ANASTACIA

Advanced Networked Agents for Security and Trust Assessment in CPS/IoT Architectures

D6.5

Final use cases implementation and tests Report

This deliverable presents the results of ANASTACIA Task 6.5. The aim of the task is to define the deployment and operational aspects of the use cases in the frame of ANASTACIA, as well as to support the implementation and testing of the considered use cases.

Distribution level	CO
Contractual date	30.06.2017 [M09]
Delivery date	30.11.2019 [M35]
WP / Task	WP6
WP Leader	UTRC
Authors	Miloud Bagaa, Mohammed Boukhalfa (AALTO), Diego Rivera (MONT), Rafael Marín Pérez (ODINS), Jorge Bernal, Alejandro Molina (UMU), Dallal Belabed (THALES), Alie Mady, Satyarana Vuppala, Deepak Mehta, Piotr Sobonski (UTRC), Enrico Cambiaso, Ivan Vaccari, and Silvia Scaglione (CNR), Kristian Slavov, Nicklas Beijar (Ericsson)
EC Project Officer	Carmen Ifrim carmen.ifrim@ec.europa.eu
Project Coordinator	Softeco Sismat SpA Stefano Bianchi Via De Marini 1, 16149 Genova – Italy +39 0106026368 stefano.bianchi@softeco.it
Project website	www.anastacia-h2020.eu

ANASTACIA has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement N° 731558 and from the Swiss State Secretariat for Education, Research and Innovation

Table of contents

Index of figures

Index of tables

PUBLIC SUMMARY

Following Y2 demonstration and EU review feedback, as well as rewriting of the D.6.2 document, ANASTACIA team focused effort on delivering more complex scenario that will be showcased during Y3 review in University of Murcia in Spain (UMU). The consortium will use smart building scenario to demonstrate ANSTACIA framework capabilities to detect, alert and mitigate range of threats including showcasing new approach towards security by design principle with help of dynamic security and privacy seal components.

Modifications made to ANASTACIA architecture to enable better component integration, as well as scalability and resilience of the platform, were described in latest ANASTACIA architecture document (D1.5 ***[2]***). New integration effort was focused on smart building scenario. Flexibility of new ANASTACIA framework has been demonstrated by including use case demonstration based on technologies brought by 5G. In final year of the project, the consortium decided to move all infrastructure to final UMU demonstration site. All test cases listed in this document are measured on that infrastructure.

Finally updated end user questionnaire has been developed to help users better evaluate ANASTACIA framework. Final validation and test results report based on this work will be described in detail in D6.6 ***[10]***.

1 INTRODUCTION

1.1 AIMS OF THE DOCUMENT

Overall this document aims to provide clear description of test cases used in final ANASTACIA demonstration at UMU site. All test cases described in the report were tested in Murcia site. Reader should have in mind that all the work listed below is referring to final ANASTACIA framework demonstrator unless stated otherwise.

Changes applied to ANASTACIA test framework during final integration were illustrated in the chapter 2. Changes applied to ANASTACIA framework were described in D1.5 ***[2]***. Main bulk for the document is focused around smart building scenario and its test cases illustrated in chapter 3. Flexibility of ANASTACIA integration with new set of technologies such as 5G have been described in chapter 4 where network slicing scenario is used to protect user against malicious traffic. Moreover, new end user questionnaire was developed to get more insightful feedback from ANASTACIA end user during final validation phase of the framework. Final conclusions of the work are included in chapter 6. The questionnaire will be later used to validate ANASTACIA framework. Full description of this work can be found in D6.6 ***[10]***.

Annex section provides examples of ANASTACIA infrastructure components proof work captured during execution of test cases, as well as examples of component configurations, their screenshots and final results achieved during testing.

1.2 APPLICABLE AND REFERENCE DOCUMENTS

This document refers to following documents:

- D1.2 – "User-centered Requirement Initial Analysis" ***[1]***
- D1.5 – "Final Architectural Design" ***[2]***
- D2.2 – "Attacks and Threats Analysis and Contingency Actions" ***[3]***
- D2.8 – "Secure Software Development Guidelines Final Report" ***[4]***
- D4.4 – "Final Monitoring Component Services Implementation Report" ***[5]***
- D4.5 – "Final Reaction Component Services Implementation Report" ***[6]***
- D4.6 – "Final Agents Development Report" ***[7]***
- D6.2 – "Initial use cases implementation and tests Report" ***[8]***
- D6.4 – "Final Technical integration and validation Report" ***[9]***
- D6.6 – "Final End-user validation and evaluation Report" ***[10]***

Please note that at the time of writing this document some of the future deliverables weren't completed in accordance to ANASTACIA DoW in particular D1.5 ***[2]*** and D6.6 ***[10]***.

1.3 REVISION HISTORY

Version	Date	Author	Description
v0.1	03/07/2019	UTRC	Initial draft of the document
v0.2	25/10/2019	UTRC	ToC of the document
v0.3	31/10/2019	UTRC	New version of ToC aligned to ANASTACIA GA meeting outcomes

Version	Date	Author	Description
v0.4	13/11/2019	UTRC	Updated ToC with all test cases being tested in UMU
v0.5	20/11/2019	UTRC	UTRC input to chapters 1, 2, 3, and 6
v1.0	27/11/2019	All	CNR, OdinS, MONT, MAND/DG, Ericsson contributions
v1.1	28/11/2019	MAND	Adding internal review feedback by MAND
v1.2	28/11/2019	UTRC	Improving document based on feedback and release to EU

1.4 Acronyms and Definitions

Acronym	Meaning
AAA	Authentication Authorization and Accounting
AI	Artificial Intelligence
AMF	Access and Mobility Management Function
API	Application Programming Interface
APT	Advanced Persistent Threat
BMS	Building Management System
CAPEC	Common Attack Pattern Enumeration and Classification
CCTV	Close circuit television
CP	Constraint Programming
CSO	Chief Security Officer
DNS	Domain Name Service
DoS	Denial of Service attack
DDoS	Distributed Denial of Service attack
DPO	Data Protection Officer
ETSI	European Telecommunications Standards Institute
FTP	File Transfer Protocol
HVAC	Heat Ventilation Air Conditioning system

Acronym	Meaning
HTTP	Hyper Text Transport Protocol
ICT	Information and Communication Technologies
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IoT	Internet of Things
ISG	Industry Standardization Group
Kafka	Message broker used in ANASTACIA framework to enable distributed communication between components
KPIs	Key Performance Indicators
LDoS	Low-rate DoS
MEC	Mobile Edge Computing
MiTM	Man in the Middle attack
ML	Machine Learning
NFV	Network function virtualization
OVS	Open virtual switch
PHP	Hypertext Preprocessor
QoS	Quality of Service
REST	Representational State Transfer
SBA	Smart Building Automation
SDA	Slow DoS Attack
SDN	Software Defined Network
SEP	Security Enforcement Plane
SM	System Model (part of SO implementation that maintains SEP inventory)
SIEM	Security information and event management
SO	Security Orchestrator
SOAP	Simple Object Access Protocol

Acronym	Meaning
TC	Test-Case
TE	Test event
UC	Use-Case
UE	User Equipment
UMU	University of Murcia
VNF	Virtualized Network Functions
XML	eXtensible Markup Language

2 INTEGRATION FRAMEWORK FOR TEST CASES (UTRC)

This chapter contains description of latest update to test framework architecture used in ANASTACIA project. Changes made to the framework helped monitor test cases used in final ANASTACIA framework demonstration. Content below illustrates in detail all changes made to the framework in comparison to description placed in D6.2 ***[8]***.

2.1 TEST FRAMEWORK ARCHITECTURE

Figure 1 illustrates updated ANASTACIA test framework architecture used to validate integration between components in Y3 demonstration. Main purpose of updating test case framework was to enable better component cooperation and precise monitoring of external component KPIs generated during tests.

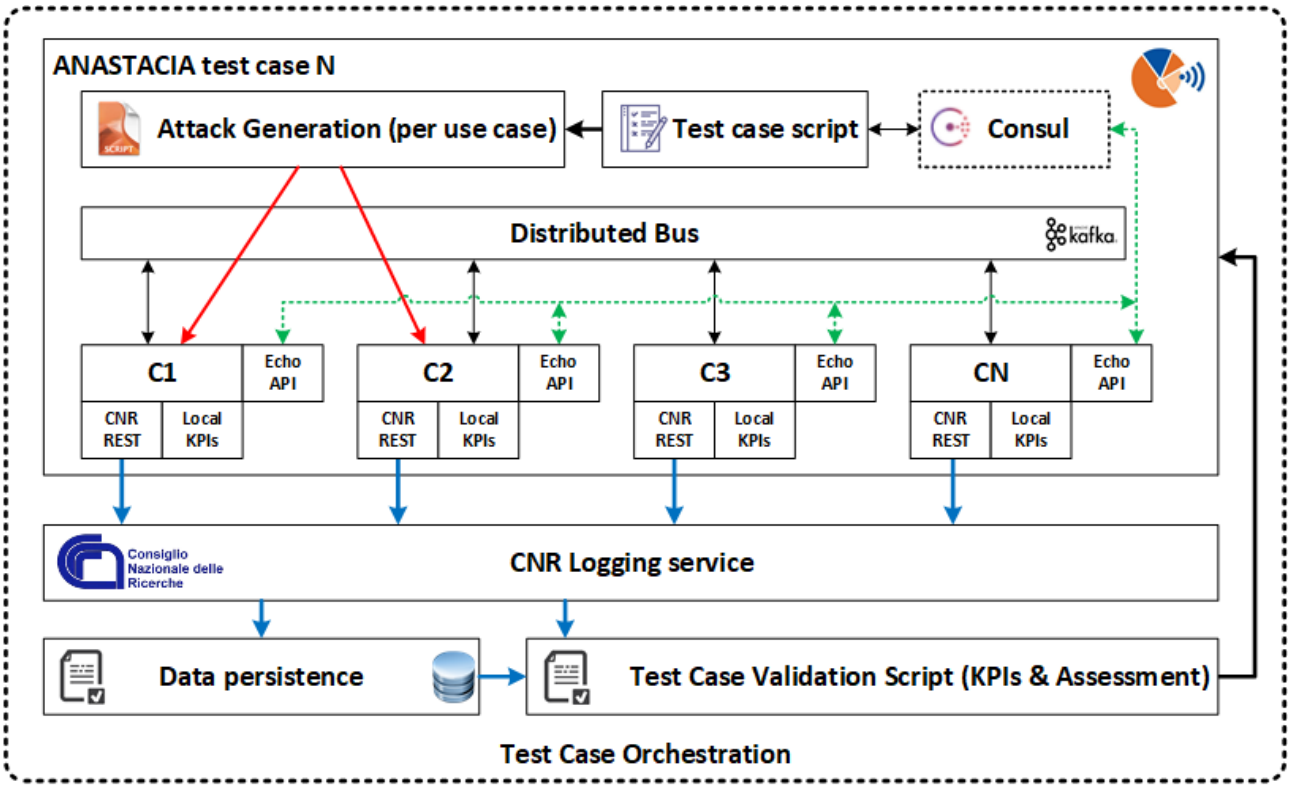

Figure 1. Test case architecture for ANASTACIA components.

Main changes to test framework include modifications made by all project partners involved in demonstration, following instrumentation changes at component level:

- **Consul interface** ***[13]*** – REST API service to enable health status check for each component. This is very simple REST API that allows Consul Service to query periodically ANASTACIA components about their status. Each component hosts REST API endpoint service that should reply depending on the component status code 200 for component online or 5XX error for notifying Consul Service about potential problem with the component. In the final implementation, all ANASTACIA components implemented this interface.
- **Internal KPI logging and measurement** – responsible for recording component performance. Each ANASTACIA component is recording their own KPIs in accordance to description placed in chapter 3 and 4. The results are stored locally and shared later with consortium during KPIs evaluation.
- **CNR REST API** – to register externally important actions made by ANASTACIA components from test case perspective. This is later used to help evaluate overall test scenario at each step of scenario execution by comparing log time stamps from each component.

In addition to instrumentation changes made at component level, CNR logging service ***[11]*** have been equipped with backend in form of a database to ensure that all logs are safely preserved for further KPI, performance evaluation.

Most of the ANASTACIA components except SO and parts of Ericsson 5G network slicing demonstration are packed into independent and scalable Docker ***[12]*** containers. This approach enables ANASTACIA to resolve challenges scalability and performance through component containerization. Each test case will have test script that checks requirements for particular components availability on Consul Service and triggers actions required to execute test. Usually the test sequence includes start up sequence and post-test wrap up to clear infrastructure for next test.

2.2 COMPONENTS

Table 1 lists new components added to ANASTACIA test architecture with their respective description and references in ANASTACIA documentation where more information can be found.

Table 1. List of new components added to ANASTACIA framework for test cases validation.

#	Component name	Description	Reference
1	**CNR logging service**	**Component designed to log ANASTACIA components external actions in order to assess test case performance.**	**ANASACIA GitLab repository – please check section 2.3**
2	**Consul**	**Server and agents installed by UBITECH to simplify ANASTACIA framework monitoring.**	**D6.4**

2.3 INTERFACES

As new components were added to test framework, new interfaces had to be developed to ensure correct flow of KPI measurement information between test infrastructure and ANASTACIA components. Table 2 lists interfaces used to incorporate new abilities of test framework.

Table 2. List of new interfaces added to ANASTACIA framework that will be used in test cases validation.

#	Interface name	Description	Reference/Link
1	**CNR_Log**	**CNR logging interface**	https://gitlab.com/anastacia-project/anastacia-logging-service
2	**Echo component API**	**Echo service example. This is example code that was adopted by ANASTACIA components to enable Consul service request component status.**	https://gitlab.com/anastacia-project/echo_service

2.4 SELECTED ADVANCED ATTACKS

In final demonstration of ANASTACIA framework, a selection of IoT based attacks were chosen to illustrate ANASTACIA ability to thwart the cybersecurity attacks and intrusions as well as showcase resilience, performance and scalability of the framework. In following sections, attacks used to test ANASTACIA framework will be described.

2.4.1 Slow DoS attack (CNR)

The innovative threat developed by CNR in the context of the project is a denial of service attack called SlowComm ***[14]***. It is a Long Request DoS attack ***[15]*** exploiting requests management on server side. As other Slow DoS Attacks, SlowComm makes use of a vulnerability on most server applications, which limit the number of simultaneous threads on the machine. Unlike flooding DoS attacks, which aim to overwhelm different network parameters of the victim host, slow DoS threats adopt a different approach, seizing all the available connections with the application listening daemon and exploiting specific timeouts ***[16]***. Although each connection includes application layer payload, compared to other categories of DoS attacks (like flooding), this approach requires a very limited amount of attack bandwidth, since the number of connections a server is able to manage simultaneously at the application layer is sensibly lower that the number it is able to manage at the transport layer.

Particularly, SlowComm sends a large amount of slow and potentially endless requests to the server, saturating the available connections at the application layer on the server inducing it to wait for the completion of the requests. Such completion is never triggered by the attacker. Referring for instance to the HTTP protocol, where the characters sequence \r\n\r\n represent the end of the request, SlowComm never sends these characters, hence forcing the server to an endless wait. Additionally, the request is sent abnormally slowly. Similar behavior could be adopted for other protocols as well (SMTP, FTP, SSH, etc.). As a consequence, by applying this behavior to a large amount of connections with the victim, a DoS may be reached.

The attack bandwidth requirements for the executed tests are often less than 1 KB/s. Such value should be considered extremely low, for a successful attack. Also, such minimum amount of bandwidth should highlight how dangerous an attack could be, since it may be particularly difficult to identify a running threat, considering that the malicious traffic may be part of a wide traffic dump.

Analyzing how the attack works, since the attacker's purpose is to seize, as soon as possible, all the available connections on the targeted host, we could assume that a DoS is reached some instants after the attack is launched. Nevertheless, the server may have already established active and legitimate connections with other clients. Those connections are working until they are closed. As soon as a connection closure happens, its relative resources on the server are released, thus allowing clients to establish new incoming connections. Because of this, the purpose of the attacker is to replace all the “just available” connections with malicious ones. While doing that, there could be a race condition between the attacker and some other legitimate clients. Nevertheless, we could assume that sooner or later the attacker would obtain the connections, since it would repeatedly try to connect to the victim with an intelligent algorithm, turning aggressiveness and stealthiness of the attack.

From the stealth perspective, the proposed attack is particularly difficult to detect while it is active, since log files on the server are often updated only when a complete request is received or a connection is closed: being our requests typically endless, during the attack log files do not contain any trace of attack. Therefore, a log analysis can’t produce an appropriate warning in a reasonable time. Nevertheless, anomaly-based systems may reveal the malicious behavior of the threat, for instance through approaches based on statistic ***[17, 18],*** machine learning ***[19, 20*** *and* ***21]***, Deep Learning ***[22]***, or spectral analysis ***[23]***.

In the context of ANASTACIA, the approach proposed in ***[24]*** is adopted to identify a running attack.

2.4.2 MiTM attack (UTRC)

MiTM attack is designed to change temperature on ANASTACIA SEP (Security Enforcement Plane) is emulating APT (Advanced Persistent Threat) behavior when potential attacker penetrated perimeter of interest and lay silently until called back from command and control center to commence malicious activity to further compromise physical or cyber protection of the smart building. The UTRC MiTM agent has been described in more detail in D4.6 ***[7]*** in section 3.2.1.

From attack scenario, when 'start' command is received by APT agent, it will perform following steps to perform attack:

1. Probe current state of the ANASTACIA SEP infrastructure to ensure quick removal of attack values if required. The values are stored internally inside agent.
2. Random selection of victims to attack from SEP IoT available sensor pool. The understanding is that agent already performed current environment mapping, so this knowledge is already acquired before attack.
3. Start of an attack by randomly changing values on selected sensors.
4. Iterate over sensor for requested amount time or until 'stop' or 'restore' command is issued to the agent.
5. In case of 'restore' command is issued, the APT will stop attack and restore last known data points to attacked sensors.
6. The agent tracks its own attack and after attack is completed, it will resume silent probing of ANASTAICA SEP to keep up to data its own SEP mapping to enable future attacks. In ANASTACIA Y3 demonstration, this step was used as automatic cleanup operation after scenario testing. However, in real live scenario, this step facilitates removing exfiltration by removing all possible forensic traces from infected system.

Those steps are just emulation of potential adversary, however it is important to note that similar actions were performed on industrial systems in the past in real scenarios (i.e. Stuxnet and other recent attacks ***[25]***).

2.4.3 Link Flooding Attacks (Thales)

The incessant growth of the Distributed Denial-of-Service (DDoS) attacks has arisen a lot of questions in the cyber-attack domain. The current mechanism of DDoS attacks is typically accomplished by flooding the targeted service by sending superfluous requests to the victim. This kind of DDoS attacks has already gotten a lot of attention from the research community. Recently, two new distributed Link-flooding attacks with high destruction potential have been introduced named the Coremelt and the Crossfire attacks. Unlike the traditional DDoS attacks, these two attacks isolate the victim from the rest of internet while the traffic is not sent to it. Moreover, these attacks are indistinguishable since the adversary keeps each per-flow rate, to flood the target network links, low for the Crossfire attack and only legitimate traffic is used for the Coremelt attack. The previous characteristics make these attacks undetectable by the current protection mechanisms in the routers or by intrusion detection systems (IDS). The concept of the attack is based on the fact that specific network links, the Target Links, lead to both the Decoy Servers and the Target Area. Therefore, attacker can employ bots to flood the Target Links by sending traffic only to the Decoy Servers. As a consequence, when the Target Links are flooded, the Target Area becomes unreachable from the rest of the Internet. Moreover, the flows generated by the bots have low-rate so that no current security solution can classify the traffic as malicious.

- Target area: Area containing chosen target servers, e.g., an organization, a city, a state, or a country.
- Target link: Network link selected for flooding.
- Decoy server: Publicly accessible servers surrounding the target area.

Lately some studies that deal with link-flooding attacks have been proposed ***[26, 27, 28*** and ***29]***. In the context of ANASTACIA, a new study was done during the year three that can detect the involved sources (the bots), based on learning mechanisms ***[30].*** The study has been introduced a new mechanism in order to recover from these link flooding attacks. This mechanism can be included into the resource planning module lately for a next stage of TRL. The proposed solution is a research paper, tested in lab environment, the solution is a node-based solution enabled by Software-Defined Networking (SDN) environment, characterized by its centralized view and its routing flexibility based on flows. In the proposed solution, the detection phase is done locally and at the controller level. Each local node (vSwitch) that notices a congested link sorts out the legitimate sources from the suspect ones, by monitoring the sources that will have suspicious behavior. Each local node keeps a local log that will be sent periodically to the controller. Thus, we exploit the centralized view of the SDN Controller to construct the global list of malicious and benign sources.

The mitigation of the attack during the detection phase is done by categorizing the different paths as safe paths or suspect paths. This categorization depends on how many demands from suspect sources the path is carrying. Consequently, this categorization leads to route the flows from the same source type on the same path type and hence some paths can be hidden from the malicious bots. Hence, the benign flows can be routed in these hidden paths, which mitigate the attack. Besides, we exploit the routing flexibility to load balance only the flows affecting the congested links, limiting the routing changes locally.

3 USE CASE - SMART BUILDING AUTOMATION (UTRC)

ANASTACIA project Y3 demonstration will be demonstrated in UMU premise as smart building scenario. The main scenario was described in first section below and contains nine main steps. Each step sequence is then described in consecutive sub sections with each sequence mark as atomic sequence to carefully trace interactions between components in ANASTACIA framework. There are total 72 test-cases described in this chapter that define order of the smart building automation scenario.

3.1 OVERVIEW

Figure 2 depicts ANASTACIA framework demonstration deployed to UMU. The idea behind the scene is to showcase real smart building implementation of ANASTACIA in attack and cyber-perimeter defense. The diagram illustrates multiple swimming lanes corresponding to each actor of the scenario. On top adversary actor is placed – his role is to penetrate smart building premise to get access to BMS data while keeping low profile during the attack. Physical and cyber infrastructure hosted by UMU is represented by physical rooms in which IoT sensor will be placed. Each red/yellow star represents attack being performed in the room. All of the IoT devices are being continuously monitored by ANASTACIA framework. Next lane represents ANASTACIA framework actions performed during scenario. Below approximate status of DSPS (Dynamic Security and Privacy Seal) is represented by seal color displayed on DSPS UI that DPO/CSO might observe. Next DPO/CSO preventive actions are listed. Most of them are just observations from autonomous ANASTACIA actions. Below each part of the demonstration have a leader (ANASTACIA project partner) responsible for scenario test steps, their performance and orchestration. Finally, at the bottom as well as on top, there are scenario step numbers with color marking at each step color of the DSPS seal.

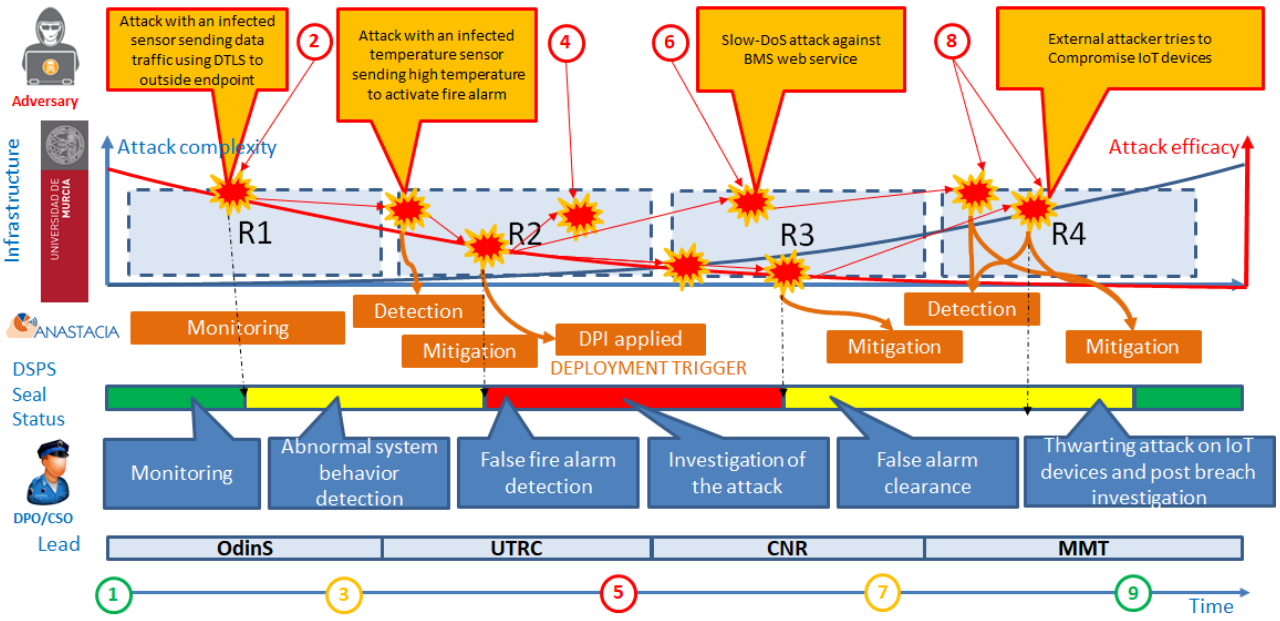

Figure 2. ANASTACIA Y3 smart building scenario demonstration deployed in UMU.

The real scenario will be playout in four rooms of computer department of UMU called for demonstration purpose R1-R4. Each step name will be associated with room number to help identify where cyber activities are being performed. All scenario steps are divided into test cases numbers with TC abbreviation and number for each sub-step in TC. All details regarding each step and test cases definition are listed below.

3.2 TEST-CASE R1 – ATTACK WITH AN INFECTED SENSOR SENDING DATA TRAFFIC USING DTLS TO OUTSIDE ENDPOINT (ODINS, UMU)

This test-case focuses on testing the first phase of Smart Building Use Case in the Room1 about the attack with an infected sensor sending data traffic using DTLS connection to an outside endpoint. The main objective of this phase R1 is to evaluate ANASTACIA framework protecting the system against an insider attack in an infected sensor sending sensitive information outside of the smart building environment. In this phase, the attacker exploits the sensors deployed in building operation workspace to request the activation of malicious traffic generated by an infected device. Hence, this section identifies a set of test-cases to implement and evaluate the phase R1 on Smart Building testbed.

The next subsections present a set of test-cases grouped in 3 main steps to evaluate the interaction among components of ANASTACIA framework grouped in different modules such as IoT Infrastructure, Monitoring Module, Reaction Module, Seal Manager, Orchestration Plane and Control Domain.

3.2.1 Step 1 – R1

In this subsection, we describe the following test-cases involved in the step 1 (Figure 3) for the phase R1 of an infected sensor used for sending data to an outside endpoint in order to show ANASTACIA framework for Monitoring and Proactive Security Policies related to the deployment of IoT devices in smart building:

- TC1.1 MMT Solution is monitoring the 6LowPAN Network. (MONT)
- TC1.2 DSPS is GREEN. (AS/DG/MI)
- TC1.3 Proactive Security Policies definition: (UMU)
    - All traffic is denied by default.
    - Allow AuthN traffic between IoT devices and AuthN agent.
    - Allow DTLS+CoAP traffic from IoT Controller to IoT devices.
    - Authorize PUT/POST DTLS/HTTPS operations from IoT devices to BMS.
    - Allow DTLS/HTTPS traffic from IoT devices to BMS.
- TC1.4 Proactive Security Policies enforcement (AALTO)
- TC1.5 Inclusion of the malicious IoT device (ODINS)
- TC1.6 Bootstrapping process (ODINS/UMU)
- TC1.7 Reactive Security Policies enforcement (AALTO)

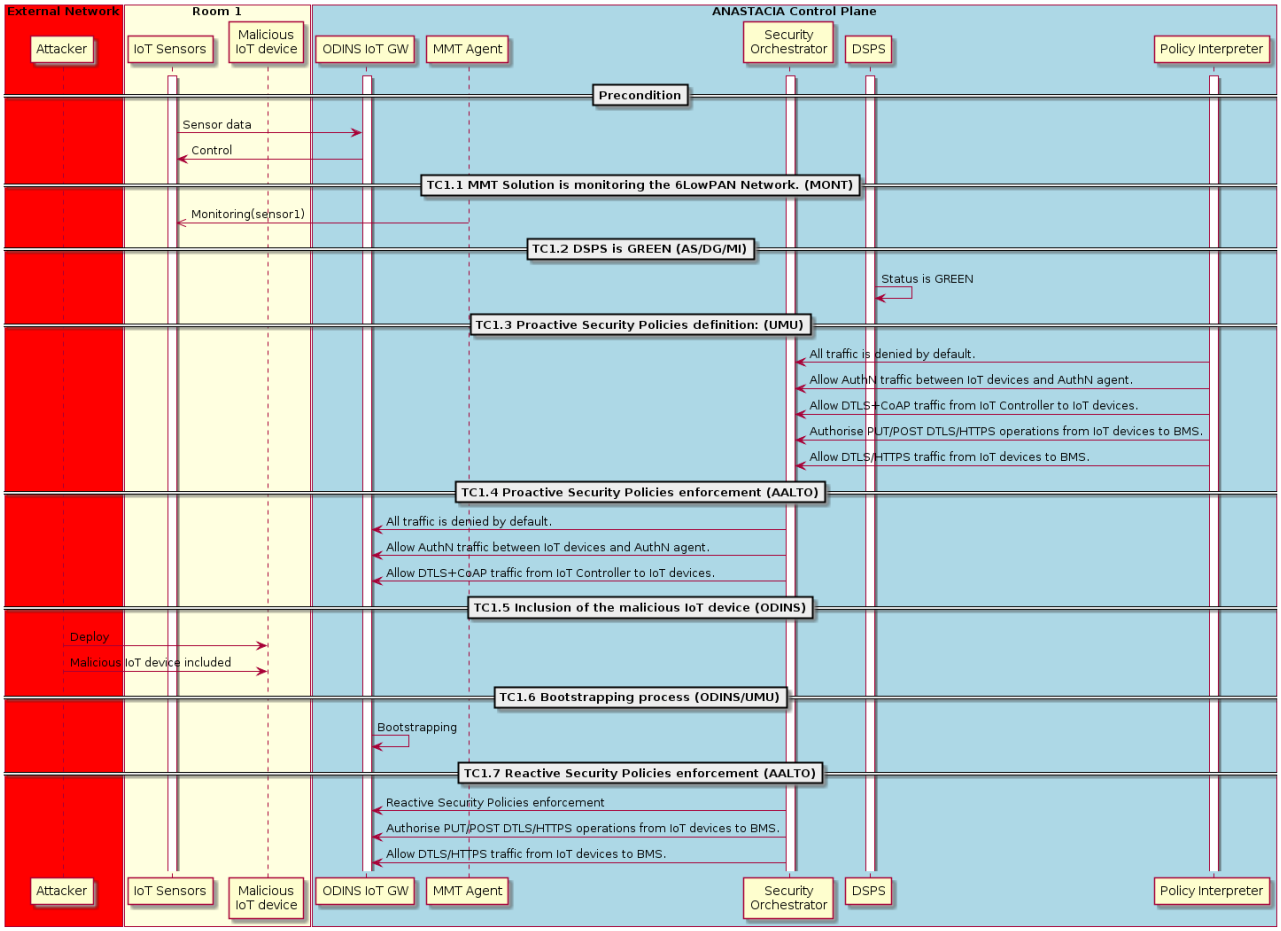

Figure 3. Step R1 test scenario combined with all test cases.

3.2.1.1 TC1.1 MMT Solution is monitoring the 6LowPAN Network (MONT)

TC1.1 MMT Solution is monitoring the 6LowPAN Network. (MONT)	
Preconditions	• MMT-IoT Sniffer is physically installed near the IoT devices to monitor. • MMT-Probe machine is located in the IoT network. • MMT-IoT Sniffer is connected to the MMT-Probe machine using a USB line.
Components	• IoT devices • MMT-IoT Sniffer • MMT-Probe
Execution	• The MMT Probe software is started on the MMT-Probe machine. • The MMT-IoT Bridge tool is started on the MMT-Probe machine. • Send the activation command to the MMT-IoT Sniffer by using the CLI of MMT-IoT Bridge.
Expected results	• The MMT-IoT Sniffer starts extracting the traffic. • The MMT-Probe software starts analysing the extracted traffic. • MMT-Probe generates statistical reports about the analysed traffic.
Expected completion	• Month 35 – November 2019

KPI(s)	• Monitoring is enabled within 500msec
Fail criteria	• The MMT-IoT sniffer does not extract IoT traffic. • MMT-Probe does not generate statistical reports about the extracted traffic.

3.2.1.2 TC1.2 DSPS seal changing status to GREEN (AS/DG/MI)

TC1.2 DSPS seal changing status to GREEN (AS/DG/MI)	

Preconditions	• DSPS components are initiated:
	  ○ DSPS Seal Creation Service has empty cache
	  ○ DSPS GUI backend has empty cache
	  ○ No previous states kept in Secured Storage

Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Components are started
Expected results	• Security part of **seal** is GREEN
Expected completion	• Month 35 – November 2019
KPI(s)	• None
Fail criteria	• Security part of **seal** is not GREEN

3.2.1.3 TC1.3 Proactive Security Policies definition (UMU)

TC1.3 Proactive Security Policies definition: (UMU)	
Preconditions	• Policy Editor Tool is up and running • Policy Interpreter is up and running • Policy Conflict detector is up and running • Policy Repository is up and running • System model is up, running and correctly updated. • Security Enablers Provider is up and running.
Components	• Policy Editor Tool • Policy Interpreter • Policy Conflict Detector • Policy Repository • Security Enablers Provider
Execution	• HSPL Orchestration policy is defined by using the Policy Editor Tool in order to: ○ Allow DTLS communication between IoT Controller and IoT devices

	○ Allow AuthN traffic between Authentication Agent and IoT devices. ○ Authorise devices to PUT temperature in IoT broker. Include on AuthN success dependency. ○ Allow DTLS traffic between IoT devices and IoT broker. Include on AuthN success dependency. • Press “Refinement” button in order to refine the HSPL-OP. • Press “Modify” button and add a security policy in order to deny AuthN traffic with the same priority. • Press “Refinement” button and verify the enforcement is not allowed. • Press “Modify” and remove the conflictive policy. • Press “Refinement” in order to refine the HSPL-OP by using the Policy Editor Tool. • Press “Enforcement” in order to request the MSPL-OP enforcement. • Advanced Proactive enforcement request by using directly an MSPL file in order to allow HTTPS between powerful IoT devices and end-point service.
Expected results	• MSPL-OP refinements including event dependencies detection. • MSPL-OP refinements including conflict detection. • MSPL-OP enforcement request by using Policy Editor Tool. • MSPL-OP enforcement request by using command line.
Expected completion	• Month 35 – November 2019
KPI(s)	• Policy refinement times. • Conflict detection times.
Fail criteria	• Policy Refinement fails due absence of required enablers in security enablers provider. • Policy Refinement fails due unexpected parameters or absence of required values in system model. • Conflict detection fails due unexpected parameters or absence of required values in system model.

3.2.1.4 TC1.4 Proactive Security Policies enforcement (AALTO)

TC1.4 Proactive Security Policies enforcement (AALTO)	
Preconditions	• Policy Interpreter is up and running. • Policy Conflict detector is up and running. • System model is up, running and correctly updated. • Security Enablers Provider is up and running. • SDN Controller is up and running.
Components	• Security Orchestrator • Policy Interpreter • Security Enablers Provider
Execution	• Receiving the MSPL file through the REST API of the security orchestrator. The MSPL file consists of: ○ Allow DTLS communication between IoT Controller and IoT devices. ○ Allow AuthN traffic between Authentication Agent and IoT devices. ○ Authorise devices to PUT temperature in IoT broker. Include on AuthN success dependency.

	○ Allow DTLS traffic between IoT devices and IoT broker. Include on AuthN success dependency. • The Security Orchestrator communicates with Security Enablers Provider. • The Security Enablers Provider returns ONOS as enabler. • The Security Orchestrator communicates with the Policy Interpreter to get the low configuration from the MSPL file. • The Security Orchestrator enforces the open flow rules by communicating with the REST API of ONOS controller.
Expected results	• Translating the received MSPL files to the low configuration. • Generating the required open flow rules from the received configuration. • Enforcing the open flow rules through ONOS REST API.
Expected completion	• Month 35 – November 2019
KPI(s)	• Required time to translate the MSPL files to low configuration. • Required time to generate the open flow rules and enforcing them.
Fail criteria	• Security Orchestrator fails due absence of required enablers in security enablers provider. • Security Orchestrator fails due unexpected parameters or absence of required values in Policy Interpreter. • Conflict detection fails due unexpected parameters or absence of required values in system model.

3.2.1.5 TC1.5 Inclusion of the malicious IoT device (ODINS)

TC1.5 Inclusion of the malicious IoT device (ODINS)	
Preconditions	• Script is ready to emulate the attack query. • IoT device is working – DTLS + COAP API is available online.
Components	• Malicious Script • IoT Device
Execution	• Malicious Script is launched. • IoT device receives a DTLS+COAP query to modify the internal configuration with a new IP destination address to send the information collected. • IoT device accepts the DTLS+COAP query to change the IP address of the destination endpoint.
Expected results	• IoT device is restarted with the new configuration in order to send the data messages to the new IP destination address outside the smart building network.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time for processing the received query and modified the IP destination address in the internal configuration.
Fail criteria	• Exception caught during the query reception

	• Excessive time for the query processing

3.2.1.6 TC1.6 Bootstrapping process (ODINS/UMU)

TC1.6 Bootstrapping process (ODINS/UMU)	
Preconditions	• PANA Agent is ready to authenticate new devices in IoT network. • IoT Register is ready to receive publication of new devices in IoT network.
Components	• IoT Device • PANA Agent • IoT Register
Execution	• IoT device is restarted in order to use the new configuration and start the bootstrapping process • IoT device sends a query for network access to PANA Agent • PANA Agent evaluates if the identity of new device is valid or invalid • If it is valid, PANA Agent sends a publication of new device to IoT Register
Expected results	• Network authentication is successful • New device publication is successful
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the response time for network authentication between IoT device and PANA Agent • Obtain the response time for new device publication between PANA Agent and IoT Register
Fail criteria	• Excessive time for network authentication. • Excessive time for new device publication.

3.2.1.7 TC1.7 Reactive Security Policies enforcement (AALTO)

TC1.7 On AuthN success dependency Security Policies enforcement (AALTO)	
Preconditions	• IoT Register is subscribed to Authentication agent. • Security Orchestrator is ready to process MSPL-OP policies. • Policy Interpreter, Conflict detector and Policy repository are up and running. • Security Enablers Provider is up and running. • System Model is up and running. • SDN Controller is up and running.
Components	• IoT Register. • Security Orchestrator. • Policy Interpreter. • Security Enablers Provider.
Execution	• IoT Register receives IoT register notification. • IoT Register registers the IoT device in the system model. • IoT Register notifies the event to the Security Orchestrator. • Security Orchestrator verifies policies with pending event dependencies: ○ Authorise PUT temperature in IoT Broker.

	○ Allow DTLS traffic between IoT device and IoT broker. • Security Orchestrator communicates with the Security Enablers Provider for selecting the best enabler. • The Security Enablers Provider selects and returns ONOS as enabler to the Security Orchestrator. • Security Orchestrator requests policies translation to policy Interpreter. • Policy Interpreter verifies policy conflicts or dependencies by using the conflict detector. • Policy Interpreter retrieves enabler plugins from the Security Enabler Provider and translates the MSPL-OP into final configurations. • Policy Interpreter returns final configurations and conflicts and dependencies detection to Security Orchestrator. • Security Orchestrator generates the open flow rules from the received configuration. • Security enforces the generated open flow rules through the REST API of ONOS. • Security Orchestrator updates the System Model by adding the status about the enforced MSPL file.
Expected results	• IoT registration notification • Authorisation configurations • IoT devices are able to communicate with the IoT Broker after the authentication.
Expected completion	• Month 35 – November of 2019
KPI(s)	• Policy translation times. • Conflict detection times. • Required time to translate the MSPL files to low configuration. • Required time to generate the open flow rules and enforcing them.
Fail criteria	• Policy translation fails due absence of required enablers in security enablers provider. • Policy translation fails due to unexpected parameters or absence of required values in system model. • Conflict detection fails due to unexpected parameters or absence of required values in system model. • Security Orchestrator fails due to unexpected parameters or absence of required values in the Policy Interpreter.

3.2.2 Step 2 – R1

In this subsection, we describe the following test-cases involved in the step 2 (Figure 4) for the phase R1 of an infected sensor for sending data to an outside endpoint in order to show ANASTACIA framework for Attack Detection, Reaction and Seal Management related to the detection of infected IoT device in smart building:

- TC2.1 IoT device tries to establish a DTLS connection to outside. (ODINS)
- TC2.2 MMT solution detects invalid destination at 6lowPAN level. (MONT)
  - MMT solution notifies the issue.
- TC2.3 Incident detector-VDSS processes the notification and generates alert and mitigations list. (ATOS)
- TC2.4 The SAS forwards alert to DSPS. (CNR)
- TC2.5 DSPS changes to YELLOW. (AS/DG/MI)
- TC2.6 The MAS consume list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS and the VDSS. (MMT)
- TC2.7 The SAS forwards mitigation action (mitigating) to DSPS. (CNR)

- TC2.8 DSPS maintains YELLOW state. (AS/DG/MI)
- TC2.9 SAS keeps polling SO system model. (CNR)

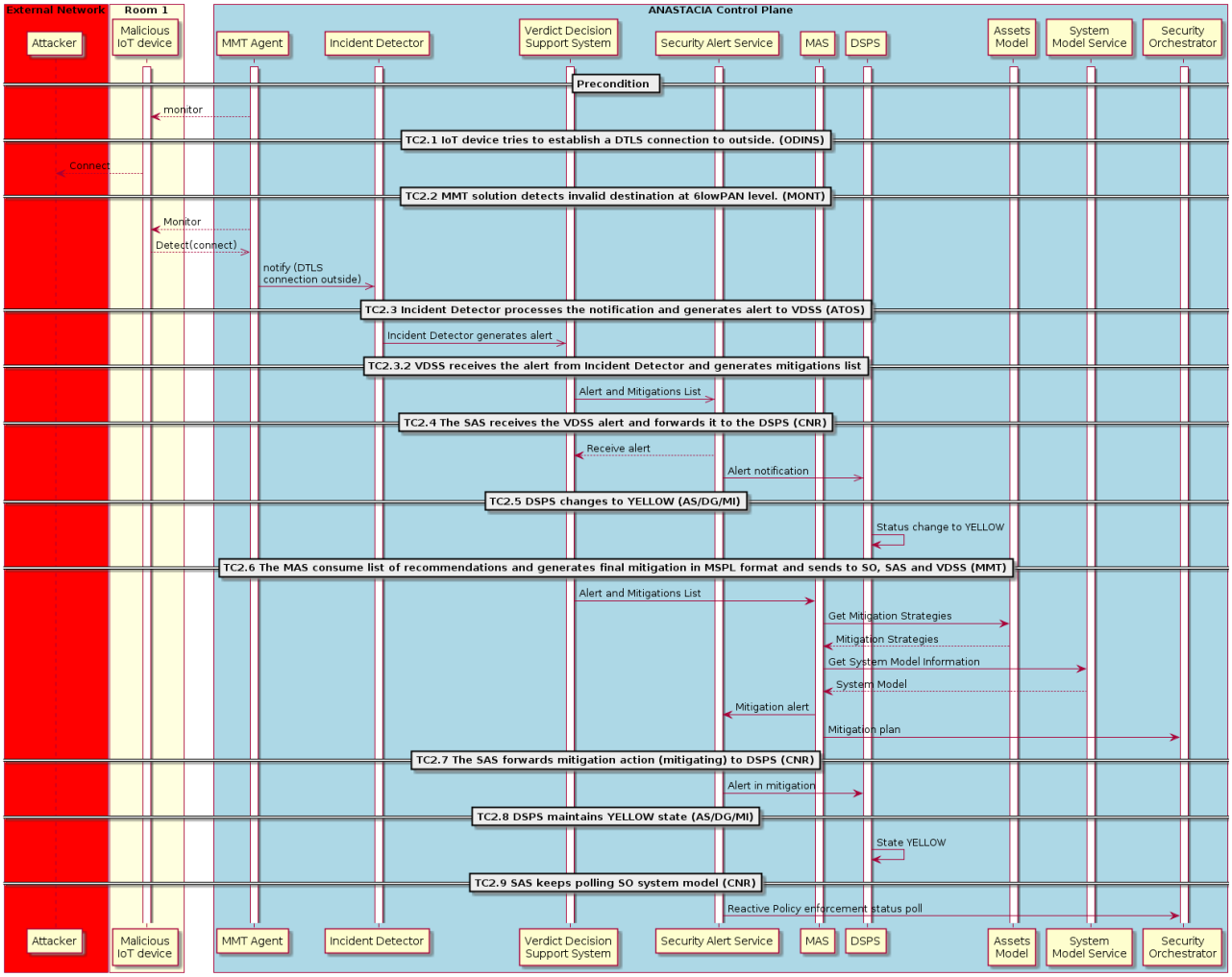

Figure 4. Step R2 test scenario combined with all test cases.

3.2.2.1 TC2.1 IoT device tries to establish a DTLS connection to outside (ODINS)

TC2.1 IoT device tries to establish a DTLS connection to outside. (ODINS)	
Preconditions	• IoT device has been infected with a new configuration for sending information collected and has done the bootstrapping process to start sending data.
Components	• IoT Device
Execution	• IoT Device starts the DTLS connection to an IP destination address outside the smart building network in order to send the data messages using DTLS+COAP protocols.
Expected results	• DTLS traffic is transmitting to an IP destination address outside the smart building network.
Expected completion	• Month 35 – November 2019

KPI(s)	• Obtain the elapsed time for establishing the DTLS connection to the IP destination address outside.
Fail criteria	• Exception caught during the DTLS connection. • Excessive time for the DTLS connection.

3.2.2.2 TC2.2 MMT solution detects invalid destination at 6lowPAN level (MONT)

TC2.2 MMT solution detects invalid destination at 6lowPAN level. (MONT)	
Preconditions	• Script is ready to launch a connection to a server outside the IoT network. • MMT-IoT Sniffer is connected and actively sniffing IoT traffic • MMT-Probe is running and actively monitoring the extracted IoT traffic.
Components	• Malicious script and device • Monitoring Agent (MMT-IoT Sniffer and MMT-Probe)
Execution	• The script is launched. • The malicious device tries to open a connection to a server outside the trusted IoT network. • MMT-IoT Sniffer extracts the IoT traffic form the network and redirects it to MMT-Probe. • MMT-Probe analyses the traffic according to the loaded security rules.
Expected results	• MMT-Probe detects the attempt of an external connection and raises an alert about the issue.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time between the trigger of the attack and the detection.
Fail criteria	• The connection is not detected correctly (false negative). • A legit connection (towards a trusted server) is detected as an attack (false positive).

3.2.2.3 TC2.3 Incident detector-VDSS process the notification and generates alert and mitigations list (ATOS)

TC2.3 Incident detector – process the notification and generates alert (ATOS)	
Preconditions	• The MMT probe generates a “Untrusted destination detected” when monitoring the IoT devices
Components	• Security sensors • Data Filtering and pre-processing broker • Incident Detector
Execution	• Raw “Untrusted destination detected” events are submitted by the MMT probe to the Data Filtering and pre-processing broker

	• The storm topology of the Data Filtering and pre-processing broker process and sends the events through rsyslog to the Incident Detector • The Incident Detector receives the events through its agent, which normalizes them to a common format and submits it to the correlator engine of the Incident Detector, which processes it and generates a "Potential data leakage detected" alert • The alert is pushed to a the RabbitMQ queue, exchange.alarms, to be consumed by DSPS and VDSS
Expected results	• Alerts triggered based on the events received from the monitoring infrastructure
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the processing time in the Incident Detector to normalize the event, correlate it, generate the alert and send it to the RabbitMQ exchange
Fail criteria	• When an attack is performed, and no alert is generated

3.2.2.4 TC2.4 The SAS forwards alert to DSPS (CNR)

TC2.4 The SAS forwards alert to DSPS (CNR)	
Preconditions	• A new threat is detected by the Monitoring component • The VDSS sends the alert related to the threat to the SAS, through a dedicate RabbitMQ queue
Components	• VDSS, Verdicts and Decision Support System • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• SAS consume the RabbitMQ queue by retrieving the generated alert • The SAS forwards the alert, as is, to another RabbitMQ server shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the alert from the RabbitMQ queue dedicated to VDSS-to-SAS communication and the sending of the (same) alert to the RabbitMQ queue dedicated to SAS-to-DSPS communication
Fail criteria	• The VDSS doesn't push the alert inside the RabbitMQ queue

3.2.2.5 TC2.5 DSPS changes to YELLOW (AS/DG/MI)

TC2.5 DSPS changes to YELLOW (AS/DG/MI)

Preconditions	• Security part of **seal** is GREEN
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• New alert is received by DSPS (AMQP queue consumption) • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security part of **seal** changes to YELLOW • New seal is generated, stored in DSPS secured storage and pushed to GUI
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until stored in DSPS secured storage • The time measurement since DSPS seal creation receives alert until DSPS GUI is notified of the new seal
Fail criteria	• changes to RED

3.2.2.6 TC2.6 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS and the VDSS (MMT)

TC2.6 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)	
Preconditions	• There is an alert on the VDSS RabbitMQ that contains the mitigations recommendations.
Components	• Verdict and Decision Support System (VDSS): RabbitMQ endpoint • Mitigation Action Service (MAS) • Assets Model (AM) • System Model Service (SMS) • Security Alert Service (SAS) • Security Orchestrator (SO)
Execution	• An alert is published by the VDSS in the RabbitMQ channel. • The MAS receives the alert and starts the mitigation strategy processing. • The MAS contacts the auxiliary services to retrieve additional data to compute the MSPL (AM and SMS). • The MAS generates the MSPL and sends it to the SO.

	• The MAS sends the computed mitigation to the SAS and the VDSS.
Expected results	• The MAS generates an MSPL policy for orchestration that contains both an L4 Filtering security capability and a Data Analysis security capability.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the general time elapsed to compute the mitigation: starting from the reception of the alert until the MSPL is sent to the SO.
Fail criteria	• The MAS does not generate a MSPL policy for orchestration. • The MAS generates a policy that does not validate against the MSPL XSD schema. • The MAS generates an MSPL policy that does not contain the aforementioned security capabilities.

3.2.2.7 TC2.7 The SAS forwards mitigation action (mitigating) to DSPS (CNR)

TC2.7 The SAS forwards mitigation action (mitigating) to DSPS (CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS generates MSPL data related to the alert
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• The MAS sends to the SAS the MSPL file, related to the given alert, previously received by the VDSS • The SAS encapsulates received data inside the previously received alert; in particular, the following attributes are added to the original alert • The SAS forward the enriched alert to the RabbitMQ server, shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the MSPL data from the MAS and the sending of the enriched alert to the RabbitMQ queue
Fail criteria	• The MAS fails to generate or send MSPL data to the SAS

3.2.2.8 TC2.8 DSPS maintains YELLOW state (AS/DG/MI)

TC2.8 DSPS maintains YELLOW state (AS/DG/MI)	
Preconditions	• Security part of seal is YELLOW

Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert with mitigation action is received (mitigating) • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security part of seal is still YELLOW
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until processed
Fail criteria	• Security part of **seal** is not YELLOW

3.2.2.9 TC2.9 SAS keeps polling SO system model (CNR)

TC2.9 SAS keeps polling SO system model(CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS sends to the SAS the MSPL data related to the alert • The SM receives MSPL information from the MAS
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • SM, System Model
Execution	• The SAS polls the SM for a specific deployment identifier, previously received by the MAS
Expected results	• The SAS retrieves from the SM deployment information related to the alert countermeasures
Expected completion	• Month 35 – November 2019
KPI(s)	• Measure the time passing between the reception of the message from the MAS and the first polling to the SM
Fail criteria	• Communication with the SM fails

3.2.3 Step 3 – R1

Following action performed in step 2, additional counter-measure is being deployed from of UTRC Data Analysis Agent in step 3. The deployment is dynamic in nature and SO will reconfigure agent to start generating verdicts for a particular set of sensors where potential malicious activity was detected in step 2. This is an additional step in which ANASTACIA framework will validate monitoring with second agent to ensure high degree of fidelity when triggering an alarm which later will be sent to VDSS, SAS, MAS and SO. The test (Figure 5) case is constructed from following steps:

- TC3.1 Reactive policy for orchestration enforcement (AALTO)
  - Orchestrator decides to reconfigure the UTRC agent (AALTO/THALES)
- TC3.2 UTRC agent enables behavioral analysis on affected device and sends ACK (UTRC)
- TC3.3 SO updates system model (AALTO)
- TC3.4 SAS polls SO system model and sends to DSPS the alert with mitigated=True
- TC3.5 DSPS changes security status GREEN state (AS/DG/MI)

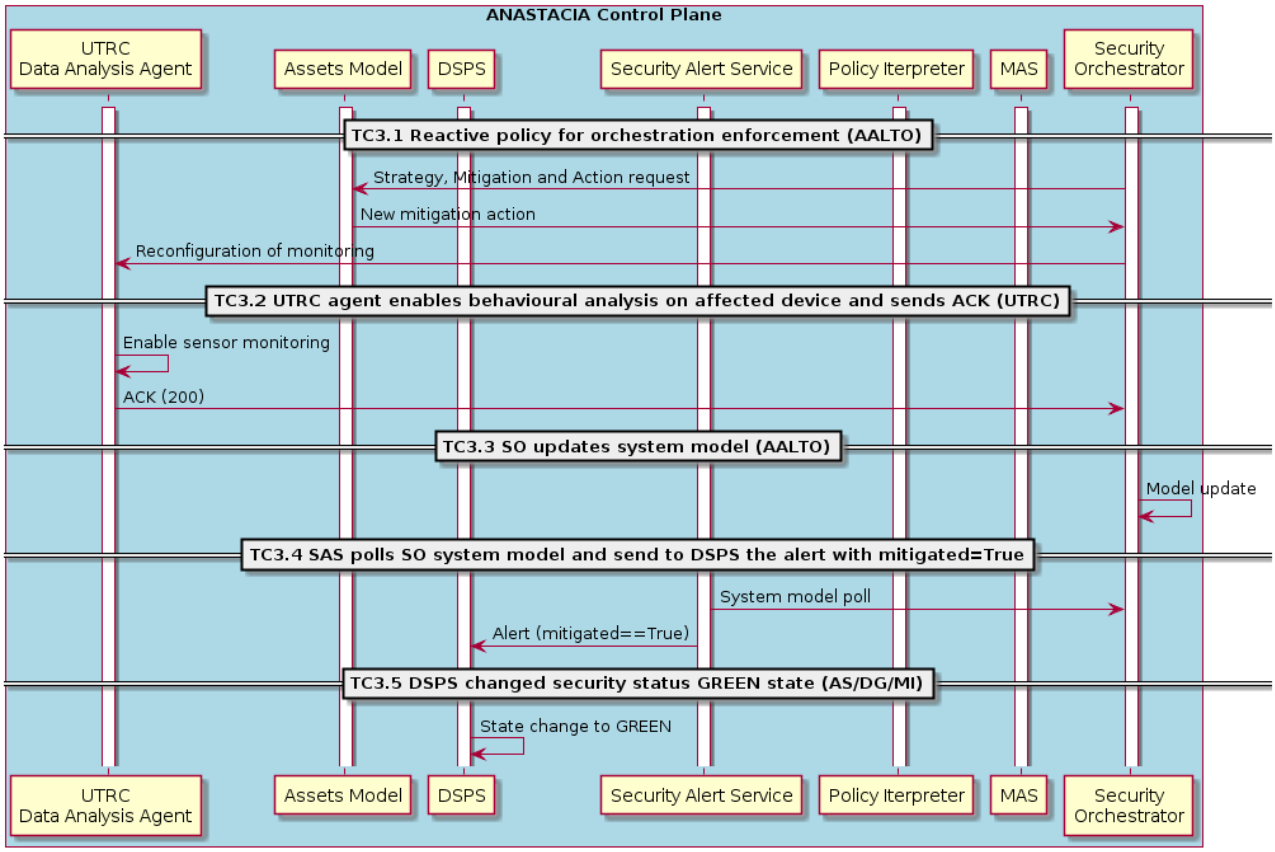

Figure 5. Step R3 test scenario combined with all test cases.

3.2.3.1 TC3.1 Reactive policy for orchestration enforcement (UMU/AALTO/THALES)

TC3.1 Reactive policy for orchestration enforcement (UMU/AALTO/THALES)	
Preconditions	• Security Orchestrator is ready to process MSPL-OP policies • Policy Interpreter, Conflict detector and Policy repository are up and running • Security Enablers Provider is up and running. • System model is up and running.

	• Policy Interpreter is up and running. • SDN Controller is up and running.
Components	• Security Orchestrator • Policy Interpreter • Security Enablers Provider

Execution	• Security Orchestrator verifies MSPL-OP policy which contains:
	  ○ Filtering
	  ○ Monitoring IoT data.
	• Security Orchestrator communicates with the Security Enablers Provider for selecting the best enabler.
	• Using the Security Enablers Provider, the Security Orchestrator decides to use ONOS as enabler for filtering and UTRC agent for data analysis.
	• Security Orchestrator requests policies translation to policy Interpreter.
	• Policy Interpreter verifies policy conflicts or dependencies by using the conflict detector.
	• Policy interpreter retrieves enabler plugins from the Security Enabler Provider and translates the MSPL-OP into final configurations.
	• Policy Interpreter returns final configurations and conflicts and dependencies detection to Security Orchestrator.
	• Security Orchestrator generates the open flow rules from the received configuration.
	• Security enforces the generated open flow rules through the REST API of ONOS to filter the traffic and enables the communication between the IoT device and the UTRC agent.
	• Security Orchestrator configures the UTRC agent through its offered REST API.

Expected results	• Filtering and Monitoring configurations • Malicious IoT device traffic is properly filtered and the UTRC agent is properly configured.
Expected completion	• Month 35 – November of 2019
KPI(s)	• Policy translation times. • Conflict detection times. • Required time to translate the MSPL files to low configuration. • Required time to generate the open flow rules and enforcing them. • Requires time to configure the UTRC agent.
Fail criteria	• Policy translation fails due absence of required enablers in security enablers provider. • Policy translation fails due unexpected parameters or absence of required values in system model. • Conflict detection fails due unexpected parameters or absence of required values in system model. • Security Orchestrator fails due to unexpected parameters or absence of required values in the Policy Interpreter. • Security Orchestrator fails due to unexpected parameters or absence of required values in the system model.

3.2.3.2 TC3.2 UTRC agent enables behavioural analysis on affected device and sends ACK (UTRC)

TC3.2 UTRC agent enables behavioural analysis on affected device and sends ACK (UTRC)

Preconditions	• ANASTACIA framework up and running • ANASTACIA framework detects first wave of anomalous behaviour
Components	• MAS • SO • Policy Interpreter • VDSS • OdinS IoT GW
Execution	• Agent continuously receives data from Kafka broker on topic “IoTBrokerTopic” • Policy Interpreter sends information about new MSPL to SO • SO sends new MSPL to Policy Interpreter • Policy Interpreter sends new configuration to UTRC Data Analysis Agent
Expected results	• UTRC Data Analysis Agent turns on data analysis on a given sensor(s) • Verdicts are sent to Kafka broker to topic called “UTRCVerdicts” • The verdict information will be sent to VDSS with information about analysis outcome using Kafka broker. The message is being sent to “UTRCVerdicts” topic. • VDSS receives successfully new verdicts from “UTRCVerdicts” topic
Expected completion	• Month 35 – November 2019
KPI(s)	• The verdict is generated immediately after new sample of data is received from IoT temperature sensor • Verdict generation per single data point is less than 300 msec
Fail criteria	• OdinS IoT GW stops working • IoT temperature sensor data is not being received via Kafka broker topic “IoTBrokerTopic” • Agent is not generating verdict messages for given IoT sensor • VDSS is not receiving verdict information on “UTRCVerdicts” topic • Kafka broker stops working

3.2.3.3 TC3.3 SO updates system model (AALTO)

TC3.3 SO updates system model (AALTO)	
Preconditions	• Security Enablers Provider is up and running • Security Orchestrator received the MSPLs translations • Security Orchestrator succeeded/failed to enforce the enablers configuration
Components	• System model. • Security Orchestrator.
Execution	• Security Orchestrator sends post request to update the enforcement API status
Expected results	• The system model updates the enforcement status
Expected completion	• Month 35 – November 2019
KPI(s)	• Measure the needed time for retrieving and updating the information.

Fail criteria	• Communication with the SM fails

3.2.3.4 TC3.4 SAS polls SO system model and send to DSPS the alert with enabled mitigation (CNR)

TC3.4 SAS polls SO system model and send to DSPS the alert with enabled mitigation	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS sends to the SAS the MSPL data related to the alert • The SM receives MSPL information from the MAS
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • SM, System Model
Execution	• The SAS polls the SM for a specific deployment identifier, previously received by the MAS • The SM answers with deployment data • The SAS extracts relevant fields from the data received by the SM • The SAS enriches (again) the alert to include relevant information received from the SM • The SAS forwards the enriched alert to the DSPS, through the dedicated RabbitMQ queue
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time spent between the reception of the (latest/valid) message from the SM and the sending of the enriched alert to the DSPS
Fail criteria	• The SM fails to send required data to the SAS

3.2.3.5 TC3.5 DSPS changed security status GREEN state (AS/DG/MI)

TC3.5 DSPS changed security status GREEN state (AS/DG/MI)	
Preconditions	• Security part of seal is YELLOW
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert with mitigated flag/information is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal

Expected results	• Security part of seal changes to GREEN
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until stored in DSPS secured storage • The time measurement since DSPS seal creation receives alert until DSPS GUI is notified of the new seal
Fail criteria	• Security part of seal doesn't change to GREEN • Seal is not stored in DSPS secured storage • DSPS GUI is not notified about new seal

3.3 Test-Case R2 – Attack with an infected temperature sensor (UTRC, OdinS)

3.3.1 Step 4 – R2

In this subsection, we describe the following test-cases involved in the step 4 for the phase R2 of an infected temperature sensor sending malicious data to the Building Management System in order to show ANASTACIA framework for Data Monitoring, Alert Detection, Reaction and Seal Management related to high temperature values generated by an IoT device in smart building (Figure 6):

- TC4.1 Misbehavior script/IoT: (ODINS)
  - IoT devices/script start sending wrong temperature values.
  - Fire alarm is turned on
- TC4.2 UTRC agent detects the issue and sends alarm to Incident Detector. (UTRC)
- TC4.3 Incident detector – process the notification and generates alert and mitigations list (ATOS)
- TC4.4 The SAS forwards alert to DSPS. (Alert level between 8 and 10) (CNR)
- TC4.5 DSPS changes to RED (AS/DG/MI)
- TC4.6 VDSS generates a list of recommended mitigations (ATOS)
- TC4.7 The MAS consume list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)
- TC4.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)

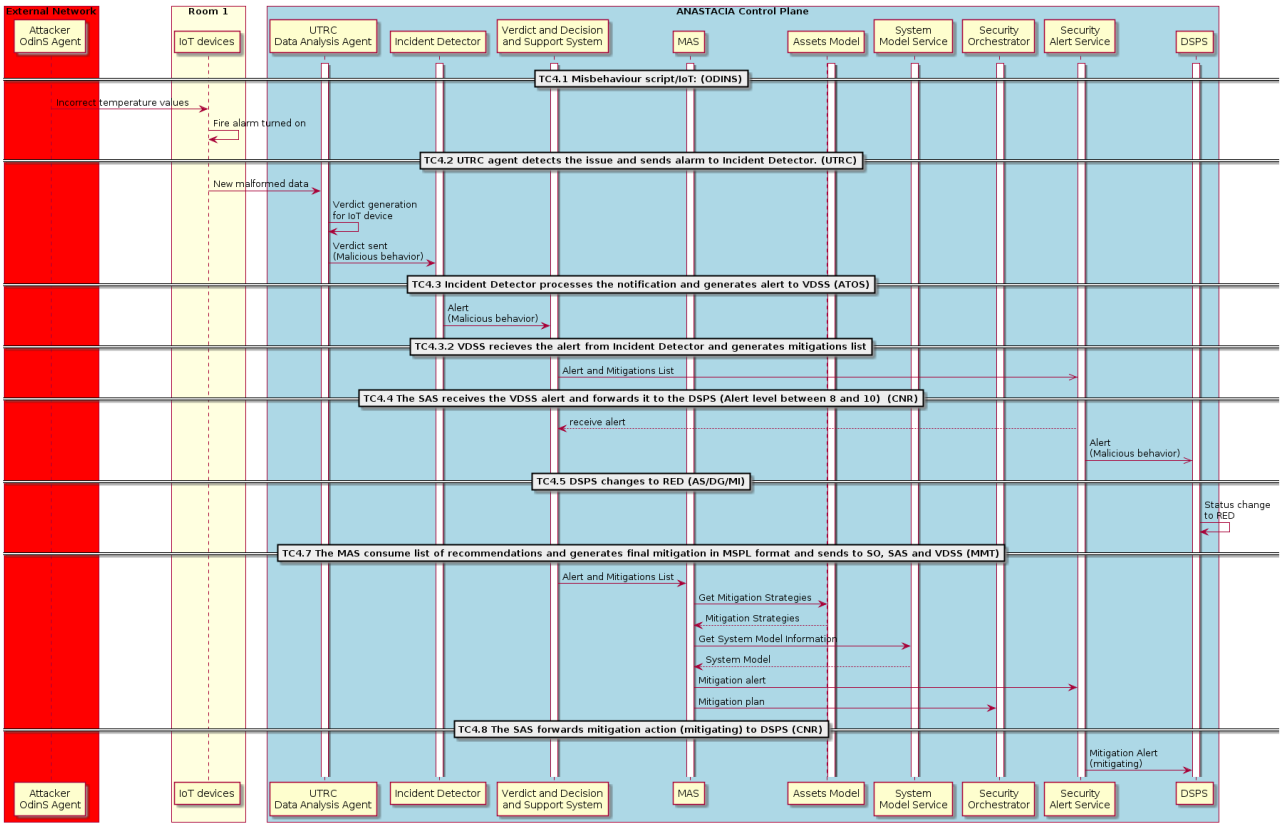

Figure 6. Step R4 test scenario combined with all test cases.

3.3.1.1 TC4.1 Misbehaviour script/IoT (ODINS)

TC4.1 Misbehaviour script/IoT (ODINS)	
Preconditions	• IoT devices are working and publishing temperature data via DTLS+COAP API towards the IoT-Broker of the Building Management System. • IoT-broker is forwarding the data information to Kafka message broker of ANASTACIA framework to be monitored. • Kafka message broker is configured and up and running. • Data Analysis is subscribed to the monitored data topic from the Kafka broker.
Components	• IoT Devices • BMS IoT Broker • Kafka Broker • Data Analysis
Execution	• IoT device receives a DTLS+COAP query to modify the temperature value generated to be sent towards the IoT broker. • IoT broker receives, stores and forwards temperature values in JSON format to inform the ANASTACIA framework. • Receiving JSON payload of temperature messages coming from OdinS IoT-broker on Kafka Broker being logged on Data Analysis.
Expected results	• Data Analysis is consuming physical values from IoT-broker via Kafka Broker • Data Analysis is storing data to a dataset correctly

Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time for transmitting the temperature data from IoT devices to be received by the Data Analysis component through the BMS IoT broker and the ANASTACIA Kafka broker.
Fail criteria	• Some part of data lost during conversion and not published on Kafka message broker • Exception caught during Data Analysis execution • Data records not stored on Data Analysis component

3.3.1.2 TC4.2 UTRC agent detects the issue and sends alarm to Incident Detector (UTRC)

TC4.2 UTRC agent detects the issue and sends alarm to Incident Detector. (UTRC)	
Preconditions	• ANASTACIA framework is working correctly • Attack running on IoT temperature sensors on ANASTACIA SEP premise
Components	• OdinS IoT GW • Kafka broker • ATOS VDSS
Execution	• Agent receives new IoT temperature sensor information (attacked values) • Agent detects anomalous behaviour on the sensor • Agent generates new verdict about the incident corresponding to sensor
Expected results	• Agent detects incident on the attacked IoT sensor(s) • Agent sends correct verdict to VDSS component
Expected completion	• Month 35 – November of 2019
KPI(s)	• New verdict is generated with less than 300ms • Successful detection ratio is higher than 80%
Fail criteria	• OdinS IoT GW is not working • Kafka broker is not working • Data is not being received from Kafka broker • Agent incorrectly interpreted data – detection is below 80% • VDSS is not working • VDSS is not received verdict information from UTRC Data Analysis Agent

3.3.1.3 TC4.3 Incident detector-VDSS processes the notification and generates alert and mitigations list (ATOS)

TC4.3 Incident detector – processes the notification and generates alert (ATOS)	
Preconditions	• The UTRC probe generates a "Data analysis report: Anomaly detected" when monitoring the IoT devices
Components	• Security sensors • Data Analysis • Data Filtering and pre-processing broker • Incident Detector
Execution	• Raw "Data analysis report: Anomaly detected" events are submitted by the MMT probe to the Data Filtering and pre-processing broker • The storm topology of the Data Filtering and pre-processing broker processes and sends the events through rsyslog to the Incident Detector • The Incident Detector receives the events through its agent, which normalizes them to a common format and submits it to the correlator engine of the Incident Detector, which processes it and generates a "Man in the Middle on IoT data" alert • The alert is pushed to a the RabbitMQ queue, exchange.alarms, to be consumed by DSPS and VDSS
Expected results	• Alerts triggered based on the events received from the monitoring infrastructure
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the processing time in the Incident Detector for the received event and send the generated alert to the RabbitMQ exchange
Fail criteria	• When an attack is performed, and no alert is generated

3.3.1.4 TC4.4 The SAS forwards alert to DSPS. (Alert level between 8 and 10) (CNR)

TC4.4 The SAS forwards alert to DSPS. (Alert level between 8 and 10) (CNR)	
Preconditions	• A new threat is detected by the Monitoring component • The VDSS sends the alert related to the threat to the SAS, through a dedicate RabbitMQ queue
Components	• VDSS, Verdicts and Decision Support System • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• SAS consume the RabbitMQ queue by retrieving the generated alert • The SAS forwards the alert, as it is, to another RabbitMQ server shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue

Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the alert from the RabbitMQ queue dedicated to VDSS-to-SAS communication and the sending of the (same) alert to the RabbitMQ queue dedicated to SAS-to-DSPS communication
Fail criteria	• The VDSS doesn't push the alert inside the RabbitMQ queue

3.3.1.5 TC4.5 DSPS changes to RED (AS/DG/MI)

TC4.5 DSPS changes to RED (AS/DG/MI)	
Preconditions	• Security part of **seal** is GREEN
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Security alert with high alert level is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security part of **seal** changes to RED
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until stored in DSPS secured storage • The time measurement since DSPS seal creation receives alert until DSPS GUI is notified of the new seal
Fail criteria	• Security part of seal doesn't change to RED • Seal is not stored in DSPS secured storage • DSPS GUI is not notified about new seal

3.3.1.6 TC4.6 VDSS generates a list of recommended mitigations (ATOS)

TC4.6 VDSS generates a list of recommended mitigations (ATOS)	
Preconditions	• A Man in the Middle on IoT data incident has been detected by the Incident Detector and an alarm is triggered and pushed to the RabbitMQ exchange.alarms
Components	• Incident Detector • VDSS • Assets Model • Mitigation Action Service • Security Alert Service • System Model Service
Execution	• The VDSS received an alert from the Incident Detector • The VDSS retrieves from the Assets Model the list of available mitigation strategies capable of mitigating the detected alert • The VDSS retrieves from the System Model Service information about the devices affected by the incident • The VDSS calculates suitability scores for every mitigation strategy and pushes it to an exchange.recommendations queue, to be consumed by the MAS and the SAS.
Expected results	• List of mitigations strategies with a suitability score calculated for each of them
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time since the VDSS receives the alert until it sends the recommendations to RabbitMQ
Fail criteria	• No mitigation is received and therefore no suitability scores are calculated

3.3.1.7 TC4.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)

TC4.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)	
Preconditions	• There is an alert on the VDSS RabbitMQ that contains the mitigations recommendations
Components	• Verdict and Decision Support System (VDSS): RabbitMQ endpoint • Mitigation Action Service (MAS) • Assets Model (AM) • System Model Service (SMS) • Security Alert Service (SAS) • Security Orchestrator (SO)
Execution	• An alert is published by the VDSS in the RabbitMQ channel. • The MAS receives the alert and starts the mitigation strategy processing.

	• The MAS contacts the auxiliary services to retrieve additional data to compute the MSPL (AM and SMS). • The MAS generates the MSPL and sends it to the SO. • The MAS sends the computed mitigation to the SAS and the VDSS.
Expected results	• The MAS generates an MSPL policy for orchestration that contains: two IoT Control security capabilities (for Power off and Reset), a L4 Filtering security capability, Network Traffic Analysis security capability and a Traffic Divert security capability.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the general time elapsed to compute the mitigation: starting from the reception of the alert until the MSPL is sent to the SO.
Fail criteria	• The MAS does not generate a MSPL policy for orchestration. • The MAS generates a policy that does not validate against the MSPL XSD schema. • The MAS generates an MSPL policy that does not contain the aforementioned security capabilities.

3.3.1.8 TC4.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)

TC4.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS generates MSPL data related to the alert
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• The MAS sends to the SAS the MSPL file, related to the given alert, previously received by the VDSS • The SAS encapsulates received data inside the previously received alert; in particular, the following attributes are added to the original alert • The SAS forward the enriched alert to the RabbitMQ server, shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the MSPL data from the MAS and the sending of the enriched alert to the RabbitMQ queue
Fail criteria	• The MAS fails to generate or send MSPL data to the SAS

3.3.2 Step 5 – R2

Step 5 validates interaction between reaction module components of ANASTACIA framework. In particular DSPS to maintain alert state while other components are preparing mitigation for active threat introduced in step 4. Main focus of this test is to prepare security policies that will be later used in following steps to mitigate attack (Policy Interpreter) and testing of SO and SAS communication about system model. The step is divided into following sequence (Figure 7):

- TC5.1 Reactive policy for orchestration enforcement (UMU/AALTO)
  - Turn off device which could reach other Rooms (IoT control).
  - Reboot compromised device (IoT control).
  - Drop traffic for compromised device (Filtering).
  - Monitor the BMS network.
  - Mirroring traffic in order to allow traffic inspection (Traffic Forward (Mirroring)).
- TC5.2 SO updates system model. (AALTO)
- TC5.3 SAS polls SO system model and sends to DSPS the alerts.
- TC5.4 DSPS remains RED. (AS/DG/MI) – SEE STEP3 as for "mitigating"

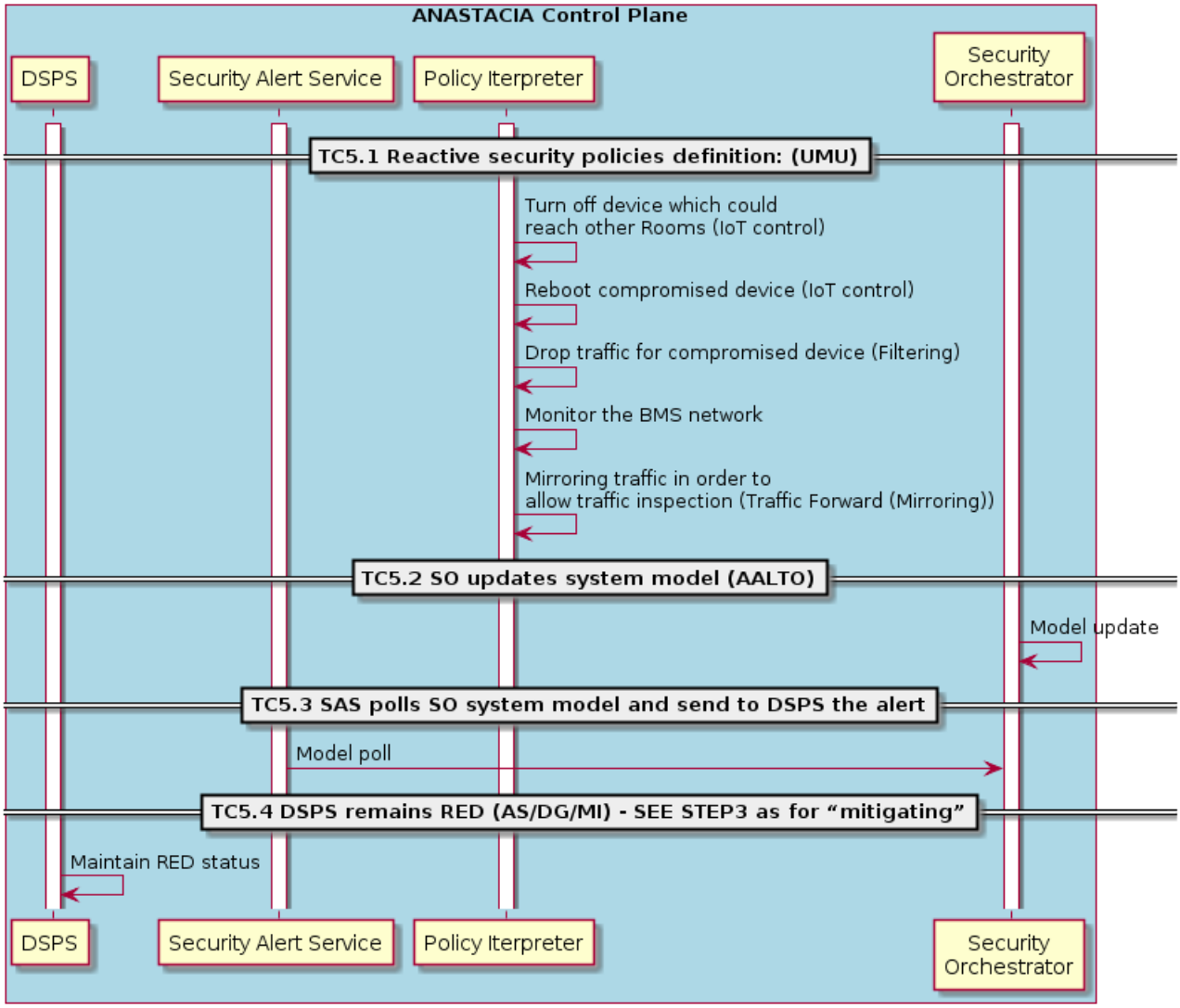

Figure 7. Step R5 test scenario combined with all test cases.

3.3.2.1 TC5.1 Reactive policy for orchestration enforcement (UMU/AALTO)

TC5.1 Reactive policy for orchestration enforcement (AALTO)	
Preconditions	• Security Orchestrator is ready to process MSPL-OP policies. • Policy Interpreter, Conflict detector and Policy repository are up and running. • Security Enablers Provider is up and running. • IoT Controller is up and running. • SDN Controller is up and running. • NFV Orchestrator is up and running. • OpenStack Virtual Infrastructure Manager (VIM) is up and running.
Components	• Security Orchestrator. • Policy Interpreter. • Security Enablers Provider.

Execution	• Security Orchestrator verifies MSPL-OP policy which contains:
	  ○ IoT control (Turn off).
	  ○ IoT control (Reboot)
	  ○ Filtering.
	  ○ BMS Network Monitoring.
	  ○ Traffic Forward (Mirroring)
	• Security Orchestrator communicates with the Security Enablers Provider for selecting the best enabler.
	• Security Orchestrator decides to use IoT Controller enabler for IoT control, ONOS for filtering and MMT Probe for network monitoring.
	• Security Enablers Provider informs the Security Orchestrator to use IoT controller enabler and ONOS enabler for mirroring and filtering the traffic.
	• Security Orchestrator requests policies translation from the Policy Interpreter.
	• Policy Interpreter verifies policy conflicts or dependencies by using the conflict detector.
	• Policy interpreter retrieves enabler plugins from the Security Enabler Provider and translates the MSPL-OP into final configurations.
	• Policy Interpreter returns final configurations and conflicts and dependencies detection to Security Orchestrator.
	• Security Orchestrator communicates with IoT controller to turn off and reboot the IoT devices.
	• Security Orchestrator communicates with NFV orchestrator to instantiate the network service instance (NSI) using the network service descriptor (NSD) of the MMT Probe.
	• NFV orchestrator communicates with OpenStack VIM to instantiate and configure the MMT Probe.
	• Security Orchestrator generates the open flow rules from the received configuration.
	• Security enforces the generated open flow rules through the REST API of ONOS to filter the traffic and mirror the generated traffic from the IoT devices to the MMT Probe for network monitoring.
	• Security Orchestrator configures the MMT Probe.

Expected results	• IoT Controller configurations. • Instantiation and configuration of MMT Probe component. • Configuration of traffic filtering. • IoT device which could reach other room is turned off. • Compromised IoT device is rebooted. • Traffic from compromised IoT device is dropped. • MMT Probe is dynamically deployed in the BMS.

	• MMT Probe in BMS is configured. • Traffic is mirrored to the MMT Probe.
Expected completion	• Month 35 – November of 2019
KPI(s)	• Policy translation times. • Conflict detection times. • IoT Controller enforcement times. • MMT Probe instantiation times. • MMT Probe configuration times. • Traffic mirroring and filtering times.
Fail criteria	• Policy translation fails due absence of required enablers in security enablers provider. • Policy translation fails due unexpected parameters or absence of required values in system model. • Conflict detection fails due unexpected parameters or absence of required values in system model. • Security Orchestrator fails due unexpected parameters or absence of required values in policy interpreter. • Security Orchestrator fails due unexpected parameters or misconfiguration of the NFV orchestrator. • Security Orchestrator fails due unexpected parameters or misconfiguration of the ONOS SDN controller.

3.3.2.2 TC5.2 SO updates system model (AALTO)

TC5.2 SO updates system model (AALTO)	
Preconditions	• Security Enablers Provider is up and running • Security Orchestrator received the MSPLs translations • Security Orchestrator succeeded/failed to enforce the enablers configuration
Components	• System model • Security orchestrator
Execution	• Security Orchestrator sends post request to update the enforcement API status
Expected results	• The system model updates the enforcement status
Expected completion	• Month 35 – November 2019
KPI(s)	• Measure the request time
Fail criteria	• Communication with the SM fails

3.3.2.3 TC5.3 SAS polls SO system model and sends to DSPS the alert (CNR)

TC5.3 SAS polls SO system model and sends to DSPS the alert	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS sends to the SAS the MSPL data related to the alert • The SM receives MSPL information from the MAS
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • SM, System Model
Execution	• The SAS polls the SM for a specific deployment identifier, previously received by the MAS • The SM answers with deployment data • The SAS extracts relevant fields from the data received by the SM • The SAS enriches (again) the alert to include relevant information received from the SM • The SAS forwards the enriched alert to the DSPS, through the dedicated RabbitMQ queue
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time spent between the reception of the (latest/valid) message from the SM and the sending of the enriched alert to the DSPS
Fail criteria	• The SM fails to send required data to the SAS

3.3.2.4 TC5.5 DSPS remains RED (AS/DG/MI)

TC5.5 DSPS remains RED (AS/DG/MI) – Same as TC2.8 but with RED status.	
Preconditions	• Security part of seal is RED
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert with mitigation action is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security part of **seal** remains RED

Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until processed
Fail criteria	• Security part of **seal** is not RED

3.4 TEST-CASE R3 – SLOW-DOS ATTACK AGAINST BMS WEB SERVICE (CNR, MONT)

3.4.1 Step 6 – R3

This step will validates MMT probe detection capabilities on ANASTACIA monitoring module section and later validates the ability of reaction module to start mitigating SlowDos attack. The steps for this test case are following:

- TC6.1 SlowDos attack is launched: (CNR)
  - IoT device starts HTTPS SlowDos against BMS web service.
- TC6.2 MMT probe detects the issue and notifies it to Incident Detector. (MONT)
- TC6.3 Incident detector-VDSS processes the notification and generates alert and mitigations list. (ATOS)
  - VDSS receives the alert from Incident Detector and generates mitigations list.
- TC6.4 The SAS forwards alert to DSPS. (Alert level between 6 and 8) (CNR)
- TC6.5 DSPS changes to YELLOW or RED according to severity & criticality. (AS/DG/MI)
- TC6.6 VDSS generates a list of recommended mitigations. (ATOS)
- TC6.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS. (MMT)
- TC6.8 The SAS forwards mitigation action (mitigating) to DSPS. (CNR)
-

Figure 8 illustrates ANASTACIA component interactions in test case for step R6 of the demonstration.

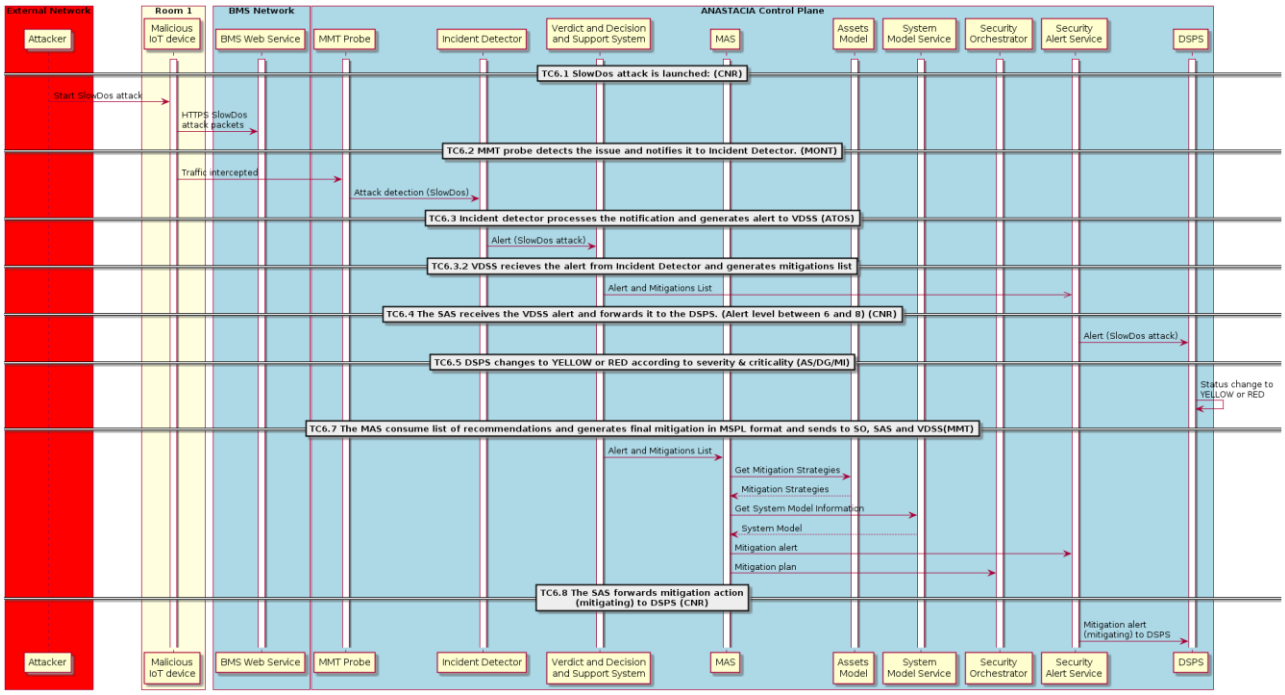

Figure 8. Step R6 test scenario combined with all test cases.

3.4.1.1 TC6.1 SlowDos attack is launched (CNR)

TC6.1 SlowDos attack is launched: (CNR)	
Preconditions	• Victim server online and reachable, without any active connection on the HTTP port • Attacker host deployed on the network and able to communicate with the victim
Components	• Attacker (Raspberry PI 3 Model B+) • Victim (VM running Apache2)
Execution	• The attacker runs the Slow DoS Attack tool targeting the victim
Expected results	• After a few seconds, the victim is not reachable on port 80
Expected completion	• Month 35 – November 2019
KPI(s)	• [No KPI for this test case]
Fail criteria	• The victim is not reachable on port 80, before the execution of the attack • At time of attack, the victim already has some established connections on port 80

3.4.1.2 TC6.2 MMT probe detects the issue and notifies it to Incident Detector (MONT)

TC6.2 MMT probe detects the issue and notifies it to Incident Detector. (MONT)	
Preconditions	• The MMT-Probe instance deployed as a result of TC5.4 is correctly running • The deployed instance of MMT-Probe is receiving a copy of the traffic directed to the BMS server.
Components	• Dynamic MMT-Probe instance • SlowDoS attacker • SlowDoS Victim (BMS server)
Execution	• The SlowDoS attack is launched as described in TC6.1.
Expected results	• The MMT-Probe detects the SlowDoS attack and raises a verdict about the issue detected.
Expected completion	• Month 35 – November 2019.
KPI(s)	• Obtain the elapsed time between the trigger of the attack and the detection.
Fail criteria	• The MMT-Probe does not detect the SlowDoS attack.

3.4.1.3 TC6.3 Incident detector-VDSS processes the notification and generates alert (ATOS)

TC6.3 Incident detector – processes the notification and generates alert (ATOS)	
Preconditions	• The MMT probe generates a “SlowComm attack detected” when monitoring the IoT devices
Components	• Security sensors • Data Filtering and pre-processing broker • Incident Detector
Execution	• Raw “SlowComm attack detected” events are submitted by the MMT probe to the Data Filtering and pre-processing broker • The storm topology of the Data Filtering and pre-processing broker processes and sends the events through rsyslog to the Incident Detector • The Incident Detector receives the events through its agent, which normalizes them to a common format and submits it to the correlator engine of the Incident Detector, which processes it and generates a “Slow DDoS Attack Detected” alert • The alert is pushed to a the RabbitMQ queue, exchange.alarms, to be consumed by SAS and VDSS
Expected results	• Alerts triggered based on the events received from the monitoring infrastructure
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the processing time in the Incident Detector to normalize the event, correlate it, generate the alert and send it to the RabbitMQ exchange

Fail criteria	• When an attack is performed, and no alert is generated

3.4.1.4 TC6.4 The SAS forwards alert to DSPS (CNR)

TC6.4 The SAS forwards alert to DSPS. (Alert level between 6 and 8) (CNR)	
Preconditions	• A new threat is detected by the Monitoring component • The VDSS sends the alert related to the threat to the SAS, through a dedicate RabbitMQ queue
Components	• VDSS, Verdicts and Decision Support System • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• SAS consumes the RabbitMQ queue by retrieving the generated alert • The SAS forwards the alert, as it is, to another RabbitMQ server shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the alert from the RabbitMQ queue dedicated to VDSS-to-SAS communication and the sending of the (same) alert to the RabbitMQ queue dedicated to SAS-to-DSPS communication
Fail criteria	• The VDSS doesn't push the alert inside the RabbitMQ queue

3.4.1.5 TC6.5 DSPS changes to YELLOW or RED according to severity & criticality (AS/DG/MI)

TC6.5 DSPS changes to YELLOW or RED according to security risk (AS/DG/MI)	
Preconditions	• Security part of seal is RED
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service

	• Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security part of **seal** remains RED, or lowered to YELLOW, according to security risk received
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until processed
Fail criteria	• Security part of seal becomes GREEN

3.4.1.6 TC6.6 VDSS generates a list of recommended mitigations (ATOS)

TC6.6 VDSS generates a list of recommended mitigations (ATOS)	
Preconditions	• A Slow DDoS Attack Detected incident has been detected by the Incident Detector and an alarm is triggered and pushed to the RabbitMQ exchange.alarms
Components	• Incident Detector • VDSS • Assets Model • Mitigation Action Service • Security Alert Service • System Model Service
Execution	• The VDSS receives an alert from the Incident Detector • The VDSS retrieves from the Assets Model the list of available mitigation strategies capable of mitigating the detected alert • The VDSS retrieves from the System Model Service information about the devices affected by the incident • The VDSS calculates suitability scores for every mitigation strategy and pushes it to an exchange.recommendations queue, to be consumed by the MAS and the SAS.
Expected results	• List of mitigations strategies with a suitability score calculated for each of them
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time since the VDSS receives the alert until it sends the recommendations to RabbitMQ
Fail criteria	• No mitigation is received and therefore no suitability scores are calculated

3.4.1.7 TC6.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)

TC6.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)	
Preconditions	• There is an alert on the VDSS RabbitMQ that contains the mitigations recommendations.
Components	• Verdict and Decision Support System (VDSS): RabbitMQ endpoint • Mitigation Action Service (MAS) • Assets Model (AM) • System Model Service (SMS) • Security Alert Service (SAS) • Security Orchestrator (SO)
Execution	• An alert is published by the VDSS in the RabbitMQ channel. • The MAS receives the alert and starts the mitigation strategy processing. • The MAS contacts the auxiliary services to retrieve additional data to compute the MSPL (AM and SMS). • The MAS generates the MSPL and sends it to the SO. • The MAS sends the computed mitigation to the SAS and the VDSS.
Expected results	• The MAS generates an MSPL policy for orchestration that contains the following security enablers: a L4 Filtering capability, an IoT Honeynet deployment capability and a Traffic Divert capability.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the general time elapsed to compute the mitigation: starting from the reception of the alert until the MSPL is sent to the SO.
Fail criteria	• The MAS does not generate a MSPL policy for orchestration. • The MAS generates a policy that does not validate against the MSPL XSD schema. • The MAS generates an MSPL policy that does not contain the aforementioned security capabilities.

3.4.1.8 TC6.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)

TC6.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS generates MSPL data related to the alert
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• The MAS sends to the SAS the MSPL file, related to the given alert, previously received by the VDSS

	• The SAS encapsulates received data inside the previously received alert; in particular, the following attributes are added to the original alert • The SAS forwards the enriched alert to the RabbitMQ server, shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the MSPL data from the MAS and the sending of the enriched alert to the RabbitMQ queue
Fail criteria	• The MAS fails to generate or send MSPL data to the SAS

3.4.2 Step 7 – R3

Step 7 illustrated on Figure 9 will validate reaction module to enforce malicious device isolation and ability of ANASTACIA framework to signal state change through DSPS UI as well as other reaction module components. The test case includes following steps:

- TC7.1 Reactive policy for orchestration enforcement (UMU/AALTO)
  - Isolate compromised Rooms (Filtering)
  - Deploy and configure IoT Honeynet
  - Traffic Divert (Mirroring) of external traffic against MMT Probe for traffic inspection
- TC7.2 SO updates system model (AALTO)
- TC7.3 SAS polls SO system model and send to DSPS the alert
- TC7.4 DSPS depends from previous status

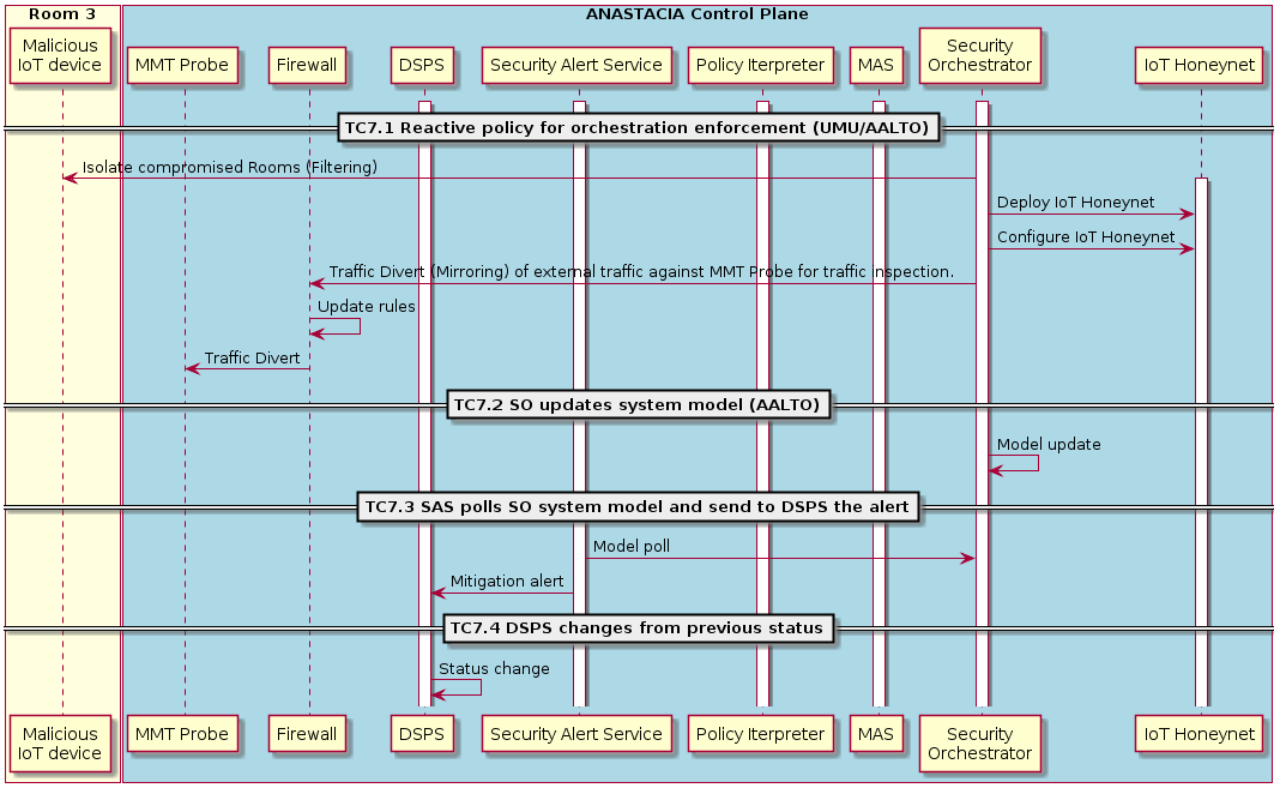

Figure 9. Step R7 test scenario combined with all test cases.

3.4.2.1 TC7.1 Reactive policy for orchestration enforcement (UMU/AALTO)

TC7.1 Reactive policy for orchestration enforcement (UMU/AALTO)	
Preconditions	• Security Orchestrator is ready to process MSPL-OP policies • Policy Interpreter, Conflict detector and Policy repository are up and running • Security Enablers Provider is up and running. • SDN Controller is up and running. • NFV Orchestrator is up and running. • OpenStack Virtual Infrastructure Manager (VIM) is up and running.
Components	• Security Orchestrator • Policy Interpreter • Security Enablers Provider

Execution	• Security Orchestrator verifies MSPL-OP policy which contains:
	  ○ Filtering MSPL
	  ○ IoT Honeynet MSPL
	  ○ Traffic divert (Mirroring)
	• Security Orchestrator communicates with the Security Enablers Provider for selecting the best enabler.
	• Security Orchestrator decides to use: i) OVS-FW enabler for filtering the traffic; ii) IoT Honeynet; iii) ONOS enabler for traffic network monitoring.
	• Security Orchestrator requests policies translation from the Policy Interpreter.
	• Policy Interpreter verifies policy conflicts or dependencies by using the conflict detector.
	• Policy Interpreter retrieves enabler plugins from the Security Enabler Provider and translates the MSPL-OP into final configurations.
	• Policy Interpreter returns final configurations and conflicts and dependencies detection to Security Orchestrator.
	• Security Orchestrator communicates with NFV orchestrator to instantiate the network service instance (NSI) using the NFV-OVS network service descriptor (NSD).
	• NFV orchestrator communicates with OpenStack VIM to instantiate and configure the OVS-FW.
	• Security Orchestrator generates the open flow rules from the received configuration to redirect the traffic to OVS-FW.
	• Security enforces the generated open flow rules through the REST API of ONOS to redirect the traffic to pass through the new instantiated OVS-FW.
	• Security Orchestrator communicates with NFV orchestrator to instantiate the network service instance (NSI) using the IoT Honeynet network service descriptor (NSD).
	• NFV orchestrator communicates with OpenStack VIM to instantiate and configure the IoT Honeynet.
	• Security Orchestrator generates the open flow rules from the received configuration to redirect the traffic to the IoT Honeynet.
	• Security enforces the generated open flow rules through the REST API of ONOS to redirect the traffic to the IoT Honeynet.
	• Security Orchestrator deploys a Cooja agent and enforces IoT Honeynet configurations by using the Cooja agent driver.

Expected results	• OVS-FW deployment and configurations. • Cooja IoT Honeynet deployment and configurations. • Configurations for filtering of traffic using OVS-FW. • Traffic divert (Mirroring). • Rooms 1 and 2 are isolated.

Expected completion	• Month 35 – November of 2019
KPI(s)	• Policy translation times. • Conflict detection times. • OVS-FW instantiation times. • OVS-FW configuration times. • Cooja IoT Honeynet instantiation times. • Cooja IoT Honeynet configuration times. • Traffic mirroring and filtering times.
Fail criteria	• Policy translation fails due absence of required enablers in security enablers provider. • Policy translation fails due unexpected parameters or absence of required values in system model. • Conflict detection fails due unexpected parameters or absence of required values in system model. • Security Orchestrator fails due unexpected parameters or absence of required values in policy interpreter. • Security Orchestrator fails due unexpected parameters or misconfiguration of the NFV orchestrator. • Security Orchestrator fails due unexpected parameters or misconfiguration of the ONOS SDN controller.

3.4.2.2 TC7.2 SO updates system model (AALTO)

TC7.2 SO updates system model (AALTO)	
Preconditions	• Security Enablers Provider is up and running • Security Orchestrator received the MSPLs translations • Security Orchestrator succeeded/failed to enforce the enablers configuration
Components	• System model • Security orchestrator
Execution	• Security Orchestrator sends post request to update the enforcement API status
Expected results	• The system model updates the enforcement status
Expected completion	• Month 35 – November 2019
KPI(s)	• Measure the request time
Fail criteria	• Communication with the SM fails

3.4.2.3 TC7.3 SAS polls SO system model and send to DSPS the alert (CNR)

TC7.3 SAS polls SO system model and sends to DSPS the alert

Preconditions	• The VDSS sends a given alert to the SAS • The MAS sends to the SAS the MSPL data related to the alert • The SM receives MSPL information from the MAS
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • SM, System Model
Execution	• The SAS polls the SM for a specific deployment identifier, previously received by the MAS • The SM answers with deployment data • The SAS extracts relevant fields from the data received by the SM • The SAS enriches (again) the alert to include relevant information received from the SM • The SAS forwards the enriched alert to the DSPS, through the dedicated RabbitMQ queue
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time spent between the reception of the (latest/valid) message from the SM and the sending of the enriched alert to the DSPS
Fail criteria	• The SM fails to send required data to the SAS

3.4.2.4 TC7.4 DSPS changes status (AS/DG/MI)

TC7.4 DSPS changes status	
Preconditions	• Security part of seal is RED or YELLOW
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert with mitigated flag/information is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security risk lowers • Security part of seal may change to YELLOW or GREEN, depending on previous colour

Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until processed
Fail criteria	• Security part of **seal** is RED • Seal is not stored in DSPS secured storage • DSPS GUI is not notified about new seal

3.5 Test-Case R4 – External attacker tries to Compromise IoT devices (MONT)

The fourth and last phase of the Smart Building use case is focused on preventing a potential overtake of the IoT network from an external attacker. The main objective of this phase is to evaluate the capacity of the ANASTACIA platform to protect the IoT network from an external attempt to gain control over a set of already-compromised IoT devices. In this phase, an external attacker (probably the coordinator of a large-scale attack) tried to connect to a set of IoT nodes that have already been comprised during the last phases, with the objective of escalate even further the attack and gain control over the whole network. In this sense, this section presents two sub-steps to implement and test the R4 of the Smart Building testbed.

The following subsections present the two steps that aim to detect and generate the mitigation, and to enforce the mitigation previously crafted.

3.5.1 Step 8 – R4

In this subsection we describe a set of test cases involved in the first half of the R4 phase. The main goals of this test cases are: (1) to simulate a connection from an external user (a malicious user that is trying to gain control over the already-compromised IoT devices), (2) to detect the attempt by using the security assets that are available on the network, (3) to generate the corresponding security alert, (4) to create the respective mitigation plan for the detected attack, and (5) to send the mitigation to the Security Alert Service in order to update the value of the DSPS.

- TC8.1 External attacker tries to reach IoT domain: (ODINS)
  - CoAP requests coming from external IP addresses
- TC8.2 MMT solution detects attacker from external IP address. (MONT)
- TC8.3 Incident detector-VDSS processes the notification and generates alert and mitigations list. (ATOS)
- TC8.4 The SAS forwards alert to DSPS. (CNR)
- TC8.5 DSPS changes to RED. (AS/DG/MI)
- TC8.6 VDSS generates a list of recommended mitigations. (ATOS)
- TC8.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS. (MMT)
- TC8.8 The SAS forwards mitigation action (mitigating) to DSPS. (CNR)
- TC8.9 DSPS maintains RED state. (AS/DG/MI)
- TC8.10 SAS keeps polling SO system model. (CNR)

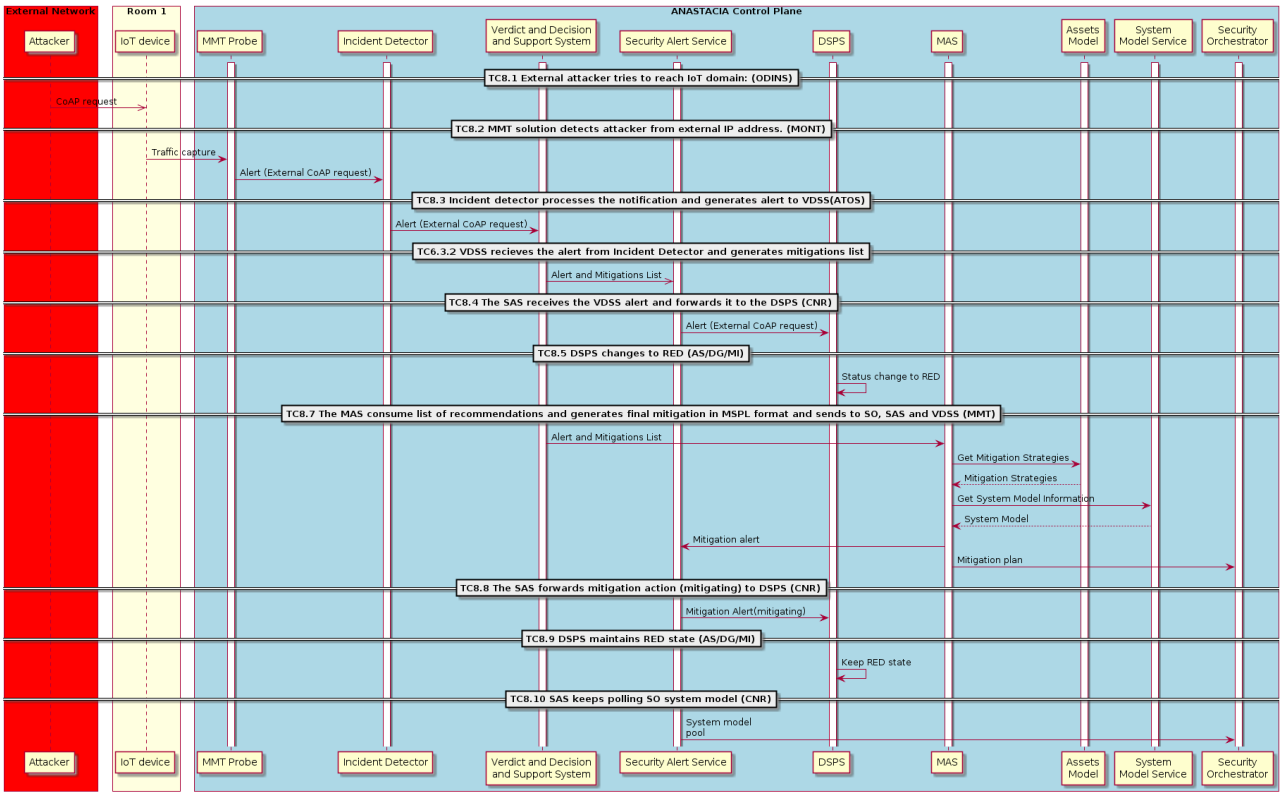

Figure 10. Step R8 test scenario combined with all test cases.

3.5.1.1 TC8.1 External attacker tries to reach IoT domain: (ODINS)

TC8.1 External attacker tries to reach IoT domain: (ODINS)	
Preconditions	• Script is ready to emulate the external attack query. • IoT devices are working with online available DTLS + COAP API.
Components	• Malicious Script for External Attack • IoT Device
Execution	• Malicious Script is launched. • Data traffic from an external IP address is trying to reach a IoT device.
Expected results	• External data traffic must be received in the network infrastructure of the smart building.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time for generating the external data traffic of the received query and modified the IP destination address in the internal configuration.
Fail criteria	• Exception caught during the query reception • Excessive time for the query processing

3.5.1.2 TC8.2 MMT solution detects attacker from external IP address. (MONT)

TC8.2 MMT solution detects attacker from external IP address. (MONT)	
Preconditions	• Outsider malicious script is ready to try to connect to the compromised IoT devices. • The MMT-Probe instance deployed as a result of TC5.4 is correctly running.
Components	• Malicious script and outsider device. • Dynamic MMT-Probe instance.
Execution	• The external connection is launched as described in T8.1.
Expected results	• The MMT-Probe detects the external connection and raises a verdict about the issue detected
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time between the trigger of the attack and the detection.
Fail criteria	• The MMT-Probe does not detect the SlowDoS attack

3.5.1.3 TC8.3 Incident detector-VDSS processes the notification and generates alert and mitigations list (ATOS)

TC8.3 Incident detector – processes the notification and generates alert (ATOS)	
Preconditions	• The MMT probe generates a “SQL injection” event when monitoring the IoT devices
Components	• Security sensors • Data Analysis • Data Filtering and pre-processing broker • Incident Detector
Execution	• Raw “SQL injection” events are submitted by the MMT probe to the Data Filtering and pre-processing broker • The storm topology of the Data Filtering and pre-processing broker processes and sends the events through rsyslog to the Incident Detector • The Incident Detector receives the events through its agent, which normalizes them to a common format and submits it to the correlator engine of the Incident Detector, which processes it and generates a “SQL Injection Detected” alert • The alert is pushed to a the RabbitMQ queue, exchange.alarms, to be consumed by DSPS and VDSS
Expected results	• Alerts triggered based on the events received from the monitoring infrastructure
Expected completion	• Month 35 – November 2019

KPI(s)	• Obtain the processing time in the Incident Detector to normalize the event, correlate it, generate the alert and send it to the RabbitMQ exchange
Fail criteria	• When an attack is performed, and no alert is generated

3.5.1.4 TC8.4 The SAS forwards alert to DSPS (CNR)

TC8.4 The SAS forwards alert to DSPS (CNR)	
Preconditions	• A new threat is detected by the Monitoring component • The VDSS sends the alert related to the threat to the SAS, through a dedicate RabbitMQ queue
Components	• VDSS, Verdicts and Decision Support System • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• SAS consumes the RabbitMQ queue by retrieving the generated alert • The SAS forwards the alert, as it is, to another RabbitMQ server shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the alert from the RabbitMQ queue dedicated to VDSS-to-SAS communication and the sending of the (same) alert to the RabbitMQ queue dedicated to SAS-to-DSPS communication
Fail criteria	• The VDSS doesn't push the alert inside the RabbitMQ queue

3.5.1.5 TC8.5 DSPS changes to RED (AS/DG/MI)

TC8.5 DSPS changes to RED (AS/DG/MI)	
Preconditions	• Security part of **seal** is YELLOW
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Security alert is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal

Expected results	• Security part of **seal** changes to RED
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until stored in DSPS secured storage • The time measurement since DSPS seal creation receives alert until DSPS GUI is notified of the new seal
Fail criteria	• Security part of seal changes doesn't change to RED • Seal is not stored in DSPS secured storage • DSPS GUI is not notified about new seal

3.5.1.6 TC8.6 VDSS generates a list of recommended mitigations (ATOS)

TC8.6 VDSS generates a list of recommended mitigations (ATOS)	
Preconditions	• A "SQL Injection Detected" incident has been detected by the Incident Detector and an alarm is triggered and pushed to the RabbitMQ exchange.alarms
Components	• Incident Detector • VDSS • Assets Model • Mitigation Action Service • Security Alert Service • System Model Service
Execution	• The VDSS receives an alert from the Incident Detector • The VDSS retrieves from the Assets Model the list of available mitigation strategies capable of mitigating the detected alert • The VDSS retrieves from the System Model Service information about the devices affected by the incident • The VDSS calculates suitability scores for every mitigation strategy and pushes it to an exchange.recommendations queue, to be consumed by the MAS and the SAS.
Expected results	• List of mitigations strategies with a suitability score calculated for each of them
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the elapsed time since the VDSS receives the alert until it sends the recommendations to RabbitMQ
Fail criteria	• No mitigation is received and therefore no suitability scores are calculated

3.5.1.7 TC8.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)

TC8.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)	
Preconditions	• There is an alert on the VDSS RabbitMQ that contains the mitigations recommendations.
Components	• Verdict and Decision Support System (VDSS): RabbitMQ endpoint • Mitigation Action Service (MAS) • Assets Model (AM) • System Model Service (SMS) • Security Alert Service (SAS) • Security Orchestrator (SO)
Execution	• An alert is published by the VDSS in the RabbitMQ channel. • The MAS receives the alert and starts the mitigation strategy processing. • The MAS contacts the auxiliary services to retrieve additional data to compute the MSPL (AM and SMS). • The MAS generates the MSPL and sends it to the SO. • The MAS sends the computed mitigation to the SAS and the VDSS.
Expected results	• The MAS generates an MSPL policy for orchestration that contains a single Traffic divert security capability.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the general time elapsed to compute the mitigation: starting from the reception of the alert until the MSPL is sent to the SO.
Fail criteria	• The MAS does not generate a MSPL policy for orchestration. • The MAS generates a policy that does not validate against the MSPL XSD schema. • The MAS generates an MSPL policy that does not contain the aforementioned security capabilities.

3.5.1.8 TC8.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)

TC8.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS generates MSPL data related to the alert
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • DSPS, Dynamic Security and Privacy Seal
Execution	• The MAS sends to the SAS the MSPL file, related to the given alert, previously received by the VDSS

	• The SAS encapsulates received data inside the previously received alert; in particular, the following attributes are added to the original alert • The SAS forwards the enriched alert to the RabbitMQ server, shared with the DSPS
Expected results	• The DSPS consumes the data from the RabbitMQ queue
Expected completion	• Month 35 – November 2019
KPI(s)	• The time passing between the reception of the MSPL data from the MAS and the sending of the enriched alert to the RabbitMQ queue
Fail criteria	• The MAS fails to generate or send MSPL data to the SAS

3.5.1.9 TC8.9 DSPS maintains RED state (AS/DG/MI)

TC8.9 DSPS status remains RED (AS/DG/MI)	
Preconditions	• Security part of seal is RED
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert with mitigation action is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service • Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal
Expected results	• Security part of **seal** remains RED
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until processed

Fail criteria	• Security part of **seal** is not RED

3.5.1.10 TC8.10 SAS keeps polling SO system model (CNR)

TC8.10 SAS keeps polling SO system model(CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS sends to the SAS the MSPL data related to the alert • The SM receives MSPL information from the MAS
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • SM, System Model
Execution	• The SAS polls the SM for a specific deployment identifier, previously received by the MAS
Expected results	• The SAS retrieves from the SM deployment information related to the alert countermeasures
Expected completion	• Month 35 – November 2019
KPI(s)	• Measure the time passing between the reception of the message from the MAS and the first polling to the SM
Fail criteria	• Communication with the SM fails

3.5.2 Step 9 – R4

This step represents last test case (Figure 11) in main ANASTACIA framework demonstration where SO will deploy reactive policy by redirecting traffic to Honeynet and changing state of dynamic security and privacy seal to green status as consequence of successful ANASTACIA security policies deployment into SEP. The test case includes following steps:

- TC9.1 Reactive policy for orchestration enforcement (AALTO)
  - Redirect traffic to the honey network.
- TC9.2 DSPS will change to GREEN after the security and privacy analysis (AS/DG/MAND)

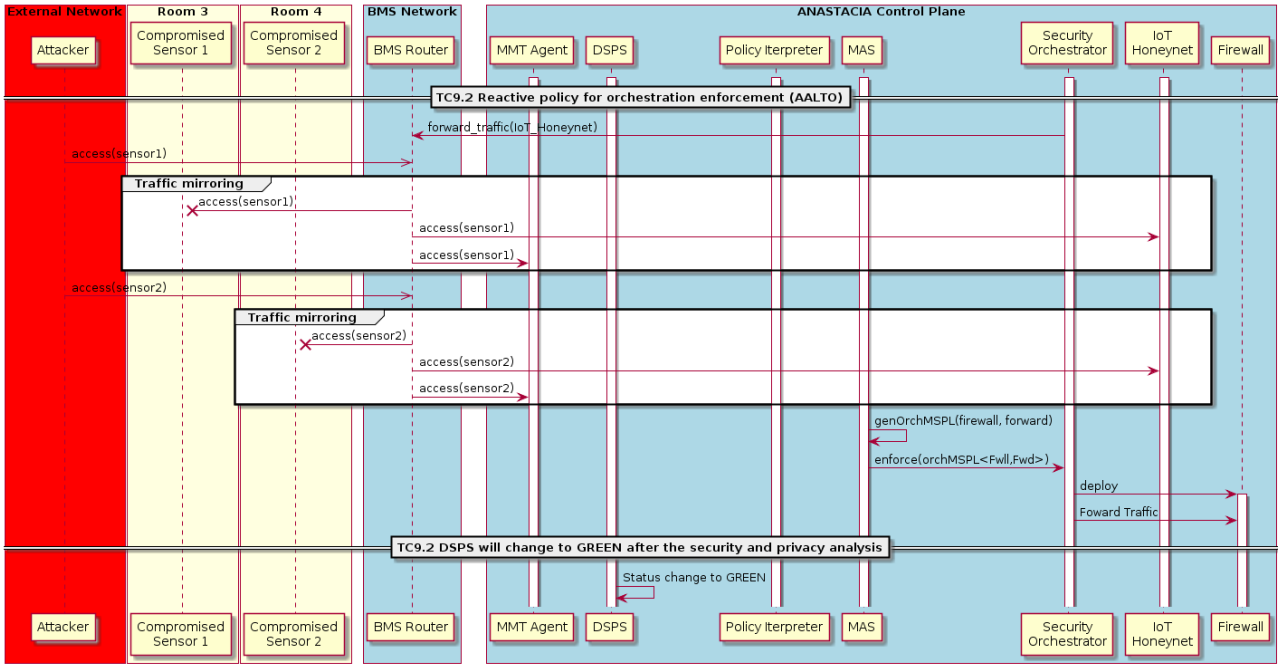

Figure 11. Step R9 test scenario combined with all test cases.

3.5.2.1 TC9.1 Reactive policy for orchestration enforcement (UMU/AALTO)

TC9.1 Reactive policy for orchestration enforcement (AALTO)	
Preconditions	• Security Orchestrator is ready to process MSPL-OP policies • Policy Interpreter, Conflict detector and Policy repository are up and running • Security Enablers Provider is up and running. • SDN Controller is up and running. • NFV Orchestrator is up and running. • OpenStack Virtual Infrastructure Manager (VIM) is up and running.
Components	• Security Orchestrator. • Policy Interpreter. • Security Enablers Provider.
Execution	• Security Orchestrator verifies MSPL-OP policy which contains: ○ Traffic Divert (Forward) MSPL • Security Orchestrator communicates with the Security Enablers Provider for selecting the best enabler. • Security Orchestrator decides to use OVS-FW enabler for filtering the traffic and ONOS enabler for redirecting the traffic. • Security Orchestrator requests policies translation from the Policy Interpreter. • Policy Interpreter verifies policy conflicts or dependencies by using the conflict detector. • Policy Interpreter retrieves enabler plugins from the Security Enabler Provider and translates the MSPL-OP into final configurations. • Policy Interpreter returns final configurations and conflicts and dependencies detection to Security Orchestrator. • Security Orchestrator communicates with NFV orchestrator to instantiate the network service instance (NSI) using the NFV-OVS network service descriptor (NSD). • NFV orchestrator communicates with OpenStack VIM to instantiate and configure the OVS-FW.

	• Security Orchestrator generates the open flow rules from the received configuration to redirect the traffic to OVS-FW. • Security enforces the generated open flow rules through the REST API of ONOS to redirect the traffic to pass through the new instantiated OVS-FW.
Expected results	• OVS-FW deployment and configurations. • Configurations for filtering of traffic using OVS-FW. • Traffic divert (Mirroring).
Expected completion	• Month 35 – November of 2019
KPI(s)	• Policy translation times. • Conflict detection times. • OVS-FW instantiation times. • OVS-FW configuration times. • Traffic mirroring and filtering times.
Fail criteria	• Policy translation fails due absence of required enablers in security enablers provider. • Policy translation fails due unexpected parameters or absence of required values in system model. • Conflict detection fails due unexpected parameters or absence of required values in system model. • Security Orchestrator fails due unexpected parameters or absence of required values in policy interpreter. • Security Orchestrator fails due unexpected parameters or misconfiguration of the NFV orchestrator. • Security Orchestrator fails due unexpected parameters or misconfiguration of the ONOS SDN controller.

3.5.2.2 TC9.2 DSPS will change to GREEN after the security and privacy analysis (AS/DG/MI)

TC9.2 DSPS will change to GREEN after the security and privacy analysis	
Preconditions	• Security part of seal is RED
Components	• DPSP Agent • DSPS Seal Creation Service • DSPS Security alert to Privacy risk mapping service • DSPS Secured Storage • DSPS GUI
Execution	• Alert with mitigated flag/information is received • The message format of the alert is converted using STIX • Alert is processed by DSPS Creation Service

	• Privacy risks are mapped from security alert • New seal is created with updated security and privacy risk levels • New seal is stored • GUI is notified about new seal • CISO/DPO restores seal for security and privacy
Expected results	• Both security and privacy seal changes to GREEN
Expected completion	• Month 35 – November 2019
KPI(s)	• The time measurement since AMQP message is consumed from queue until posted to DSPS seal creation service • The time measurement since DSPS seal creation receives alert until stored in DSPS secured storage • The time measurement since DSPS seal creation receives alert until DSPS GUI is notified of the new seal
Fail criteria	• Security part of seal doesn't change to GREEN • Seal is not stored in DSPS secured storage • DSPS GUI is not notified about new seal

4 5G USE CASE – NETWORK SLICING (ERICSSON)

4.1 OVERVIEW

In addition to the main scenario, a new 5G ***[31]*** scenario was developed to demonstrate ability of ANASTACIA framework to integrate with new technologies fast and without much interruption to overall development time and adjustment required for technology assimilation with ANASTACIA.

A key feature of 5G mobile networks is the support of network slicing. Network slicing allows operating virtual mobile networks on top of the physical infrastructure, as enabled by virtualization of network functions and SDN. Each slice can be customized for a particular type of traffic and can provide network functions and configuration specialized for the particular use case. Network slices are isolated both in terms of traffic and resources (bandwidth, CPU, storage, etc.). Thus, depletion of resources in one slice does not impact other slices. From a security perspective, this is an important property, as a security breach in one slice is prevented from spreading to other slices. Isolation of resources plays a major role in preventing DDoS attacks between slices.

While current 5G network will first support a low number of slices for generic traffic classes as well as for large enterprises, we envision a future where slices operate at a much more granular level. In the scope of research, we propose slices specific to particular use cases, applications, regions, device types and even to individual devices. This is enabled with the lower cost and overhead of operating slices due to virtualization as well as due to automated network management.

In the ANASTACIA project, we propose the use of *quarantine slices*. These are slices reserved for devices with reduced trust, e.g. as a result of suspicious behavior observed by a monitoring system. The malicious devices are isolated from other devices to prevent the effects of an attack or from preventing malware from spreading. The system might prefer to isolate these devices through quarantine slices rather than completely blocking or disconnecting device especially in the cases where the malicious behavior is unconfirmed or as a first reaction. This prevents critical services from becoming inoperational and prevents breaking potential service agreements.

Figure 12 illustrates Ericsson 5G network slicing demonstration interaction with ANASTACIA framework ***[32]***. The scenario is split into the part running in the UMU testbed and a 5G core network part that, for logistics reasons, is located in Ericsson's premises. Locally in Murcia there is a 5G Base Station Emulator to which two devices are connected: one legitimate video producer and one video producer that acts as an attacker. The 5G Base State Emulator is a software implementation of the central base station components running in an Intel NUC with the radio network implemented using Wi-Fi. The client devices are both implemented with Raspberry Pis and connect to the Base Station Emulator using Wi-Fi. One of the client devices is infected with malware (attack script) that causes it to send massive traffic toward an external target. Inside the 5G Base Station Emulator an MMT Probe is installed that monitors the traffic and detects non-legitimate traffic. In this case, non-legitimate traffic is defined as non-video traffic with a bandwidth exceeding a defined threshold.

The 5G Core Network is running in a remote data center. The 5G Network supports multiple network slices orchestrated via an orchestration platform including a Multi-domain Slice Orchestrator, an NFV Orchestrator (NFVO), a Network Slice Selection Function (NSSF), an Overlay Network Controller and an SDN controller. These components are implemented outside the ANASTACIA project but have been modified with functionality to support security operations. The new features include the possibility to deny access to a given slice and to transfer client devices between network slices.

In the normal situation, the client devices are connected to a network slices dedicated to camera streaming traffic. This slice contains a streaming server implemented using the VLC software that allows video receivers connect with video transmitters and performs adaptation of the video format. There is a pre-existing definition of a quarantine slice with strict limitations in the available bandwidth. Once the exceptional

behavior in the attacking device has been detected, the device is blocked from the normal slice and redirected to the quarantine slice.

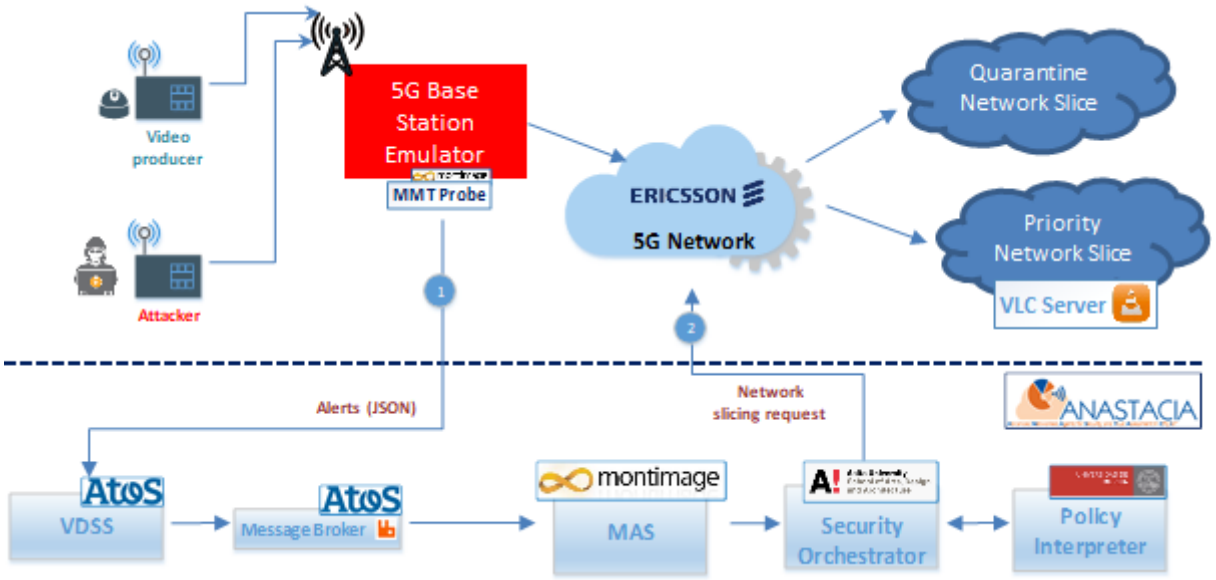

Figure 12. Ericsson 5G network slicing demonstration.

4.2 TEST-CASE OF 5G SCENARIO – ATTACK BY 5G CAMERA

This test-case is selected to demonstrate the use of 5G network slicing as mitigation and isolation method. In this test-case a network connected camera has been compromised and become part of a DDoS attack. The camera has therefore started sending traffic targeting (together with other compromised devices not included in the demo) to overload a server. The objective in this use-case is to detect the abnormal traffic from the device and to isolate the device using network slicing. This section identifies a set of test-cases to implement and evaluate the 5G scenario and evaluates the interaction among components of ANASTACIA framework grouped in modules including the IoT Infrastructure, the Monitoring Module, the Reaction Module, the Seal Manager, the Orchestration Plane and the Control Domain.

4.2.1 5G SCENARIO Step 1

In this subsection, we describe the following test-cases for the 5G scenario, where an IoT device sends excessive traffic to an outside attack target:

- TC10.1 IoT device starts sending excessive traffic to an outside attack target. (ER)
- TC10.2 MMT solution detects excessive traffic to unknown destinations. (MONT)
- TC10.3 Incident detector – processes the notification and generates alert. (ATOS)
- TC10.4 The SAS forwards alert to DSPS. (CNR)
- TC10.5 DSPS changes to RED. (AS/DG/MI)
- TC10.6 VDSS generates a list of recommended mitigations. (ATOS)
- TC10.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS. (MMT)
- TC10.8 The SAS forwards mitigation action (mitigating) to DSPS. (CNR)
- TC10.9 DSPS maintains RED state. (AS/DG/MI)
- TC10.10 SAS keeps polling SO system model. (CNR)
- TC10.11 Reactive policy for orchestration enforcement. (UMU/AALTO)

- TC10.12 DSPS will change to GREEN after the security and privacy analysis. (AS/DG/MI)

Most test-cases are similar to the ones in the previous scenario, with the main differences visible in TC10.1, TC10.2, TC10.7 and TC10.11. The 5G network slicing scenario has been illustrated in Figure 13.

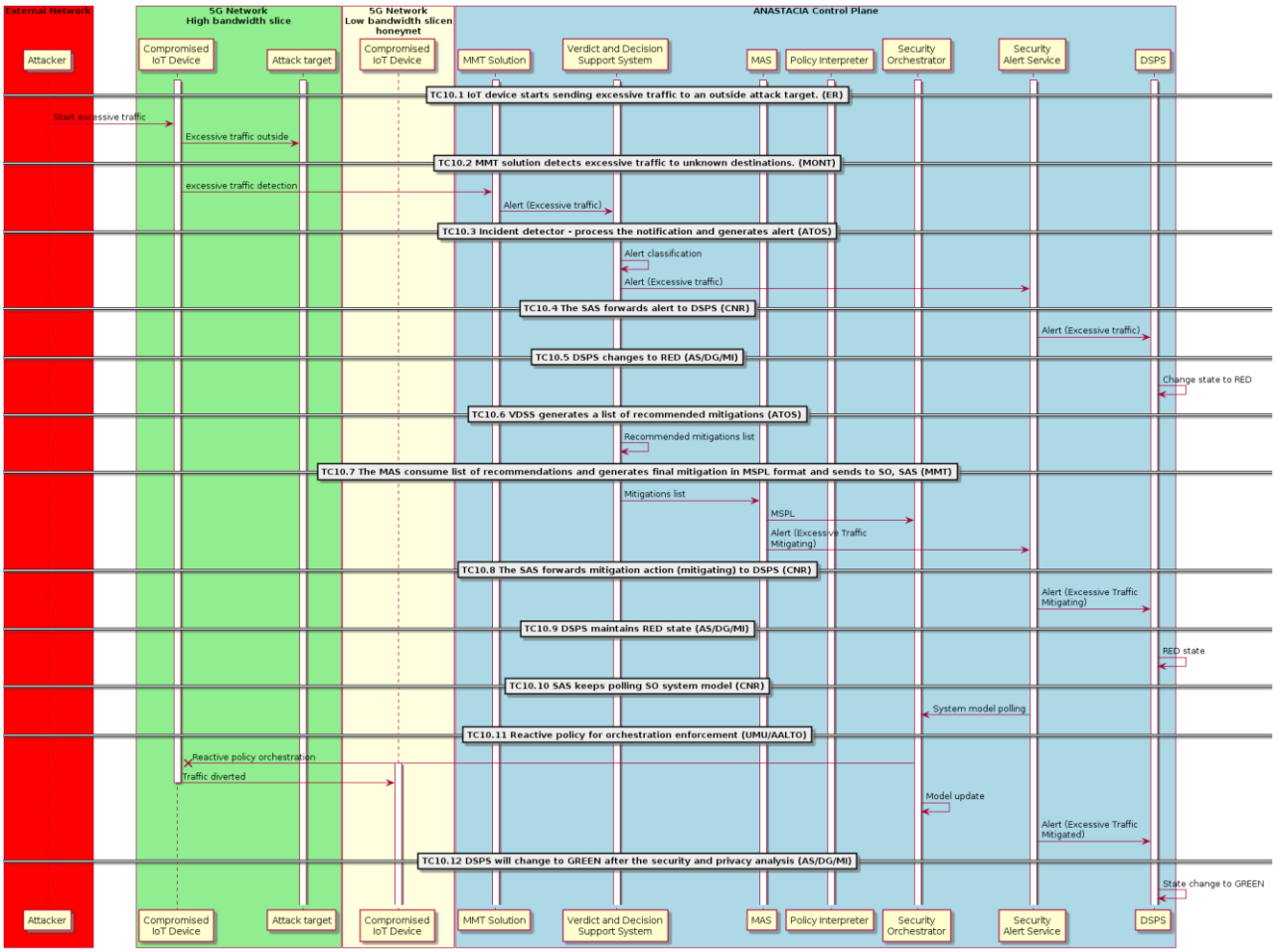

Figure 13. Ericsson 5G network slicing test case steps.

4.2.1.1 TC10.1 IoT device starts sending excessive traffic to an outside attack target (ER)

TC10.1 IoT device starts sending excessive traffic to an outside attack target. (ER)	
Preconditions	• IoT device has been infected with malware (malicious script) which causes it to send large amounts of traffic to an outside target.
Components	• IoT Device
Execution	• IoT Device starts a new HTTP connection to an IP destination address external to the network in order to send a bandwidth intensive stream of data
Expected results	• HTTP traffic is transmitting to an IP destination address outside the network.

Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the bandwidth received by the IP destination outside the network.
Fail criteria	• Insufficient amount of HTTP traffic reaches the attack target.

4.2.1.2 TC10.2 MMT solution detects excessive traffic to unknown destinations (MONT)

TC10.2 TC10.2 MMT solution detects excessive traffic to unknown destinations. (MONT)	
Preconditions	• Malicious IoT device is ready to try to send traffic to outside target. • The MMT-Probe instance is deployed in the 5G network and is correctly running.
Components	• IoT device with malicious script • Dynamic MMT-Probe instance.
Execution	• The traffic to the outside target is started as described in TC10.1
Expected results	• The MMT-Probe detects the exceptional traffic to the outside target and raises a verdict about the issue detected.
Expected completion	• Month 35 – November 2019
KPI(s)	• Elapsed time between the trigger of the attack and the detection
Fail criteria	• The MMT-Probe does not detect the exceptional traffic from the IoT device.

4.2.1.3 TC10.3 Incident detector-VDSS processes the notification and generates alert and mitigations list (ATOS)

Please refer to same test case as in **3.2.2.3**

4.2.1.4 TC10.4 The SAS forwards alert to DSPS (CNR)

Please refer to same test case as in **3.2.2.4.**

4.2.1.5 TC10.5 DSPS changes to RED (AS/DG/MI)

Please refer to same test case as in **3.5.1.5.**

4.2.1.6 TC10.6 VDSS generates a list of recommended mitigations (ATOS)

Please refer to same test case as in **3.3.1.6.**

4.2.1.7 TC10.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)

TC10.7 The MAS consumes list of recommendations and generates final mitigation in MSPL format and sends to SO, SAS (MMT)	
Preconditions	• There is an alert on the VDSS RabbitMQ that contains the mitigations recommendations.
Components	• Verdict and Decision Support System (VDSS): RabbitMQ endpoint • Mitigation Action Service (MAS) • Assets Model (AM) • System Model Service (SMS) • Security Alert Service (SAS) • Security Orchestrator (SO)
Execution	• An alert is published by the VDSS in the RabbitMQ channel. • The MAS receives the alert and starts the mitigation strategy processing. • The MAS contacts the auxiliary services to retrieve additional data to compute the MSPL (AM and SMS). • The MAS generates the MSPL and sends it to the SO. • The MAS sends the computed mitigation to the SAS and the VDSS.
Expected results	• The MAS generates an MSPL policy for orchestration that contains a single Traffic divert security capability.
Expected completion	• Month 35 – November 2019
KPI(s)	• Obtain the general time elapsed to compute the mitigation: starting from the reception of the alert until the MSPL is sent to the SO.
Fail criteria	• The MAS does not generate a MSPL policy for orchestration. • The MAS generates a policy that does not validate against the MSPL XSD schema. • The MAS generates an MSPL policy that does not contain the aforementioned security capabilities.

4.2.1.8 TC10.8 The SAS forwards mitigation action (mitigating) to DSPS (CNR)

Please refer to same test case as in **3.2.2.7.**

4.2.1.9 TC10.9 DSPS maintains RED state (AS/DG/MI)

Please refer to same test case as in **3.5.1.9.**

4.2.1.10 TC10.10 SAS keeps polling SO system model (CNR)

TC10.10 SAS keeps polling SO system model(CNR)	
Preconditions	• The VDSS sends a given alert to the SAS • The MAS sends to the SAS the MSPL data related to the alert • The SM receives MSPL information from the MAS
Components	• VDSS, Verdicts and Decision Support System • MAS, Mitigation Action Service • SAS, Security Alert Service • SM, System Model
Execution	• The SAS polls the SM for a specific deployment identifier, previously received by the MAS
Expected results	• The SAS retrieves from the SM deployment information related to the alert countermeasures
Expected completion	• Month 35 – November 2019
KPI(s)	• Measure the time passing between the reception of the message from the MAS and the first polling to the SM
Fail criteria	• Communication with the SM fails

4.2.1.11 TC10.11 Reactive policy for orchestration enforcement (UMU/AALTO/ER)

TC10.11 Reactive policy for orchestration enforcement (AALTO)	
Preconditions	• Security Orchestrator is ready to process MSPL-OP policies • Policy Interpreter, Conflict detector and Policy repository are up and running • Security Enablers Provider is up and running. • 5G Security Enabler is up and running • OSM and OS are up and running
Components	• Security Orchestrator • Policy Interpreter • Security Enablers Provider • 5G Security Enabler
Execution	• Security Orchestrator verifies MSPL-OP policy which contains: ○ Network Slicing Control MSPL • Security Orchestrator requests policies translation to policy Interpreter • Policy Interpreter verifies policy conflicts or dependencies by using the conflict detector. • Policy Interpreter retrieves enabler plugins from the Security Enabler Provider and translates the MSPL-OP into final configurations. • Policy Interpreter returns final configurations and conflicts and dependencies detection to Security Orchestrator. • Security Orchestrator sends request to 5G Security Enabler to deploy and enforce mitigation action recommended by policy orchestrator • 5G Security Enabler receives request from Security Orchestrator to restrict slice access

	• 5G Security Enabler communicates with AMF in 5G network which enforces the restriction
Expected results	• AMF configured with device specific slice restriction • IoT device is unregistered from current network slice • IoT device is redirected to alternative network slice
Expected completion	• Month 35 – November of 2019
KPI(s)	• Time to restrict device from current slice
Fail criteria	• 5G Security Service does not receive request from Security Orchestrator • Device is not restricted from current slice • Device can still re-connect to current slice

4.2.1.12 TC10.12 DSPS will change to GREEN after the security and privacy analysis (AS/DG/MI)

Please refer to same test case as in **3.5.2.2.**

5 END-USER QUESTIONNAIRE (ODINS)

This chapter provides generic user questionnaire that will be used for evaluating the feedback provided by end-users. This end-user questionnaire allows evaluating the second release of the integrated ANASTACIA framework regarding the validated use cases of **Building Management System** (BMS) and **Mobile Edge Computing in 5G Network** (MEC). The second release integrates monitoring, detection, reaction, orchestration and enforcement. After the validation of use cases, the end-users evaluate the results and provide their feedback. This document provides generic user questionnaire for each use case to validate the outcomes. Each question is rated in accordance to a Likert scale:

1. = **Very Low** – **fully disagree**,
2. = **Low** – **partially disagree**,
3. = **Medium** – **neutral**,
4. = **High** – **partially agree**,
5. = **Very High** – **fully agree**.

This questionnaire contains general questions that have been prepared in accordance to the end-user requirements defined in deliverable D1.2. Table 3 represents generic questions to be answered in the range from 1 fully disagree to 5 fully agree.

Table 3. End-user questionnaire

Number	Question	Assessment level (1-5)
1	Is the ANASTACIA framework **useful to monitor and detect cyber-attacks?**	
2	Does the ANASTACIA framework provide **valuable automatic reactions to mitigate cyber-attacks**?	
3	Does the ANASTACIA framework have **intuitive user interfaces for managing security policies and operations**?	
4	Does the ANASTACIA framework provide **clear seal and report about the cybersecurity status?**	
5	Would you like to have ANASTACIA framework **to protect your cybersecurity and privacy in your workspace or home**?	

Moreover, two open questions are available to collect further information from ANASTACIA users. Questions are listed in Table 4 below.

Table 4. End-user questionnaire – open questions

Number	Open question
1	What do you like the most about the ANASTACIA framework?
2	What do you dislike the most about the ANASTACIA framework?

6 SUMMARY

ANASTACIA test results proved holistic approach of security by design to monitor, react and mitigate IoT threats for smart building in real site attack demonstration in UMU. The approach taken by ANASTACIA team to deploy flexible architecture capable of defending cyber IoT perimeter against threats has been validated successfully. Multiple techniques used to detect malicious behavior on perimeter were employed and tested positively. In return, accurate detection of ANASTACIA framework enabled the rest of the ANASTACIA system to react and deploy countermeasures that protected site from further cyber-intrusion. Y3 smart building scenario in UMU proved the point where security by design plays significant role and if applied correctly, can bring benefits to building owners, occupants and IT staff by minimizing human intervention in cyber-security process only exposing operator to most important parts where significant decision is required to take place.

The most important key takeaways and lessons learnt from ANASTACIA framework testing are following:

1. Implementation of new AI-based techniques to detect zero-day vulnerabilities in the system. ANASTACIA helps mitigate this problem by implementing AI-based novel techniques to detect them. It is important to note that even most advanced techniques might provide false/positive detections, this human control is required to observe system actions. Cyber-protection system has to be continually monitored and updated if required due to emerging threats that might compromise IoT perimeter through zero-day attacks.
2. Novel approach to cybersecurity system human supervision by giving high level view for cyber-security experts by implementing dynamic security and privacy seal to alert system users about potential undesired cyber activities within protection perimeter.
3. Security Policy resolution and autonomous mitigation action to find best cybersecurity answer for observed threat. ANASTACIA approach provides cybersecurity experts very fast reaction time to answer the threat with mitigation and at the same time, enabling cybersecurity team to investigate further given threat by observing its malicious activity in a honey net.
4. Demonstration of 5G network slicing showcasing flexibility of ANASTACIA infrastructure to adopt to new technologies that can be easily integrated via range of APIs with existing system.

Further details related to ANASTACIA KPIs, scalability, performance characteristics and user feedback can be found in last deliverable of WP6 – D6.6.

7 REFERENCES

1. D1.2 – "User Centred Requirements Initial Analysis" – https://seafile.gruppoitaleaf.com/f/bb40291107/
2. D1.5 – "Final Architectural Design" – https://seafile.gruppoitaleaf.com/f/ab4d6a1858/
3. D2.2 – "Attacks and Threats Analysis and Contingency Actions" – https://seafile.gruppoitaleaf.com/f/fa9d752539/
4. D2.8 – "Secure Software Development Guidelines Final Report" – https://seafile.gruppoitaleaf.com/f/a156563f12/
5. D4.4 – "Final Monitoring Component Services Implementation Report" – https://seafile.gruppoitaleaf.com/f/904b40367e/
6. D4.5 – "Final Reaction Component Services Implementation Report" – https://seafile.gruppoitaleaf.com/f/1828800cb8/
7. D4.6 – "Final Agents Development Report" – https://seafile.gruppoitaleaf.com/f/41a647f439/
8. D6.2 – "Initial use cases implementation and tests Report" – https://seafile.gruppoitaleaf.com/f/6906d7daa9/
9. D6.4 – "Final Technical Integration and Validation Report" – https://seafile.gruppoitaleaf.com/f/0f736b7f1d/
10. D6.6 – "Final End-user validation and evaluation Report"
11. CNR logging service – https://gitlab.com/anastacia-project/anastacia-logging-service
12. Docker - https://www.docker.com/
13. Consul service – https://www.consul.io/docs/agent/services.html
14. Cambiaso, E., Papaleo, G., & Aiello, M. (2017). "Slowcomm: Design, development and performance evaluation of a new slow DoS attack" – Journal of Information Security and Applications, 35, 23-31.
15. Cambiaso E, Papaleo G, Aiello M. (2012) "Taxonomy of slow DoS attacks to web applications", International Conference on Security in Computer Networks and Distributed Systems 2012 Oct 11 (pp. 195-204). Springer, Berlin, Heidelberg.
16. Cambiaso, E., Papaleo, G., Chiola, G., & Aiello, M. (2013). "Slow DoS attacks: definition and categorization" – International Journal of Trust Management in Computing and Communications, 1(3-4), 300-319.
17. Aiello, Maurizio, et al. "A similarity based approach for application DoS attacks detection", 2013 IEEE Symposium on Computers and Communications (ISCC). IEEE, 2013. p. 000430-000435.
18. Shamsolmoali, P., & Zareapoor, M. (2014, September). "Statistical-based filtering system against DDOS attacks in cloud computing". In 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 1234-1239). IEEE.
19. Katkar, V., Zinjade, A., Dalvi, S., Bafna, T., & Mahajan, R. (2015, February). "Detection of DoS/DDoS attack against HTTP servers using naive Bayesian", 2015 International Conference on Computing Communication Control and Automation (pp. 280-285). IEEE.
20. Singh, K. J., & De, T. (2015). "An approach of DDoS attack detection using classifiers", Emerging Research in Computing, Information, Communication and Applications (pp. 429-437). Springer, New Delhi.
21. Lu, K., Wu, D., Fan, J., Todorovic, S., & Nucci, A. (2007). "Robust and efficient detection of DDoS attacks for large-scale internet", Computer Networks, 51(18), 5036-5056.
22. Yuan, X., Li, C., & Li, X. (2017, May). "DeepDefense: identifying DDoS attack via deep learning", 2017 IEEE International Conference on Smart Computing (SMARTCOMP) (pp. 1-8). IEEE.
23. Brynielsson, J., & Sharma, R. (2015, August). "Detectability of low-rate HTTP server DoS attacks using spectral analysis", 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (pp. 954-961). IEEE.
24. Cambiaso, E., Papaleo, G., Chiola, G., & Aiello, M. (2016). "A Network Traffic Representation Model for Detecting Application Layer Attacks", International Journal of Computing and Digital Systems, 5(01).
25. D. Kushner, "The real story of stuxnet," in IEEE Spectrum, vol. 50, no. 3, pp. 48-53, March 2013 – https://ieeexplore.ieee.org/document/6471059

26. S.B. Lee, M.S Kang, and V.D Gligor. “Codef: collaborative defense against large-scale link-flooding attacks”, the ninth ACM conference on Emerging networking experiments and technologies, p. 417–428, 2013.
27. C. Liaskos, V. Kotronis, and X. Dimitropoulos. “A novel framework for modeling and mitigating distributed link flooding attacks”, IEEE International Conference on Computer Communications, IEEE INFOCOM, pp. 1-9, 2016.
28. D. Gkounis, V. Kotronis, C. Liaskos, and X. Dimitropoulos. “On the interplay of link-flooding attacks and traffic engineering”, ACM SIGCOMM Computer Communication Review, pp. 5-11, 2016.
29. M.S. Kang, V.D Gligor, and V. Sekar. “Spiffy: Inducing cost-detectability tradeoffs for persistent link-flooding attacks”, The Network and Distributed System Security Symposium (NDSS), 2016.
30. D. Belabed, M. Bouet, V. Conan, “Centralized Defense Using Smart Routing Against Link-Flooding Attacks”, IEEE 2nd Cyber Security in Networking Conference (CSNet), Paris, 2018.
31. ETSI White Paper No. 28; MEC in 5G networks, First edition – June 2018,
32. Juniper White Paper; MEC use cases & Deployment options, July 2016