Guide

The “Satellite Network Monitoring” feature is the unique part of this security enabler. This section covers the system requirements, and describes how the enabler is installed and configured.

2.1 System Requirements

The “Satellite Network Monitoring” enabler consists of:

- A client-side feature (e.g. 5G-eNodeB, satellite terminals, UE....), able to collect indicators from the network component and send them to the server-side feature. This feature shall be deployed on each network component.
- A server-side feature (i.e. security monitoring server), able to configure the client-side features and provide pseudo real-time monitoring of consumed indicators. This feature shall be deployed on a central server.

The basic/minimal system requirements for this enabler are that the server and the network components should be running Linux, preferably Ubuntu Server 16.04.1 LTS or newer on an x64 architecture (you can get help from the Ubuntu server guide [5]), with several network interfaces.

The client host(s) (i.e. 5g-mon-sat-cli01.5g-ensure.eu) in the test-bed is a vSmall virtual host with the following requirement:

- 1 CPU (x86_64)
- 2 GB RAM
- 20 GB Hard disk
- Ethernet network interfaces:
    - Control interface
    - Terrestrial data interface
    - Satellite data interface

It will act as a container of 5G-eNodeBs.

The server host (i.e. 5g-mon-sat-srv01.5g-ensure.eu) in the test-bed is a vSmall virtual host with the following requirement:

- 2 CPUs (x86_64)
- 4 GB RAM
- 40 GB Hard disk
- Ethernet network interfaces:
    - Control interface
    - Terrestrial data interface
    - Satellite data interface

It will act as a security monitoring server.

The client-server API follows the RESTful Web Services (JAX-RS 2.0), therefore a Java Application Server is needed in both sides (any should work, but only Apache Tomcat has been tested):

- Client side: to configure the client-side.
- Server side: to configure the server-side.

Also, the server-side feature uses:

- An Apache Active MQ instance
- A PostgreSQL instance

Remote service access:

- Host access to Ubuntu repositories (main, universe).
- Host access to NTP server of stratum ≤ 4.
- SSH access to host (public access or restricted to specific public source IP address).
- HTTPS (TCP/443) access to host (public access or restricted to specific public source IP address).

2.2 Enabler Installation

The installation package contains two archived web applications (client-side and server-side).

Download the enabler package from the software repository of the 5G-ENSURE project (e.g. https://<5g-ensure.git>/enablers/SatelliteNetworkMonitoring/archive/SatelliteNetworkMonitoring.vXX_YY_ZZ.tar.gz) later extract the package on the temporal folder **/tmp** or clone the GitHub project (e.g. https://<5g-ensure.git>/enablers/SatelliteNetworkMonitoring.git) on the temporal folder **/tmp**.

As a result, all the installation files of this enabler will be located in **/home/tase/SatelliteNetworkMonitoring.vXX_YY_ZZ**.

2.2.1 Server-side

First of all disable unattended updates:

$ sudo systemctl disable apt-daily.service

$ sudo systemctl disable apt-daily.timer

$ sudo shutdown -r now

Add tase user to the sudoers file:

$ cd SatelliteNetworkMonitoring.vXX_YY_ZZ/server

$ mkdir cp

$ sudo cp -p /etc/sudoers cp/

$ sudo tee -a /etc/sudoers << EOF

tase ALL=(ALL:ALL) NOPASSWD:ALL

EOF

Update package information:

$ sudo apt-get install -f

$ sudo dpkg --configure -a

$ sudo apt-get update

Configure time zone:

$ sudo timedatectl set-timezone UTC

Configure host name:

$ sudo cp -p /etc/hosts cp/

$ sudo hostnamectl set-hostname '5g-mon-sat-srv01'

$ sudo sed -i s/ubuntu/"5g-mon-sat-srv01.5g-ensure.eu\t5g-mon-sat-srv01"/g /etc/hosts

Add the MON_SAT_PATH environment variable to tase user:

```
$ sudo tee -a /home/tase/.bashrc << EOF
export MON_SAT_PATH=~/SatelliteNetworkMonitoring.v01_00_00/
EOF
```

Install common dependencies:

```
$ sudo apt install -y traceroute
```

Install ActiveMQ:

```
$ sudo apt install -y openjdk-8-jre
$ wget -O /tmp/apache-activemq-5.14.0-bin.tar.gz http://apache.uvigo.es/activemq/5.14.0/apache-activemq-5.14.0-bin.tar.gz
$ sudo mkdir /opt/apache-activemq
$ sudo tar xzvf /tmp/apache-activemq-5.14.0-bin.tar.gz -C /opt/apache-activemq
```

Install PostgreSQL:

```
$ sudo apt install -y postgresql postgresql-contrib
$ sudo -u postgres psql postgres
postgres=# \password postgres
Enter new password:
Enter it again:
postgres=# \q
$ sudo su - postgres
$ psql
postgres=# CREATE USER tase WITH PASSWORD 'tasetase';
postgres=# CREATE DATABASE snm OWNER tase;
postgres=# \q
$ exit
$ sudo su - tase
$ psql snm
snm=> CREATE TABLE event (
        event_id SERIAL PRIMARY KEY,
        host TEXT NOT NULL,
        community TEXT NOT NULL,
        oid TEXT NOT NULL,
        value INT NOT NULL);
snm=> CREATE TABLE alarm (
```

```
        alarm_id SERIAL PRIMARY KEY,
        host TEXT NOT NULL,
        community TEXT NOT NULL,
        oid TEXT NOT NULL,
        value INT NOT NULL);
snm=> \q
$ exit
```

Install docker:

```
$ sudo apt-get install -y apt-transport-https ca-certificates
$ sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys 58118E89F3A912897C070ADBF76221572C52609D
$ sudo tee /etc/apt/sources.list.d/docker.list << EOF
deb https://apt.dockerproject.org/repo ubuntu-xenial main
EOF
$ sudo apt-get update
$ sudo apt-get purge lxc-docker
$ sudo apt-get install -y linux-image-extra-$(uname -r) linux-image-extra-virtual
$ sudo apt-get install -y docker-engine
$ sudo usermod -aG docker tase
```

Install docker compose:

```
$ curl -L https://github.com/docker/compose/releases/download/1.8.0/docker-compose-`uname -s`-`uname -m` > docker-compose
$ sudo mv docker-compose /usr/local/bin/docker-compose
$ sudo chmod +x /usr/local/bin/docker-compose
```

Create the terminal docker image:

```
$ cd ..
$ cd common/Dockerfile_terminal
$ docker build -t terminal_img .
$ docker save terminal_img > terminal_img.tar
$ cd ..
```

Create the terminal docker image:

```
$ cd common/Dockerfile_5g-enodeb/
$ docker build -t 5g-enodeb_img .
```

$ docker save 5g-enodeb_img > 5g-enodeb_img.tar

$ cd ..

Alternatively, use the following command:

$ /tmp/SatelliteNetworkMonitoring/server/install-server.sh

Among other things, this script allows to:

- Set time zone to UTC.
- Configure the server IP address.
- Map hostnames to IP addresses.
- Configure JAVA_HOME and PATH environment variables.
- Create the MON_SAT_PATH environment variable.
- Install the required COTS.

2.2.2 Client-side

First of all disable unattended updates:

$ sudo systemctl disable apt-daily.service

$ sudo systemctl disable apt-daily.timer

$ sudo shutdown -r now

Add tase user to the sudoers file:

$ cd SatelliteNetworkMonitoring.vXX_YY_ZZ/client

$ mkdir cp

$ sudo cp -p /etc/sudoers cp/

$ sudo tee -a /etc/sudoers << EOF

tase ALL=(ALL:ALL) NOPASSWD:ALL

EOF

Update package information:

$ sudo apt-get install -f

$ sudo dpkg --configure -a

$ sudo apt-get update

Configure time zone:

$ sudo timedatectl set-timezone UTC

Configure host name:

$ sudo cp -p /etc/hosts cp/

$ sudo hostnamectl set-hostname '5g-mon-sat-cli01'

```
$ sudo sed -i s/ubuntu/"5g-mon-sat-cli01.5g-ensure.eu\t5g-mon-sat-cli01"/g /etc/hosts
```

Install common dependencies:

Add the MON_SAT_PATH environment variable to tase user:

```
$ sudo tee -a /home/tase/.bashrc << EOF
export MON_SAT_PATH=~/SatelliteNetworkMonitoring.v01_00_00/
EOF
$ sudo apt install -y traceroute
```

Install SQLite:

```
$ sudo apt-get install -y sqlite3 libsqlite3-dev
$ cd /home/tase/SatelliteNetworkMonitoring.v01_00_00/client
$ sqlite3 snm-client.db
$ sqlite> CREATE TABLE processes (
        processes_id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
        name TEXT NOT NULL,
        status INTEGER NOT NULL,
        heartbeat INTEGER NOT NULL );
$ sqlite> INSERT INTO processes VALUES (1, 'SnmpManager', 0, 0);
$ sqlite> .exit
```

Install docker:

```
$ sudo apt-get install -y apt-transport-https ca-certificates
$ sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 --recv-keys
58118E89F3A912897C070ADBF76221572C52609D
$ sudo tee /etc/apt/sources.list.d/docker.list << EOF
deb https://apt.dockerproject.org/repo ubuntu-xenial main
EOF
$ sudo apt-get update
$ sudo apt-get purge lxc-docker
$ sudo apt-get install -y linux-image-extra-$(uname -r) linux-image-extra-virtual
$ sudo apt-get install -y docker-engine
$ sudo usermod -aG docker tase
```

Install docker compose:

```
$ curl -L https://github.com/docker/compose/releases/download/1.8.0/docker-compose-`uname -s`-
`uname -m` > docker-compose
```

$ sudo mv docker-compose /usr/local/bin/docker-compose

$ sudo chmod +x /usr/local/bin/docker-compose

Create the terminal docker image:

$ cd ..

$ cd common/Dockerfile_terminal

$ docker build -t terminal_img .

$ docker save terminal_img > terminal_img.tar

$ cd ..

Create the terminal docker image:

$ cd common/Dockerfile_5g-enodeb/

$ docker build -t 5g-enodeb_img .

$ docker save 5g-enodeb_img > 5g-enodeb_img.tar

$ cd ..

Alternatively, use the following command:

```
$ /tmp/SatelliteNetworkMonitoring/client/install-client.sh
```

Among other things, this script allows to:

- Set time zone to UTC.
- Configure the client IP address.
- Map hostnames to IP addresses.
- Configure JAVA_HOME and PATH environment variables.
- Create the MON_SAT_PATH environment variable.
- Install the required COTS.

2.3 Enabler Configuration (TBD)

2.3.1 Server-side

For security of communications it is highly recommended to enable SSL/TSL using the following command:

```
$ /tmp/SatelliteNetworkMonitoring/server/security-server.sh
```

Among other things, this script allows to:

- Generate RSA keys and self-signed certificate

2.3.2 Client-side

For security of communications it is highly recommended to enable SSL/TSL using the following command:

```
$ /tmp/FGS-InSatelliteSystems/client/security-client.sh
```

This script configures SSL/TSL accordance with the server (see 2.3.1).

3 User and Programmer Guide

This section describes how to use the enabler and how to "program" the enabler.

3.1 User Guide

This section is not applicable as the enabler is a programmer tool; therefore there are no different user and programmer guides.

3.2 Programmer Guide

The enabler provides a REST interface, a resource-oriented API accessed via HTTP that uses JSON-based representations for information interchange. This Programmers Guide mainly consists of an overview of these RESTful API calls.

Description	Configure the indicators to be collected in the network element (i.e. eNodeB
HTTP Method	POST
URI	https://5g-mon-sat-cli01.5g-ensure.eu:8080/mon-sat-cli/api/v01.00.00/sna/resource/indicators
URI Parameters	N/A
Request Header	N/A
Request Body Content-type	application/json
Request Body Content	indicators:[indicatorType]
HTTP status code	201 when success 400 when error
Response Header	N/A
Accept Content-type	application/json
Response Body Content	N/A

Description	Configure the network element
HTTP Method	POST
URI	https://5g-mon-sat-cli01.5g-ensure.eu:8080/mon-sat-

	cli/api/v01.00.00/sna/resource/topology
URI Parameters	N/A
Request Header	N/A
Request Body Content-type	application/json
Request Body Content	topologies:[{interfaceName,interfaceStatus}]
HTTP status code	201 when success 400 when error
Response Header	N/A
Accept Content-type	application/json
Response Body Content	N/A

4 Unit Tests

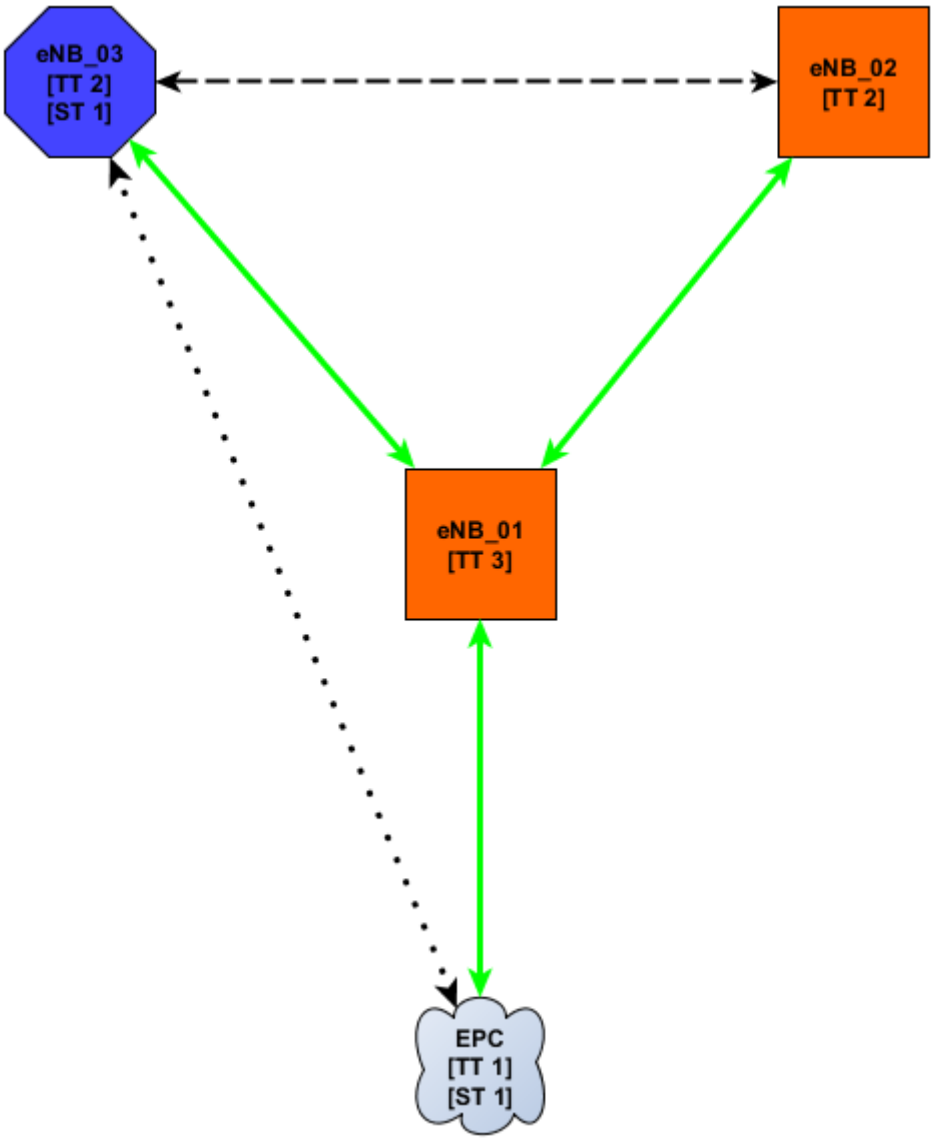
This section describes the tests needed to cover all the features of the enabler. These tests verify that the enabler is ready to be evaluated in WP4.

This features is based on RESTful HTTP API, therefore any HTTP client can be used to perform the tests. For testing purposes curl command line tool has been used.

4.1 Information about Tests

Important: the following tests must be performed in order on the server host, and using $5G_MON_SAT directory as current directory.

The scenario used is described below



Legend:

Objects		
	EPC [TT x] [ST y]	Grey cloud: EPC [TT x] states the number of Terrestrial Terminals [ST x] states the number of Satellite Terminals

		Orange square: eNodeB with terrestrial links [TT x] states the number of Terrestrial Terminals
		Blue octagon: eNodeB with terrestrial and satellite links [TT x] states the number of Terrestrial Terminals [ST x] states the number of Satellite Terminals
Link statuses		
		Green lines: link power-on
		Black lines: potential link (currently power off)
		Red lines: link not enabled
Link types		
		Solid line: fixed link
		Dashed line: Dynamic beam
		Dotted line: Satellite link

4.2 Unit Test 1

This test aims to initialize the indicators to be collected.

The content of the test/UT01/input/UT01_indicators.json file is:

```
{"indicators":[{"indicatorType":< indicatorType 1>},{"indicatorType":< indicatorType 2>},...}
```

Access to the server host and request indicators from the client:

```
$ curl -X POST -H "Content-Type: application/json" -H "Accept: application/json" --data
"@test/UT01/input/UT01_indicators.json" https://5g-mon-sat-cli01.5g-ensure.eu:8080/mon-sat-
```

```
cli/api/v01.00.00/sna/resource/indicators
```

Expected result is HTTP status code 201.

If success, server host will start collecting all the configured indicators (see Apache Tomcat log located in /var/log/tomcat8). These indicators are sent from the client to the server using JMS messages (Apache ActiveMQ).

4.3 Unit Test 2

This test aims to initialize the topology.

The content of the test/UT02/input/UT02_topology.json file is:

```
{"topology":[{"interfaceName":"EPC_FixedLink_eNodeB01","interfaceStatus":"on"},{"interfaceName":"EPC_SatelliteLink_eNodeB03","interfaceStatus":"off"},{"interfaceName":"eNodeB01_FixedLink_EPC","interfaceStatus":"on"},{"interfaceName":"eNodeB01_FixedLink_eNodeB02","interfaceStatus":"on"},{"interfaceName":"eNodeB01_FixedLink_eNodeB03","interfaceStatus":"on"},{"interfaceName":"eNodeB02_FixedLink_eNodeB01","interfaceStatus":"on"},{"interfaceName":"eNodeB02_DynamicBeam_eNodeB03","interfaceStatus":"off"},{"interfaceName":"eNodeB03_FixedLink_eNodeB01","interfaceStatus":"on"},{"interfaceName":"eNodeB03_DynamicBeam_eNodeB02","interfaceStatus":"off"},{"interfaceName":"eNodeB03_SatelliteLink_EPC","interfaceStatus":"off"}}
```

Access to the server host and forward the topology to the clients:

```
$ curl -X POST -H "Content-Type: application/json" -H "Accept: application/json" --data "@test/UT02/input/UT02_topology.json" https://5g-mon-sat-cli01.5g-ensure.eu:8080/mon-sat-cli/api/v01.00.00/sna/resource/topology
```

Expected result is HTTP status code 201.

5 Acknowledgements

The following partners contributed to this deliverable: Thales Alenia Space.

Thanks Jean-Philippe Wary (WPL), 5G-ENSURE peers and 5G-ENSURE Steering Committee for their collaboration and support.

6 Abbreviations

This section comprises a summary of terms and definitions used during the later sections:

5G-PPP	**5G infrastructure Public Private Partnership**
API	**Application Programming Interface**

eNodeB	**evolved Node B**
EPC	**Evolved Packet Core**
HTTP(S)	**HyperText Transfer Protocol (Secure)**
IP	**Internet Protocol**
JMS	**Java Message Service**
JSON	**JavaScript Object Notification**
LTS	**Long Term Support**
LVM	**Logical Volume Manager**
NTP	**Network Time Protocol**
REST	**REpresentational State Transfer**
RSA	**Rivest, Shamir Adleman public-key cryptosystem**
SSH	**Secure SHell**
SQL	**Structured Query Language**
SSL	**Secure Sockets Layer**
ST	**Satellite Terminal**
TCP	**Transmission Control Protocol**
TSL	**Transport Layer Security**
TT	**Terrestrial Terminal**
UE	**User Equipment**
URI	**Uniform Resource Identifier**
UT	**Unit Test**
UTC	**Coordinated Universal Time**

7 References

[1] 5G-Ensure Consortium, Deliverable 2.1 Use Cases. Available online at http://www.5gensure.eu/sites/default/files/Deliverables/5G-ENSURE_D2.1-UseCases.pdf.

[2] 5G-Ensure Consortium, Deliverable 3.1 5G-PPP security enablers technical roadmap. Available online at http://www.5gensure.eu/sites/default/files/Deliverables/5G-ENSURE_D2.1-UseCases.pdf.

[3] Standard: ETSI - TR 101 984 “SatelliteEarth Stations and Systems (SES); Broadband Satellite Multimedia (BSM); Services and Architectures”.

[4] 3GPP TR 22.891, “3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Feasibility Study on New Services and Markets Technology Enablers; Stage 1 (Release

14)", v1.0.0, September 2015.
Section 5.3
Section 5.20
Section 5.22
Section 5.72

[5] Ubuntu server guide, [Online]. Available: https://help.ubuntu.com/lts/serverguide/index.html