ANASTACIA

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

D4.5

Final Reaction Component Services Implementation Report

Distribution level	[PU]
Contractual date	31.10.2019 [M34]
Delivery date	31.10.2019 [M34]
WP / Task	WP4 / T4.5
WP Leader	MONT
Authors	Ivan Vaccari (CNR), Enrico Cambiaso (CNR), Elisabetta Punta (CNR), Silvia Scaglione (CNR), Ruben Trapero (ATOS), Jesus Villalobos (ATOS), Diego Rivera (MONT), Dallal Belabed (THALES)
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

PUBLIC SUMMARY

This deliverable describes the implementation of the Reaction Module of ANASTACIA, demanded to the identification and propagation of reaction countermeasures to deploy on the system, given information on a threat, retrieved directly from the Monitoring Module. The deliverable describes how the module is contextualized in the ANASTACIA framework, its internal components and how each of them is implemented and designed to communicate on the network.

1 INTRODUCTION

1.1 AIMS OF THE DOCUMENT

This document describes the final deliverable of WP4 focused on the Reaction Module implemented inside the ANASTACIA framework. This document contains detailed information on the final implementation and the final workflow of the Reaction Module to automatically trigger reactions against detected threats. Section 2 introduces the current status of the ANASTACIA framework architecture related to the Monitoring and Reaction plane.

In the following, Section 2 describes exhaustively the components of the reaction module, while Section **Errore. L'origine riferimento non è stata trovata.** details internal communication. Finally, Section 3 concludes the document.

1.2 APPLICABLE AND REFERENCE DOCUMENTS

This document refers to the following documents:

- D1.3 – Initial Architecture Design
- T2.6 – Attacks Threats Analysis and Contigency Actions Final Report
- MS12b – Reaction component services specified and agreed by the board
- D4.1 - Initial Monitoring Component Services Implementation Report
- D4.2 - Initial Reaction Component Services Implementation Report
- D4.4 – Final Monitoring Components Services Implementation Report

1.3 REVISION HISTORY

Version	Date	Author	Description
0.1	24/07/2019	Ivan Vaccari (CNR), Enrico Cambiaso (CNR)	ToC
0.2	15/09/2019	Ruben Trapero (ATOS), Jesus Villalobos (ATOS)	ATOS contributions
0.3	25/09/2019	Diego Rivera (MONT)	First draft of MONT contributions
0.4	02/10/2019	Ivan Vaccari (CNR), Enrico Cambiaso (CNR)	CNR contributions
0.5	02/10/2019	Ruben Trapero (ATOS) Jesus Villalobos (ATOS)	ATOS minor update
0.6	14/10/2019	Diego Rivera (MONT) Dallal Belabed (THALES)	MONT minor update and THALES contribution
0.7	22/10/2019	Satya Vuppala (UTRC)	Review by UTRC

0.8	22/10/2019	Ivan Vaccari (CNR), Elisabetta Punta (CNR)	Minor updates to address part of the comments received by UTRC
0.9	24/10/2019	Enrico Cambiaso (CNR), Silvia Scaglione (CNR)	Minor updates to address comments for ATOS, MONT and UMU
1.0	24/10/2019	Ivan Vaccari (CNR), Enrico Cambiaso (CNR)	Almost final version
1.1	25/10/2019	Ivan Vaccari (CNR), Enrico Cambiaso (CNR)	Final version

1.4 Acronyms and Definitions

Acronym	Meaning
SAS	Security Alert Service
VDSS	Verdicts and Decision Support System
MAS	Mitigation Action Service
DSPS	Dynamic Security and Privacy Seal
AM	Assets Model

2 THE ANASTACIA REACTION COMPONENT

One of the final aims of ANASTACIA framework is to implement an innovative strategy and system able to automatically react when threats executed against a system are detected and forwarded by the monitoring module. The monitoring and reaction modules are developed to implement this automatic execution once threats are identified/detected. The monitoring module, described in both D4.1 and D4.4, is developed in order to detect when a threat is executed against the system where the ANASTACIA framework is installed. Instead, the reaction module, described in both this document and D4.2, focuses on the identification of the reaction strategies to deploy, as a consequence of an identified threat.

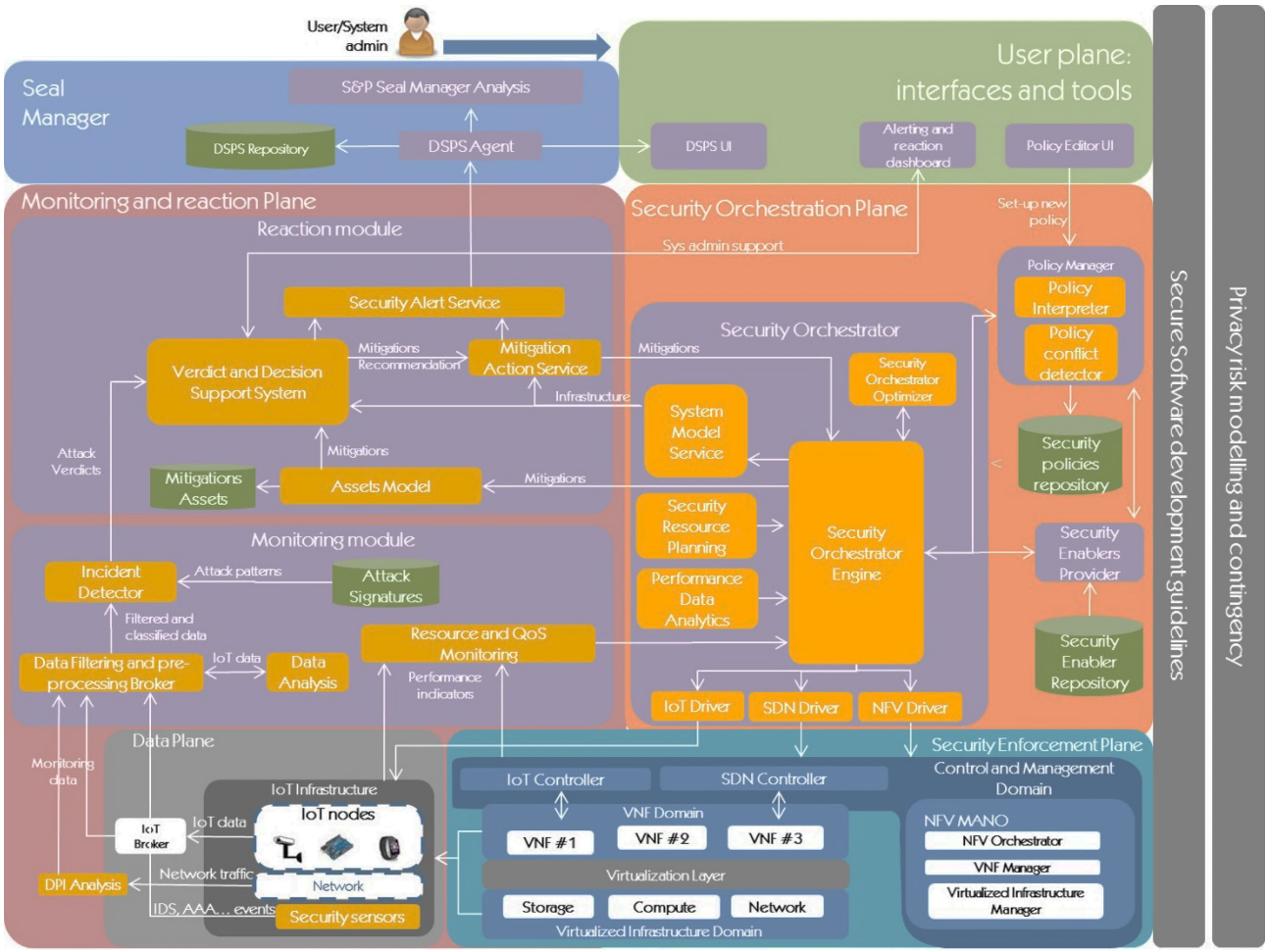

Figure 1 – The architecture of the ANASTACIA framework

As can be seen from Figure 1, the Reaction module communicates with the Monitoring module, the Seal Manager and the Security Orchestrator Plane. In collaboration with the Monitoring module, the aim of the Reaction module is indeed to automatically decide and apply countermeasures that could be deployed by the framework in order to mitigate the detected threat and to share the defined mitigation activity with the other module of the framework in order to quickly react and protect the system.

In the next sections, the final status of each component of the Reaction Module is described in detail.

2.1 THE ANASTACIA FRAMEWORK

The current version of the architecture is designed for the Reaction Module in terms of components and iterations that the different components have internally and externally with other modules of the ANASTACIA framework. This design is lightweight and reliable in order to quickly select countermeasures that can be

deployed inside the framework when a threat is identified by the Monitoring module to prevent the damage and automatically react in order to reduce damages to the system.

Figure 2 reports the final version of the Reaction Module, after some development and design iteration accomplished to refine the architecture.

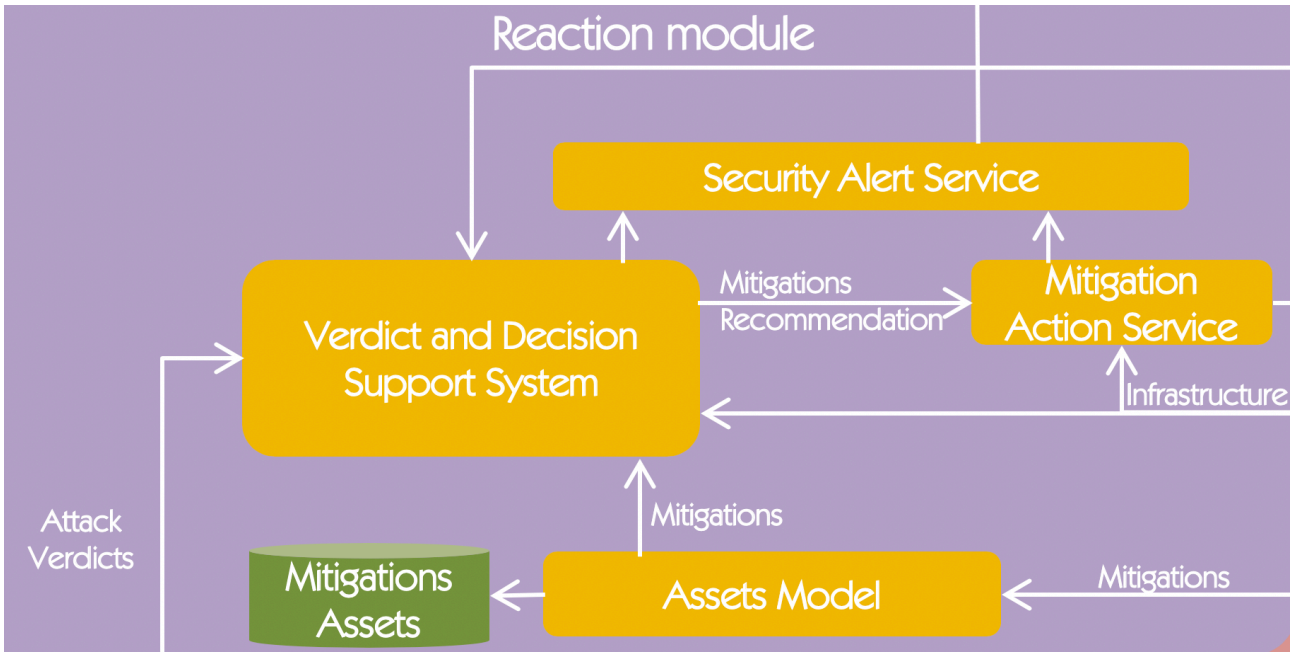

Figure 2 - Reaction module design

In the current version of the architecture, the Reaction Module consists of the following components:

- **Security Alert Service (SAS)**: this component is adopted to retrieve alerts form the VDSS component, analyse and enrich them with further information (retrieved by the MAS), and send the expanded output to the seal management plane (DSPS);
- **Verdict and Decision Support System (VDSS)**: component responsible for evaluating the incidents detected and the available mitigations in order to propose the appropriate countermeasure;
- **Assets Model (AM)**: component that continuously collect and update the existing capabilities, flavors, mitigations and their related threats and costs;
- **Mitigations Assets**: database where the information used by the Assets Model are stored;
- **Mitigation Action Service (MAS)**: this component analyses the outputs of the VDSS and transform these alerts into a dedicate MSPL file that specifies the countermeasure that as to be deployed to react against the detected attack.

2.2 Component's Description and Implementation

In this section, a detailed description of each component is reported.

2.2.1 Verdicts and Decision Support System

The Verdicts and Decision Support System (VDSS) is responsible for evaluating the incidents detected and the available mitigations in order to decide on the most appropriate one. Several variables are considered for this decision, such as the type of incident to mitigate and its importance, the type of device affected by the incident and its criticality within the IoT infrastructure (Figure 3). Additionally, it is also considered the characteristics of the mitigations. This includes variables such as the resources required to enforce it or the impact in the infrastructure of enforcing it, which might determine whether the mitigation affects to other parts of the infrastructure (i.e., turning off the fire sensors of a complete floor which might be a critical one).

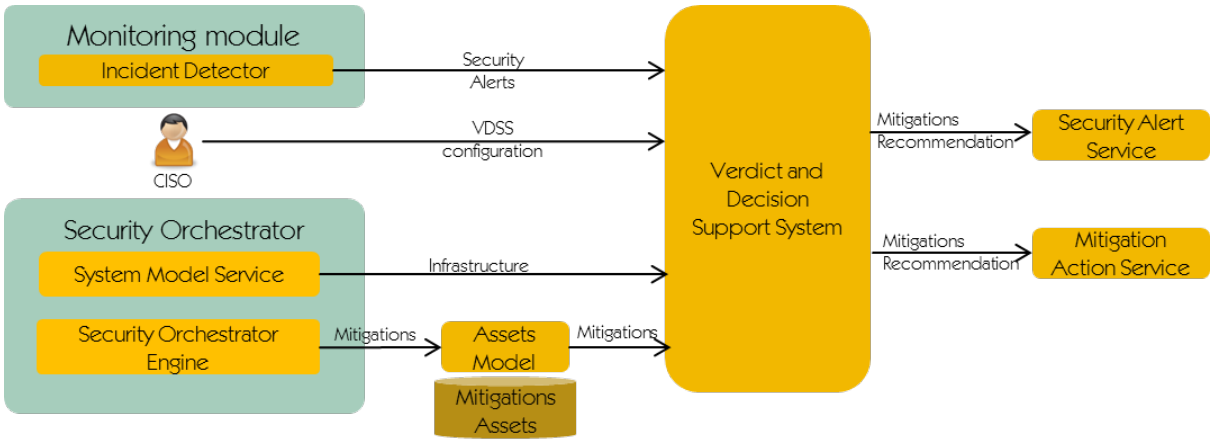

Figure 3. High level overview of the VDSS

Although the activities carried out by the VDSS are envisioned to be done in an automatic way, it is still considered certain human intervention by the CISO to fine tune the mitigation recommendation procedure. For example, it is CISO's role to adjust the cost of the mitigation as it is considered as a subjective value which depends on several factors such as the time to deploy the mitigation, the time the system will be offline, the resources devoted to mitigating or the human effort required.

All together the VDSS produces a list of mitigations attached to a score which represents the suitability of the mitigation for the incident detected and considering all the rest of the factors that impact on the IoT infrastructure. Such list is used by the Mitigation Action Service to enforce the mitigations selected by the VDSS. In addition, the list of mitigations is also notified to the DSPS through the Security Alert Service. Further details are given in Sections 2.3.2 and 2.3.3.

The following subsections detail the insights of the VDSS, detailing its internal design, inputs, outputs and activities carried out.

2.2.1.1 VDSS internal design

Figure 4 depicts the internal architecture of the VDSS and its interaction with the main external components of the ANASTACIA framework. These subcomponents can be grouped into three main parts:

- Components to manage input received from messaging queues - To this end, two components (RabbitMQ Alarms Consumer and RabbitMQ MAS Selection Consumer) interconnects with the RabbitMQ server available at the Reaction module, which is used for the exchange of information with the Incident Detector at the Monitoring module and with other components of the Reaction module, such as the MAS.
- Components related to user interfaces - A GUI is available at the VDSS which allows the CISO to carry out certain configuration and management activities. To this end, two components manage the reception of data required by the GUI:
    - GUI WebSocket, which is used to provide real time information about mitigation enforced and alarms triggered by the Incident Detector.
    - GUI API, which is used to manage REST API invocations to exchange data required or inserted by the CISO.
- Components related to the core activities of the VDSS - Here we can identify two main components:
    - The VDSS Engine Model is the core of the VDSS, which carries out the calculation of scores for the available mitigations. The VDSS uses information retrieved by the rest of the components of the VDSS from different parts of the ANASTACIA framework. Its objective is to evaluate the mitigations that are capable of reacting to an incident, considering aspects such as the impact of enforcing the mitigation, the type of device affected by the incident, its criticality within the infrastructure or the type of incident to mitigate. The result is a list

of mitigations ranked by a score which represents the suitability of the mitigation and considering the specific characteristics of the IoT infrastructure.

- VDSS database, which stores information required by the VDSS Engine model, such as impact of mitigations, criticality of devices, available mitigations or alerts.
- The Backend Controller, which orchestrates the reception of information retrieved from the Orchestrator, to be used either by the GUI, the VDSS Engine Model or to interact with the VDSS database.

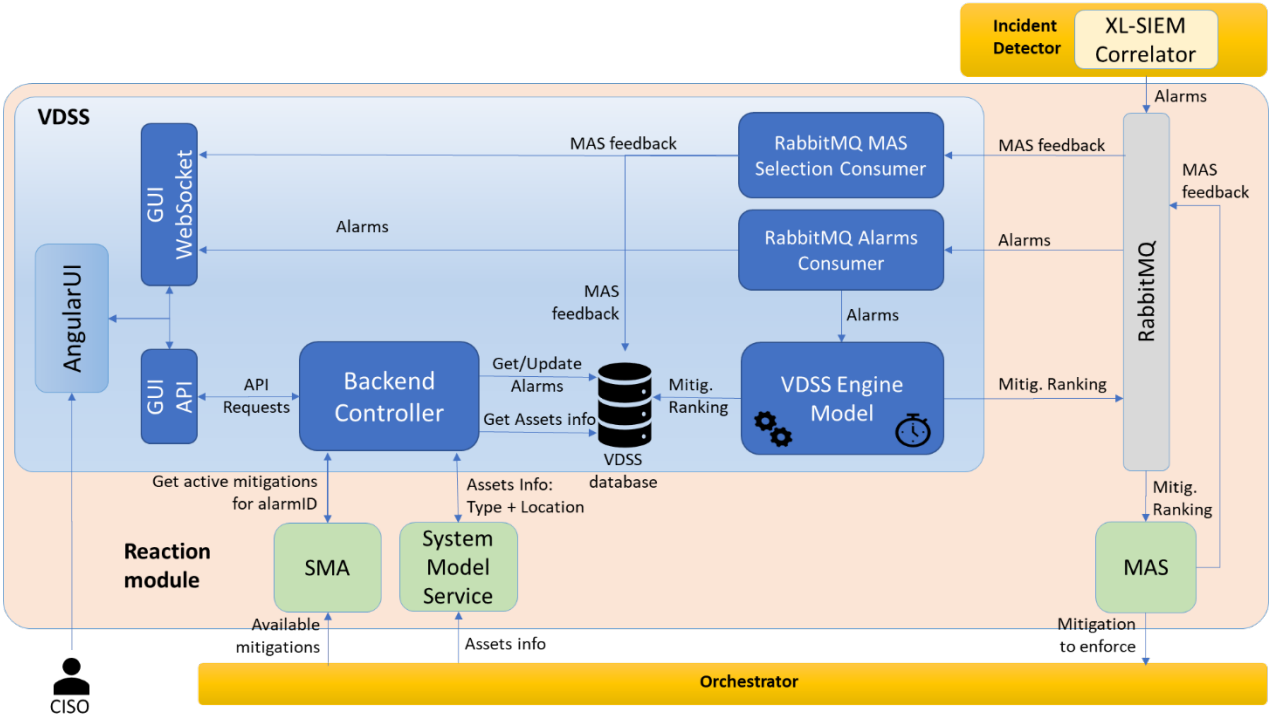

Figure 4. VDSS internal architecture

2.2.1.2 VDSS internal workflow

The VDSS can be separated into two main workflows: ranking generation and mitigation triggering. Both workflows are depicted in Figure 5.

Mitigation ranking generation

During this step all the information required by the VDSS to evaluate and recommend mitigations are collected from different parts of the ANASTACIA platform:

- Alert generated by the Incident Detector. This is the key element and the one used to retrieve the rest of the information needed. The alert contains information about the affected device, the type of incident and the risk associated to the incident which is calculated by the XL-SIEM at the Incident Detector.
- The type of incident is used to obtain the list of possible mitigations from the Orchestrator. This is, for every incident detected the orchestrator notifies about the potential mitigation strategies that are capable of mitigating such incident. With the list of mitigations, it is calculated the impact of every mitigation, supported by the CISO that set cost scores (amount of resources required to deploy such mitigation) and impact scores (how the mitigation impact on the infrastructure, i.e., turning off a complete network segment would have higher impact than just restarting a device).
- The affected device, given in general by its IP address, is used to calculate the criticality of the device affected. The VDSS retrieve information about the location of the device or the type of device to set a score that represents the criticality of the device.

These inputs are used by the VDSS to calculate a Suitability score that represents how convenient is to apply such mitigation to remediate the detected incident over a certain device. The result is a ranking of mitigations sorted by the suitability score.

Mitigation triggering

Once obtained the ranking there are two possibilities, which depends on the CISO:

- Mitigations labelled as automatic are directly enforced without CISO intervention.
- Non-automatic mitigations are notified to the CISO, which has x seconds to select the mitigation among the ones evaluated by the VDSS. The CISO can check the mitigation ranking and all the information related to every mitigation and decide on the mitigation to enforce.
- In case the CISO does not respond in those x seconds the mitigation with a higher score in the ranking is enforced.

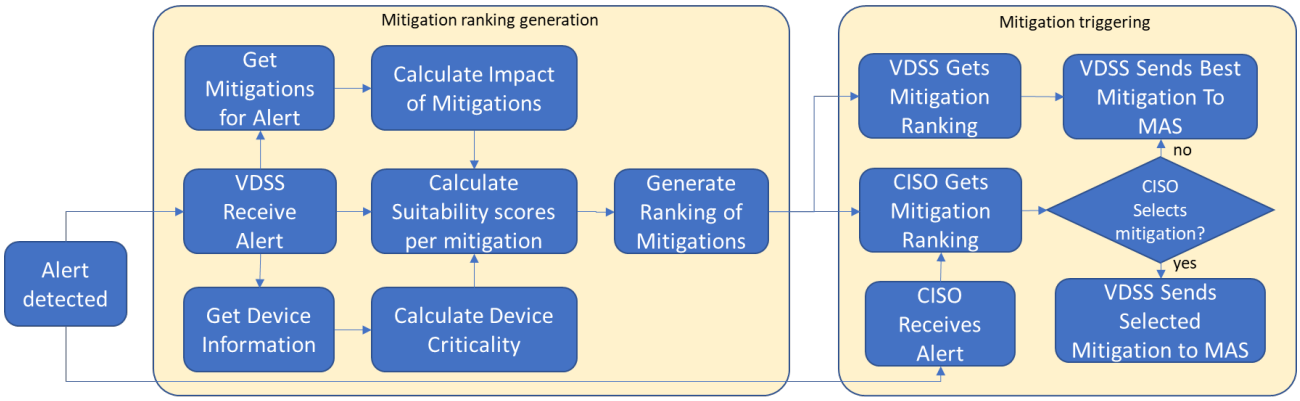

Figure 5. VDSS internal workflow

Next subsections describe, firstly, the process to calculate the suitability score. Then it is described the processes to gather inputs from the CISO GUI, and from the orchestration, the triggering of mitigations based on the ranking generated and the process to provide feedback about enforced mitigations to the CISO.

2.2.1.2.1 Suitability score calculation

The main objective of the VDSS is the calculation of scores for the mitigations that are capable of mitigating an ongoing incident. This score represents the suitability of the mitigation to the incident detected and is tailored to the context where the incident occurred, such as the type of incident, the type of device affected but also variables that represents the cost of enforcing the mitigation or the impact that enforcing it might have on the infrastructure. To this end, it is very relevant the role of the CISO as many of these aspects are defined by her considering business aspects such as the importance of certain devices with respect to others, economic priorities or simply expert knowledge.

Three main parameters are used by the VDSS to calculate the suitability score of a mitigation:

- Global impact of the mitigation within the infrastructure. To this end, we have to take into account what parts a mitigation is composed of. Rather than just one action per mitigation, a mitigation can be defined as a strategy that is composed of one or more actions (steps) against a certain target. We calculate the impact of a mitigation strategy by aggregating the impact of every action against a target. The impact of every action is calculated as a combination of two values. The first value represents the cost of enforcing it, which refers not just monetary aspects but also the usage of computational or human resources. The other value used represents the impact of the mitigation, which depends on aspects such as the number of devices affected by the mitigation (i.e., an individual device vs all the devices of a subnet), the type of actions carried by the mitigation (i.e., turning off vs restarting). It is also considered the impact of the affected target by an action. This information is calculated automatically based on input inserted by the CISO. The following table represents some examples of actions and qualitative impacts:

Mitigation strategy	Action	Cost of action	Impact of action	Target	Impact of target
Mitigation Step 1	Turn off	Low	High	Affected device	Low
Mitigation Step 2	Turn off	Low	High	Complete subnet	High
Mitigation Step 3	Deploy honeynet	High	Low	Compete subnet	Low

Step 1 and Step 2 entail to turn off devices. In this case, turning off a device is something with a minimum cost in term of resources. However, the impact is high, as the device would be out of service. However, Action 1 requires turning off simply just the affected device, therefore, its impact is low, while Step 2 requires turning off a complete subnet of devices, entailing a high impact. Finally, Step 3 entails to deploy a honeynet which requires high computational resources, although the impacts are low as the infrastructure is not affected by this action.

To calculate the impact of a mitigation strategy we calculate the impacts of every action it is composed of. We calculate separately scores for the cost and for the target. Therefore, considering that a mitigation strategy (M) is composed of one of more steps (MS):

$$M = \{MS_1 ... MS_n\}$$

And every mitigation step MS_i, is composed of an action A_i and a target T_i:

$$MSi = \{A_i, T_i\}$$

For every mitigation step it is defined its Global Action Impact (GAI) by combining scores for its cost and the impact of such action:

$$GIA_{MSi} = \frac{CI + AI}{2}$$

being CI the Cost Impact and AI the Action Impact given by the CISO through the VDSS GUI. As described above, the CI represents the usage of resources required to perform such action (computational resources, time, human effort, etc), while the AI represents the impact of such action on the service provided within the infrastructure.

Considering that TI (Target Impact) is the Impact of the target defined for a mitigation step (this is, if the target is a single device its impact would be lower than when the target is a complete subnet), we can calculate the impact of a Mitigation Step, MS_i, by aggregating the scores of the GIA and the TI:

$$\left[IMPACT = \frac{GIA + TI}{2}\right]_{MSi}$$

Finally, we can calculate the global Impact of a complete Mitigation Strategy (IMS) by aggregating the individual scores of every mitigation step. For example, having n mitigation steps for a given mitigation strategy, the IMS for the complete mitigation strategy will be:

$$IMS = \frac{1}{n}\sum_{n}^{i=1} IMPACT_{MSi}$$

Values for CI, AI and TI are given by the CISO through the GUI (Figure 6), assigning scores on a given scale 1…10, being 1 the lowest importance and 10 the highest one.

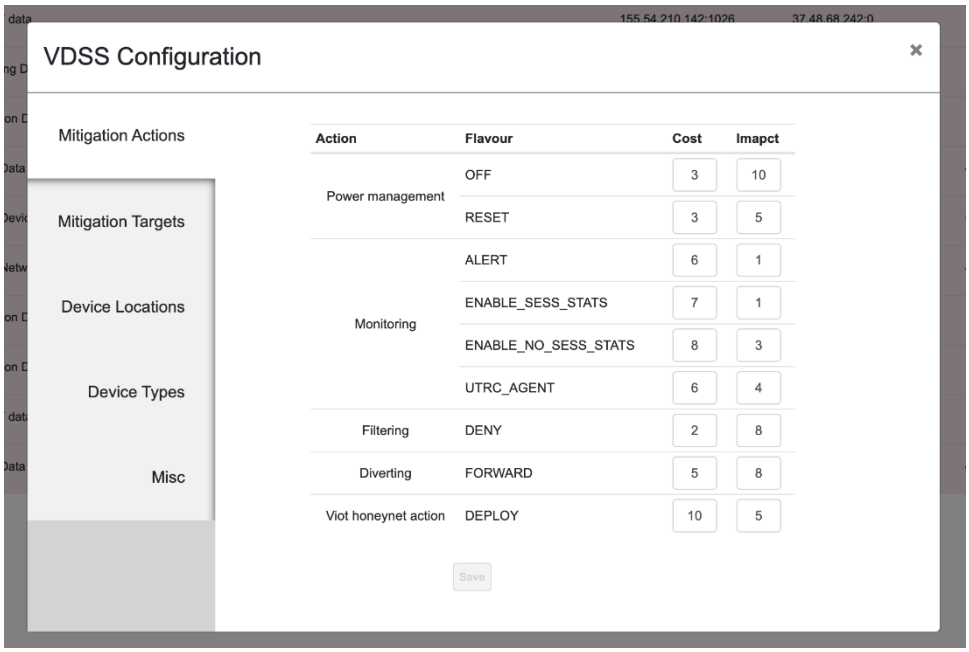

Figure 6. GUI for configuration of mitigation strategies scores

- Criticality of the device affected by the incident. This information is basically inserted by the CISO as a combination of values that are related to business aspects. The number of business aspects used to calculate this criticality can be very wide, for example, economic importance of the asset, societal criticality of the asset (i.e., whether human lives depends in its correct operation) or the importance of where it is running (i.e., critical parts of a building vs less important ones). To this end, the location and the type of device affected by an incident are retrieved through several interfaces implemented between the Reaction module and the Orchestrator module. Both location and type of device can be assigned with importance scores either predefined or adapted by the CISO. These scores are combined to compute the criticality of the device affected. An example is represented in the following table:

Device	Location (importance)	Type (importance)	Criticality
Device (ip)	Location Importance $LI = \{1...10\}$	Device Type Importance $DTI = \{1...10\}$	$C = \frac{1}{n}\sum_{n}^{i=1} Y_i$ Being Y_i the number of aspects used to calculate the criticality. For the example, if we consider the Location and the Type of Device, the device criticality results: $C = \frac{LI + DTI}{2}$

- Risk associated to the alert. This information is calculated automatically by the Incident Detector at the XL-SIEM. It takes values managed by the XL-SIEM which represents the priority of the events that has resulted into an alert, and the reliability of the events produced by the security probe that has generated the evidences of an incident. To calculate the risk, along with the aforementioned priority and reliability, it is also used the importance of the asset, which is obtained by considering the Location and Type of device. For every location and type the CISO can give an importance score by using the GUI, just as shown in the following screenshot.

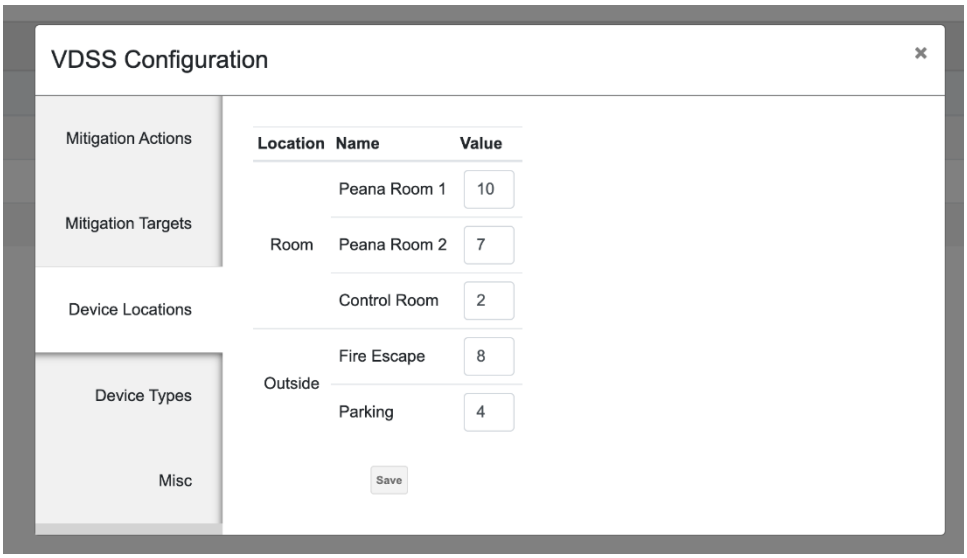

Figure 7. GUI to insert location and target importance for a device

Therefore, for every mitigation we have three scores:

$$MIT = (IMS, C, R(C)),$$

being as defined above, IMS the Impact of the Mitigation Strategy, C the Criticality of the devices affected by the incident and R(C) the risk associated to the incident, which depends on the device criticality C.

Finally, the Suitability score SU of the mitigation combines these three values:

$$SU = \frac{\sum_{n}^{i=1} MIT_i w_i}{\sum_{1}^{n} w_i}$$

where w_i are optional weights given to the aforementioned parameters which can be used to give more importance to certain values with respect to others.

2.2.1.2.2 CISO VDSS configuration

As described in the previous section, the CISO has a GUI that can be used to define importance scores to impact and costs for mitigations, actions and devices. Although default scores can be set by default, we have preferred to give the option that the CISO uses her expert knowledge to finetune the evaluation of the VDSS.

The Global Action Impact (GAI) is calculated using Cost Impact (CI) and Action Impact (AI). CI and AI are defined by the CISO using the process depicted in Figure 8. To this end, the CISO GUI requests the list of mitigation action steps to the Backend Controller, using the REST API with *GET /mitigation-actions/.* The Backend controller requests this information to the SMA through a REST API with GET /action/. The information is returned as an array of JSON data, one per action requested. This information might have changed with respect to the information that the VDSS already has from previous interactions, so it firstly update such information and retrieve the current CI and AI values from the VDSS database. The array of actions, with their impact and cost is returned to the CISO who can modify the CI and AI and updated at the VDSS using a REST API with *PUT /mitigation-actions/.* This information is then updated in the VDSS database.

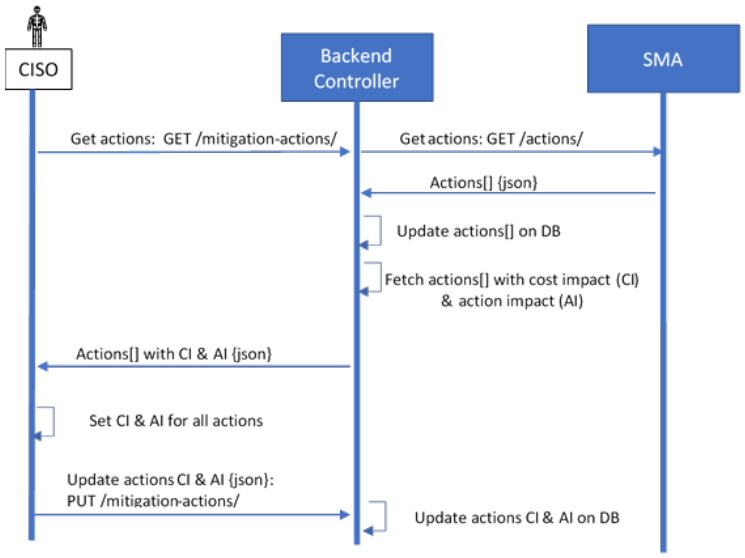

Figure 8. Process to define impact and cost for actions

Next process describes the way that the CISO can specify Target Impacts (TI). This is described in Figure 8. Similar to the previous process, the CISO requests to the Backend controller the targets available for the mitigations with the REST API with *GET /mitigation-targets/*. This is the returned by the SMA as an array of JSON data for all the available mitigation targets. The CISO receives the available targets with the default Target Impact values already available at the VDSS database, which is updated by the CISO and returned back using a *PUT /mitigation-targets/* at the Backend Controller.

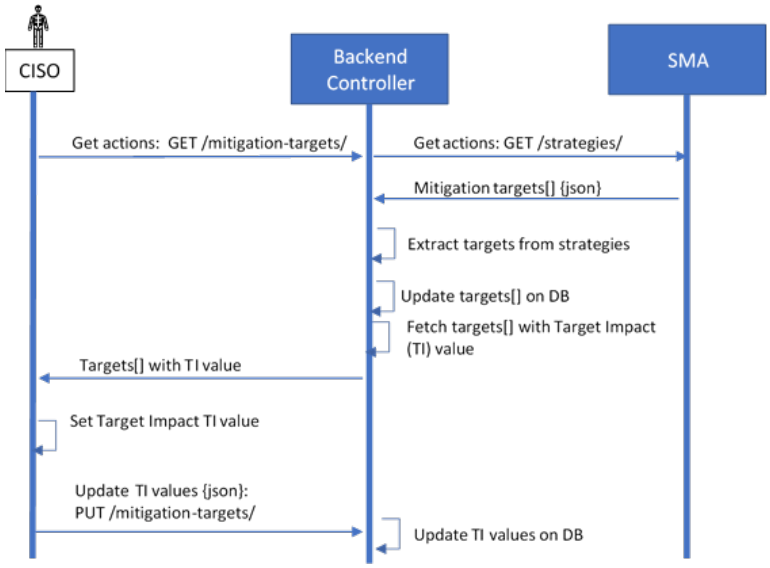

Figure 9. Process to define target impacts

Similarly the CISO can set criticality scores both for the location of the devices and for the type of devices. These processes, depicted in Figure 9, are similar to the previous ones, and results for updated values for LI and DTI for every mitigation action step of every mitigation strategy.

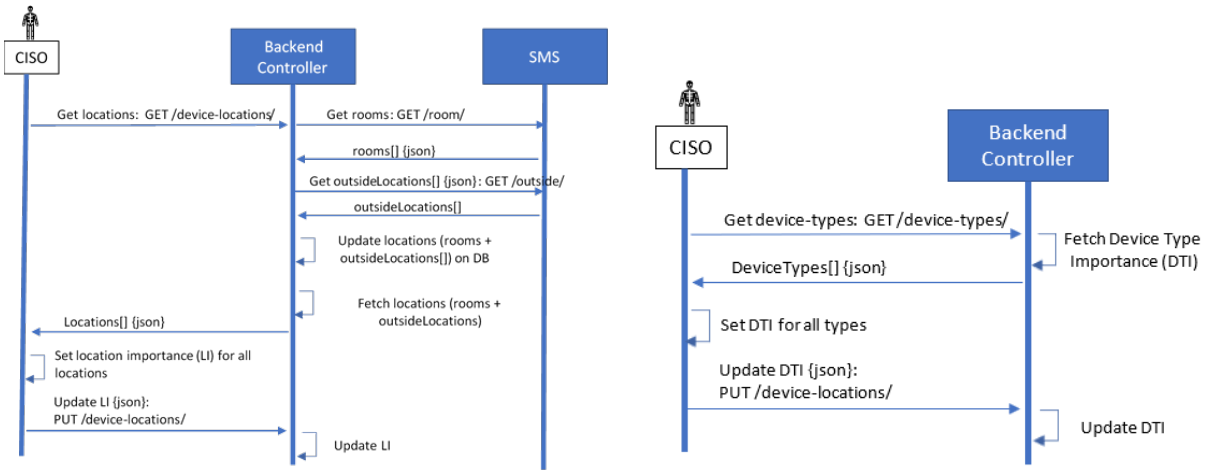

Figure 10. Process to set Location Importance (left) and device type importance (right) of devices by CISO

2.2.1.2.3 Mitigation ranking generation

Figure 11 depicts the sequence diagram of the process to retrieve the input required by the VDSS to generate the mitigation ranking. These inputs are obtained by extracting information from different components of the ANASTACIA platform. This input is generally obtained through asynchronous messaging queues using a RabbitMQ server or through REST APIs. The orchestration of the communications between the VDSS Engine model and the rest of the components are done through the Backend Controller. Alerts are pushed to a RabbitMQ message queue (called *exchange_alarm*) which are retrieved in real time by the VDSS. This alert is exported in a JSON format as described in Section 2.2.1.3. The alert is used by the VDSS to retrieve further information, namely information about devices and details about mitigations.

Information about available mitigations are obtained from the System Model Service by using a REST API. The VDSS requests, through the Backend Controller, the list of mitigations capable of mitigating the incident detected by the Incident Detector. To do so it is used the method "*getMitigationbyThreat*" of the Backend Controller, including the id of the alert as argument. The Backend Controller invokes the SMA, which has up to date information about available mitigations that the orchestrator is capable to enforce. The invocation is done using a REST API pointing to *GET /strategies/threat/{alertid}*. The list of available mitigations is returned as a JSON that is returned to the VDSS Engine Model. For every mitigation contained in such JSON it is retrieved the associated impact, which has been calculated by using the mechanism described in Section 2.2.1.2.1 and stored in a database. After this process the VDSS Engine Model has compiled the list of suitable mitigations and the impact associated to every of them.

Next step triggers the processes to obtain information about the affected devices. The affected devices are identified by its IP address which is included in the alert triggered by the Incident Detector. With this information the VDSS obtains the location and type of device from the System Model Service which offers this information through a REST API at */PNF/?ip=deviceip*. Using the location and the type of device it can be calculated the criticality of the device which is calculated by the VDSS Engine Model using information inserted by the CISO as described in section 2.2.1.2.2

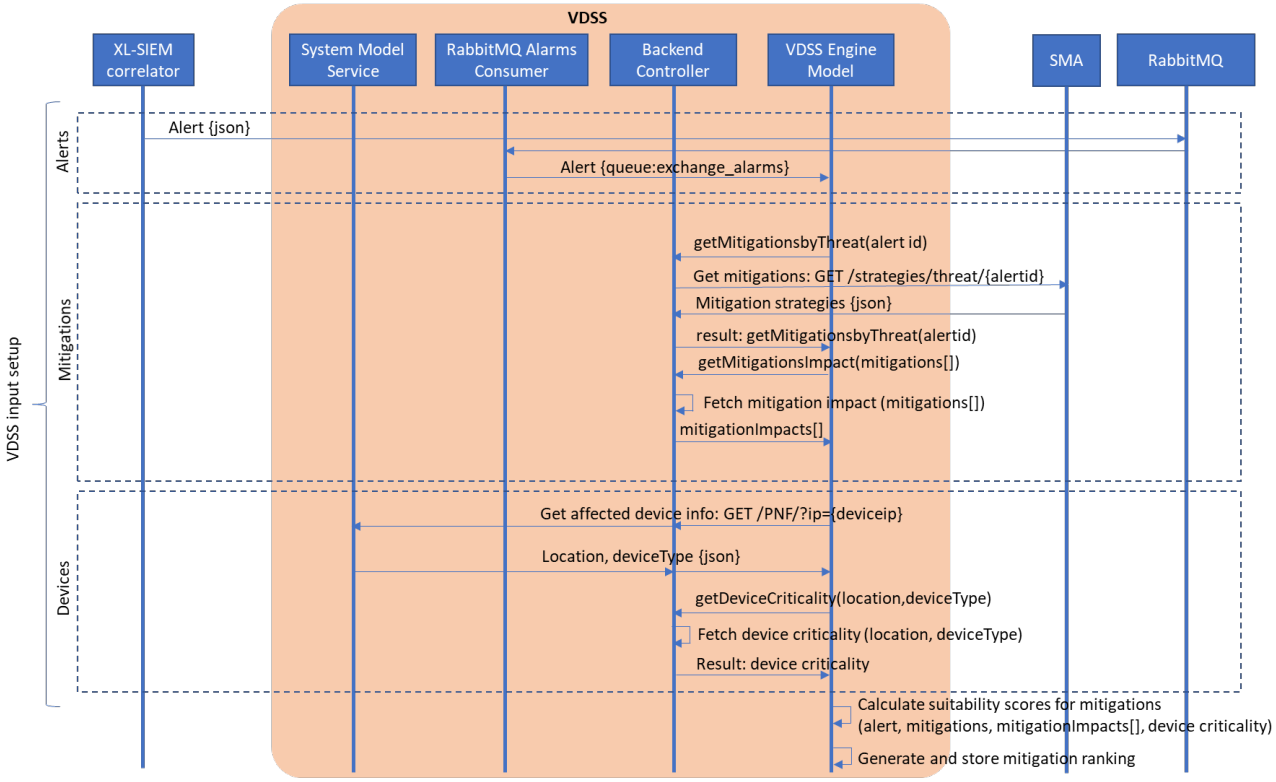

Figure 11. VDSS Input setup

Once compiled the information required (alert, mitigations, impact of mitigations and device criticality), the VDSS calculates the score for every mitigation and generates the ranking that is internally stored in a database and used in the next phase of the mitigation recommendation process.

2.2.1.2.4 Mitigation triggering and feedback

In the second phase it is used the ranking of mitigations to determine which is the mitigation finally enforced by the orchestrator. There are several possibilities, depending whether the mitigations are labelled as automatic, this is, to be deployed without human conformation, or not. Here we will detail the case of mitigations not labelled as automatic. In this case there are two options:

- The CISO can confirm the enforcement of the mitigation, or even select a new one. To do so, the CISO is alerted about a new incident in real time via WebSocket. The CISO dashboard retrieve the ranking of mitigations, calculated by the VDSS, using a REST API (*GET /mitigations/*). The ranking is returned as a JSON and the CISO can select which one to enforce. The selected mitigation is returned to the Backend Controller using a PUT method of the REST API, through */alarm-mitigations/{alarms-mitigations}/select/{value}*. The selected mitigation is finally pushed to the MAS through the *exchange_recommendations* queue of a RabbitMQ server.
- The other possibility is that the CISO does not select manually a mitigation. In this case, the ranking of mitigations is submitted to the MAS using the *exchange_recommendations* queue. The mitigation with a better score is automatically selected by the MAS for its enforcement.

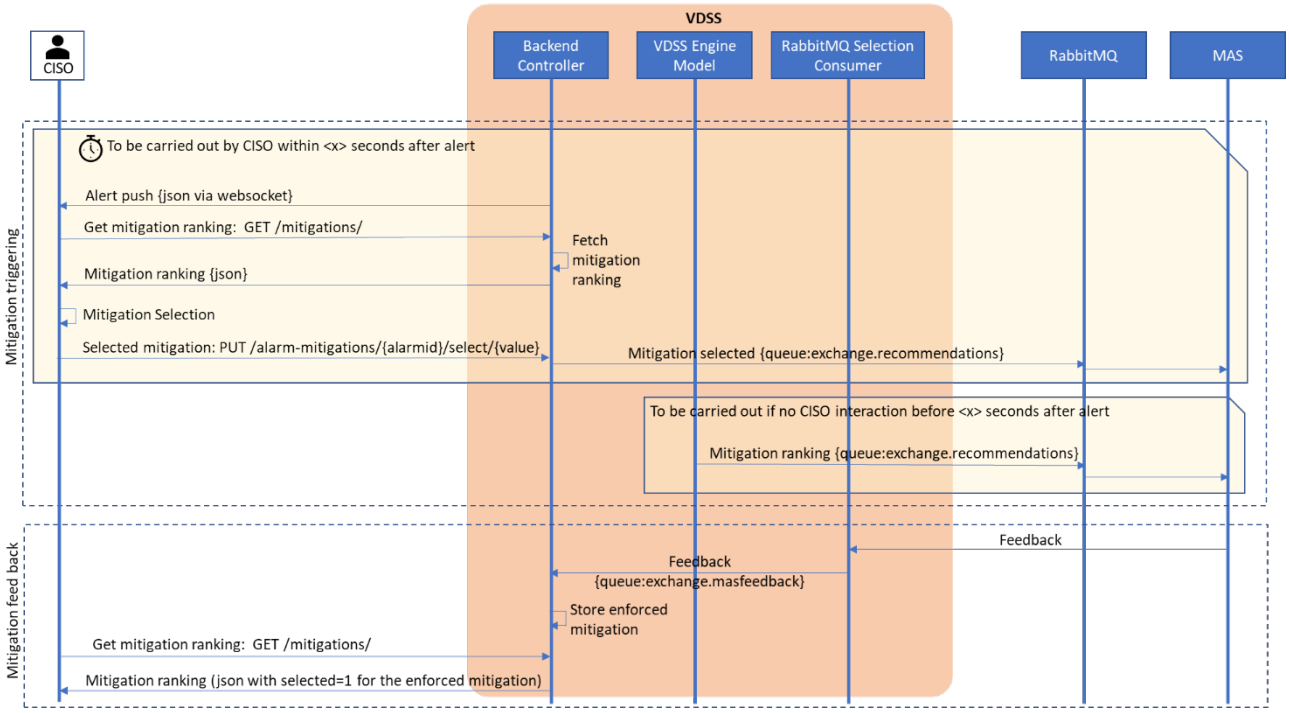

Figure 12. VDSS: Mitigation and feedback

An additional step can be carried out to submit feedback to the CISO about the mitigation enforced. To do so the MAS, through the RabbitMQ server, using the queue *exchange_masfeedback*, notifies to the backend controller, which stores the enforced mitigation in a data base. Whenever the CISO asks for this information the CISO dashboard, using a REST API, asks for the enforced mitigations using a *GET /mitigations/*. The Backend controller extracts this information from the database returning a json with the list of mitigations, having the label "selected" as 1 for the ones that has been enforced.

2.2.1.3 VDSS inputs

This section describes the format of the data received or retrieved by the VDSS and used for the calculation of the suitability scores and the generation of the ranking of mitigations.

Table 1 describes the format of the security alerts as exported by the Incident Detector (XL-SIEM).

Table 1. Data format for security alerts

Security alerts	
Source component	Incident Detector (XL-SIEM)
Data format	JSON
Interface	RabbitMQ – Exchange "eu.anastacia.siem_server_output "
Relevant information	- Type of incident - Targeted devices (IP) - Risk associated to the incident
Data sample	{ "AlarmEvent": { "DST_IP_HOSTNAME": "00000000", "RELATED_EVENTS": "[805911e9b8d5080027ea052c62fedfc2]", "DST_IP": "aaaa::2", "PLUGIN_NAME": "cyber-monitor", "SRC_IP": "aaaa::1", "PRIORITY": 4,

```
    "RELIABILITY": 6,
    "SUBCATEGORY": "Bruteforce",
    "USERDATA3": "",
    "USERDATA4": "",
    "PLUGIN_SID": "5",
    "USERDATA1": "PAA",
    "USERDATA2": "",
    "ORGANIZATION": "ATOS",
    "CATEGORY": "Authentication",
    "PLUGIN_ID": "70000",
    "USERNAME": "",
    "FILENAME": "",
    "BACKLOG_ID": "06234b5e4a05470c8e0413903edfa62a",
    "PROTOCOL": 6,
    "RISK": 4,
    "SRC_PORT": 0,
    "SENSOR": "",
    "SRC_IP_HOSTNAME": "00000000",
    "SID_NAME": "AAA Probe - Forbidden Network Authentication",
    "USERDATA7": "",
    "DATE": "2019-05-27 08:28:21",
    "USERDATA8": "",
    "USERDATA5": "",
    "USERDATA6": "",
    "PASSWORD": "",
    "USERDATA9": "",
    "DST_PORT": 716,
    "EVENT_ID": "340ff546b48b4146a0445ab3e0ac641c",
    "RELATED_EVENTS_INFO": {
     "a": {
      "date": "1558945701",
      "plugin_id": 31000,
      "log": "Ik1heSAyNyAwODoyODoyM.....ibmEifSAi",
      "interface": "enp0s3",
      "dst_ip": "aaaa::2",
      "src_ip": "aaaa::1",
      "userdata7": null,
      "fdate": "2019-05-27 08:28:21",
      "userdata8": null,
      "userdata5": null,
      "userdata6": null,
      "userdata9": null,
      "userdata3": null,
      "userdata4": null,
      "userdata1": "PAA",
      "userdata2": null,
      "src_port": null,
      "plugin_sid": 1,
      "event_id": "805911e9b8d5080027ea052c62fedfc2",
      "filename": null,
      "organization": "ATOS",
      "dst_port": 716,
      "tzone": null,
      "device": "10.0.2.4",
      "username": null
     }
    },
   }
  }
```


Table 2 describes the format of the data used to describe devices, which is offered by the Orchestrator and consumed by the VDSS.

Table 2. Device information data format

Infrastructure data – Device information	
Source component	Security Orchestrator (Security Model Service)

Data format	JSON
Interface	REST API (Base path: *http://195.148.125.100:9000/api/*) Get device info: - IPv4: GET /PNF/?ip={device_ip} - IPv6: GET /PNF/?ipv6g={device_ip}
Relevant information	- Device location (room or external location) - IP address - Type of device

Data sample	{
	"count": 0,
	"next": "string",
	"previous": "string",
	"results": [
	{
	"id": 0,
	"outside": {
	"id": 0,
	"latitude": "string",
	"longitude": "string",
	"zone": "string",
	"description": "string",
	"pnf": "string"
	},
	"type": "1",
	"state": "1",
	"name": "string",
	"tag": "string",
	"model": "string",
	"description": "string",
	"alias": "string",
	"room": "string",
	"softwares": [
	0
	],
	"cloud": [
	0
	],
	"resources": [
	0
	]
	}
	]
	}

Another information used by the VDSS is the available mitigation strategies capable of reacting to a certain incident. Table 3 describes it.

Table 3. Available mitigation strategies data format

Available mitigation strategies	
Source component	Asset Model (AM)
Data format	JSON
Interface	REST API (Base path: *https://212.101.173.75:5002*) Get mitigation strategies: /strategies/ Get mitigation actions: /actions/ Get mitigation capabilities: /capabilities/
Relevant information	- Mitigation strategies - Suitable threats per mitigation - Capabilities per mitigation - Mitigation actions and flavours

	- Action targets

Data sample	GET /strategies/:
	[
	{
	"id": 1,
	"name": "Filter_And_Deploy_UTRC",
	"impact": 10,
	"cost": 10,
	"automatic": 1,
	"availability": 1,
	"threats": [6, 7, 8],
	"mitigations": [
	{
	"capabiltiy_id": 32,
	"action_id": 3,
	"flavour_id": 1,
	"instance": 1,
	"target": {
	"device": "affected"
	}
	},
	{
	"capabiltiy_id": 20,
	"action_id": 2,
	"flavour_id": 4,
	"instance": 2,
	"target": {
	"network": "affected"
	}
	}
	]
	},
	….]
	
	GET /actions/:
	[
	{
	"id": 1,
	"name": "power_management",
	"flavors": [
	{
	"id": 1,
	"name": "OFF"
	},
	{
	"id": 2,
	"name": "RESET"
	}
	]
	},
	….]
	
	GET /capabilities/:
	[
	{
	"id": 19,
	"name": "IoT_Control",
	"availability": 1,
	"actions": [
	1
	]
	},
	…]

Finally, another piece of information is used to provide feedback about mitigations. This is described in Table 4.

Table 4. Data format used for feedback about mitigations

Selected Mitigation Strategy	
Source component	Mitigation Action Service (MAS)
Data format	JSON
Interface	RabbitMQ – Exchange “exchange.masfeedback”
Relevant information	Information about the selected mitigation strategy by the MAS component. The selection is stored to be shown in the VDSS UI.
Data sample	{ "alarm_id": " 4fc45fae13444ed9a2efc3a845619431", "mitigation_id": "4", "timestamp": "1567681220" }

2.2.1.4 VDSS outputs

The VDSS sends a list of scored mitigations to the Mitigation Action Service. This output data is sent via RabbitMQ exchange to provide a real-time notification to the MAS. The output data consists on a JSON with 3 main keys:

- **alarm**: Information about the related alarm.
- **user_selection**: Contains the ID of the mitigation strategy selected by the CISO using the VDSS UI.
- **mitigations**: Array with the scored mitigations.
    - suitability_score: Score calculated by the VDSS.
    - mitigation_id: ID of the mitigation.
    - user_selection: 1 means that this mitigation has been selected by the user; 0 means that the user hasn’t selected any mitigation.

List of scored mitigations	
Consumer component	Mitigation Action Service
Data format	JSON
Interface	RabbitMQ – Exchange “exchange.recommendations”
Relevant information	- Related alarm information - Mitigations scores - User selected mitigation
Data sample	{ **"alarm"**: { "related_events": "[813911e9b8d5080027ea052cb7dca28c]", "dst_ip": "0.0.0.0", "src_ip": "0.0.0.0", "subcategory": "ACL_Deny", "plugin_sid": "6", "plugin_id": "70000", "organization": "ATOS", "backlog_id": "7287bfca7e9d426dbb63f5ef911d6fd9", … … "reaction_timestamp_limit": "1559042111", "mitigated": true, "selectedMitigationID": 2, "mitigable": true },

```
"user_selection": "2",
"mitigations": [
  {
   "suitability_score": 9.73932,
   "id": "2",
   "user_selection": "1"
  },
  {
   "suitability_score": 4.9746,
   "id": "3",
   "user_selection": "0"
  },
  {...}, ...]
}
```


2.2.2 Security Alert Service

The Security Alert Service is the component of the Reaction module adopted to combine information generated from the other components of the module, group them together and propagate/send them to the **DSPS**.

The **SAS** module links the detailed information about the detected threat with the policies that the other components of the module have selected to protect the system from the detected threat. The adoption of this strategies will therefore lead to a link between the identified threat and the selected mitigation. The module communicates with the components as shown in Table 5.

Abbreviation	Alias	Component	Developing partner	Communication	SSL
VDSS	XL-SIEM	Verdicts and Decision Support System	ATOS	Dedicated RabbitMQ queue	Yes
MAS	-	Mitigation Action Service	MONTIMAGE	Web Server	No
DSPS	-	Dynamic Security and Privacy Seal	DG, MANDAT, ARCHIMEDE	Dedicated RabbitMQ queue	Yes

Table 5 - Communication of the Security Alert Service

In detail, the communication between the **SAS** and **VDSS** modules is based on a dedicated RabbitMQ queue, where the **SAS** listens to the queue waiting for a threat detected by the Monitoring module and propagated to the **SAS** through the **VDSS**.

After receiving detailed information about the detected threat from the **VDSS**, before propagating it to the **DSPS**, the **SAS** component waits for updates from other involved components, which should trigger specific activities as a consequence of the detection, to identify, select and automatically deploy the countermeasures able to mitigate the identified threat. When the activities are completed, the **MAS** is programmed to send the selected policies through a Web based communication to the **SAS**.

The **SAS** then combines the threat information, initially received from the **VDSS**, with the policies received by the **MAS** and sends the combined/enriched information, through another dedicated RabbitMQ queue, to the **DSPS**. If no additional information is received by the **MAS** within a pre-defined timeout, the initial data received by the **VDSS** is forwarded to the **DSPS**, specifying that an incomplete (hence, not enriched) alert is sent. If the additional data related to the same alert are received by the **MAS** after the expiration of the timeout, the enriched alert is sent anyway (in late) to the **DSPS**.

In particular, the **SAS** component combines the alert received from the **VDSS** with policies and deploy information received from the **MAS**. Produced output directed to the **DSPS** component has the format reported below.

```
{
    <alert data, as received by the VDSS component>,
    'status_complete': <a boolean value specifying if the alert is enriched or not with the data received by the MAS component>,
    'policy': <policy data, as received by the MAS component (optional, only if status_complete is true)>,
    'request_id': <deployment identifier, as received by the MAS component (optional, only if status_complete is true)>
}
```

2.2.3 Mitigation Action Service

During the second part of the ANASTACIA project, the development of the Mitigation Action Service (MAS) is focused on extending the internal logics and the communications with new endpoints, shows the general design of the MAS deployed in the ANASTACIA platform. This Figure 13 only shows the main implemented classes (in blue) and it does not provide a comprehensive list of all the classes uses to implement the functionality mentioned below.

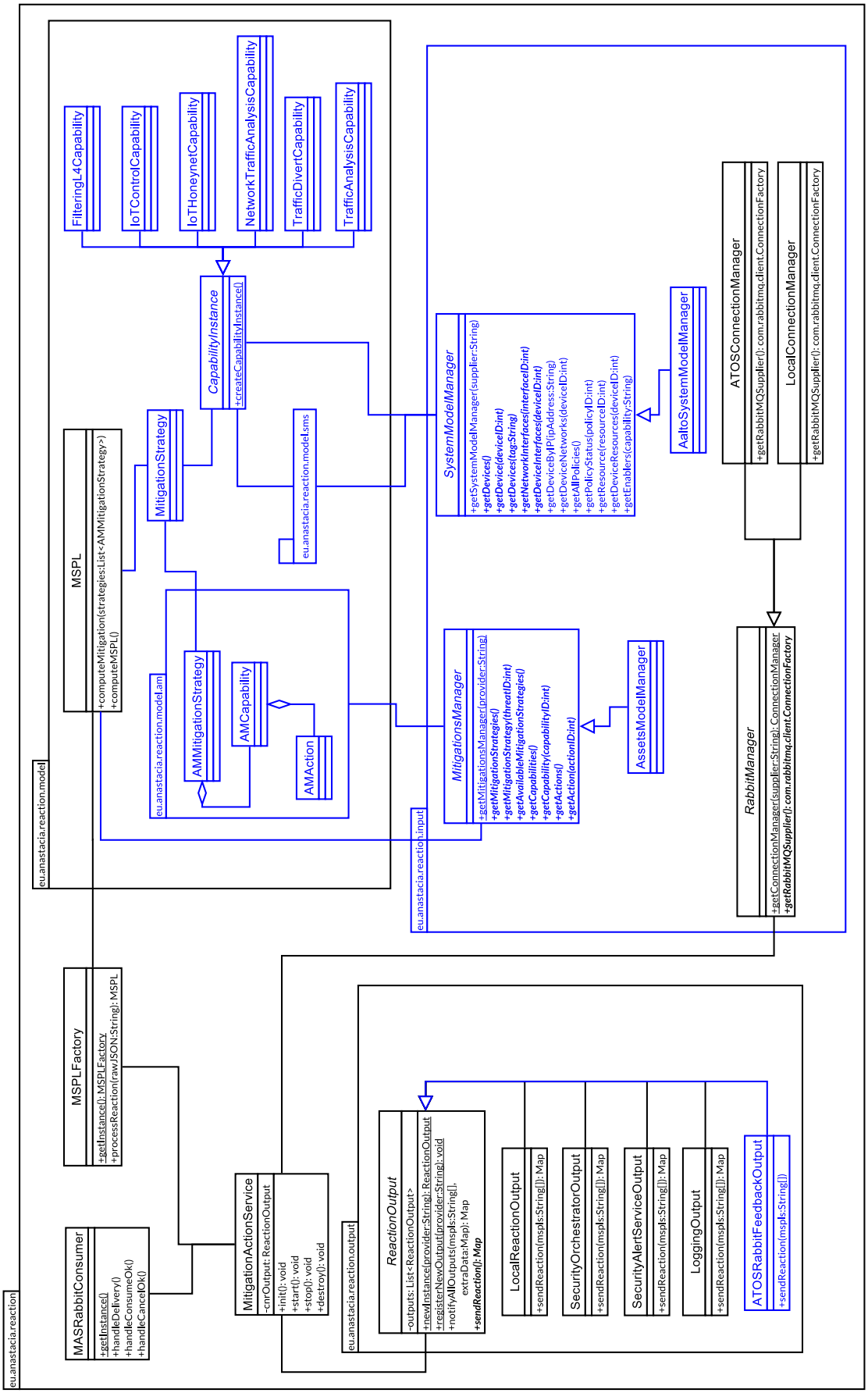

Figure 13 - Class diagram of the Mitigation Action Service

In particular, the implemented extensions allow the MAS to increase the awareness of the platform status in order to generate a mitigation plan for the detected security issues. The following sections give more details about the implemented extensions.

2.2.3.1 Generation of Countermeasures Based on Mitigation Strategies

The generation of countermeasures for any security or privacy issue is a particularly complex problem. Even for the system administrator manually selecting the appropriate countermeasure is not an easy task, depending on a set of variables that need to be considered. To this end, the ANASTACIA platform has engineered a solution that will aid both the system administrator and the MAS component to evaluate the possible countermeasures to deploy.

The *Mitigation Strategies* are the entities that have been created for this goal. They have been defined as a set of *Capabilities*, and *Actions* (including the specific *flavour* of action to perform) that can be performed and/or deployed in order to countermeasure a detected issue. In other words, these strategies define in an abstract way what type of security capability to apply, what particular actions to perform and on which devices, specifying them in an abstract way. Additionally, and in order to easy the automatic evaluation of each strategy, they also contain information about their respective cost and impact. Despite these two indicators are used during the automatic evaluation of the strategies (principally in the VDSS component), their values are assessed by the system administrator.

The MAS makes use of all the information contains in the mitigation strategies in order to generate the respective countermeasures following the strategies specified by the system administrator. In this sense, the MAS will aggregate information from several sources (specifically from the Assets Model and System Model components) and tailor the countermeasure for the detected issue following the strategies defined. By means of this process, the MAS adds value to the ANASTACIA platform by bringing awareness of the deployment in order to adjust the applied countermeasures for the network topology and using the available security enablers on the platform.

2.2.3.2 Integration with the Assets Model Service

The Assets Model (AM) Service has been conceived as a module to store information about *Mitigation Strategies* mentioned above. Considering the nature of the Mitigation Strategies, it is important for the MAS to gain knowledge about the strategies, since they will act as the base possible countermeasures to evaluate before executing.

To enable the communication Support of the AM Service, two principal subcomponents had to be implemented in the MAS: The Mitigations Manager Connector and the Assets Model data model. **Errore. L'origine riferimento non è stata trovata.** show where these classes where located in the general MAS design.

On one hand, the data stored in the AM Service had to be modelled in the MAS. To this end, three main classes were inserted with their respective fields. These classes are meant to store the data coming from the AM and provide a handy interface to access this information within the internals of the MAS.

On the other hand, the connection to the AM endpoint was designed to support multiple providers in case this is required. This was implemented by defining the *MitigationsManager* interface, which defines the abstract methods to retrieve the information from the data provider. In order to explicitly implement an Assets Model provider, a new *AssetsModelManager* class was implemented, which encloses all the specific calls to correctly send the requests to the AM endpoint, parse the response and create the model objects that contain the retrieved information.

Both components mentioned above are meant to be used when the MAS is generating the MSPL that contains the countermeasure. This process is depicted in Figure 14, showing the principal interactions between the members of the components and how the *MSPLFactory* class retrieves the list of Mitigations Strategies along with all the details about their availability, involved capabilities and actions.

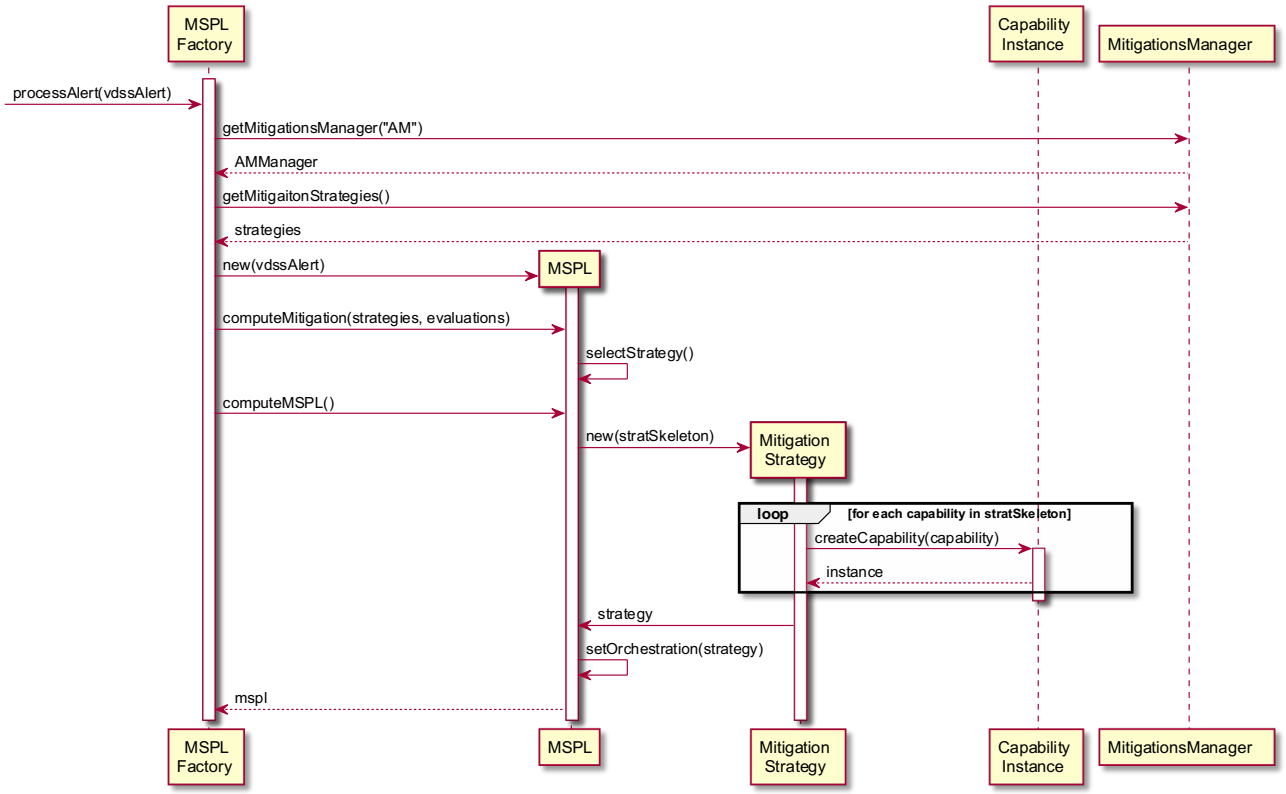

Figure 14 - Sequence Diagram for MSPL Generation using Mitigation Strategies

2.2.3.3 Integration with the System Model Service

As mentioned before, the objective of the MAS component is to provide a countermeasure that is tailored for the state of the network at the moment the security issue has been detected. Being this said, it is important for the MAS to gain knowledge about the status of the deployed system, in order to apply the *Mitigation Strategies* on the correct devices.

To this end, a set of classes and accessors have been implemented in the MAS that model all the entities that are exposed by the System Model's API. These classes are grouped in their respective package (eu.anatascia.reaction.model.sms, classes not shown to preserve the readability of the image), as it is noted in **Errore. L'origine riferimento non è stata trovata.**. The accessor classes have been integrated into the *AaltoSystemModelManager* class in order to perform the respective calls to the autogenerated code.

Finally, the model objects returned by the accessors are used in the MSPL generation code. This section of the module accesses the retrieved data (using the automatically generated classes) to enrich the countermeasures generation with the actual status of the System Model as reported by the Security Orchestrator. Figure 15 shows a particular example about how the MAS makes use of all the implemented classes in order to correctly interact with the System Model Service and craft the generated MSPL.

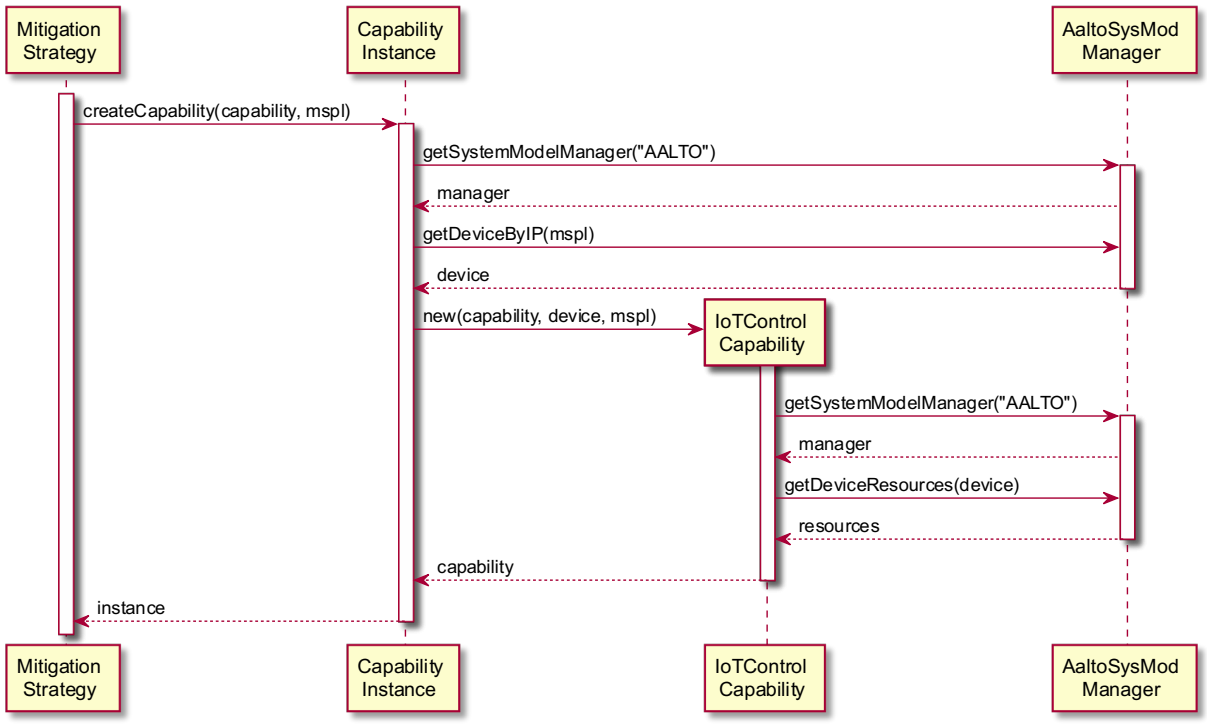

Figure 15 - Sequence Diagram to Generate an MSPL of IoT Control Capability

Figure 15 depicts the generation of an MSPL that codifies an IoT Control capability. In this particular example, the process starts when the *MitigationStrategy* class invokes the "createCapability" method. In this invocation, the *CapabilityInstance* class identifies the affected device by using its IP. This translation is made by calling the System Model Manager, which is in charge of retrieving the devices and compare their IP address with the affected one. It is important to remark that a capability might be applied not only on a device, but also a network or even a particular network flow. In this sense, the identification of the target (and the particular method used to get target information from the System Model) will vary depending on the nature of the target on which the policy is being applied. Later, when the IoT Control capability is being configured, the class needs to have knowledge about the resources (e.g. "power_off" or "reboot") of the device and how they can be triggered (i.e., particular URL, port and protocol). To this end, *IoTControlCapability* class queries the System Model in order to retrieve this information and finalise the configuration of the mitigation.

2.2.4 Asset Model

The name of the Security Model Analysis (SMA) the component of the Reaction module has been changed to the Assets Model (AS), which is more appropriate to its role. The role of this component is to continuously collect and update through the Kafka broker the existing capabilities, flavors, mitigations and their related threats and costs. In fact, the AS component retrieved from the Orchestrator the available flavors that are supported by the IoT infrastructure according to the security policy and the capabilities enforced, and collects from the VDSS the corresponding mitigations and their related threats. Moreover, it updates the cost the mitigations computed by the VDSS. Finally, the output will be sent by the AS to the MAS as a list of capability plus the information sent by the orchestrator to the MAS, the MAS generates the accurate MSPL file, the Figure 16**Errore. L'origine riferimento non è stata trovata.** shows the AS interactions, during this period the functionality defined in the D4.2 was implemented and tested. Moreover, the AS was enhanced by new functionalities.

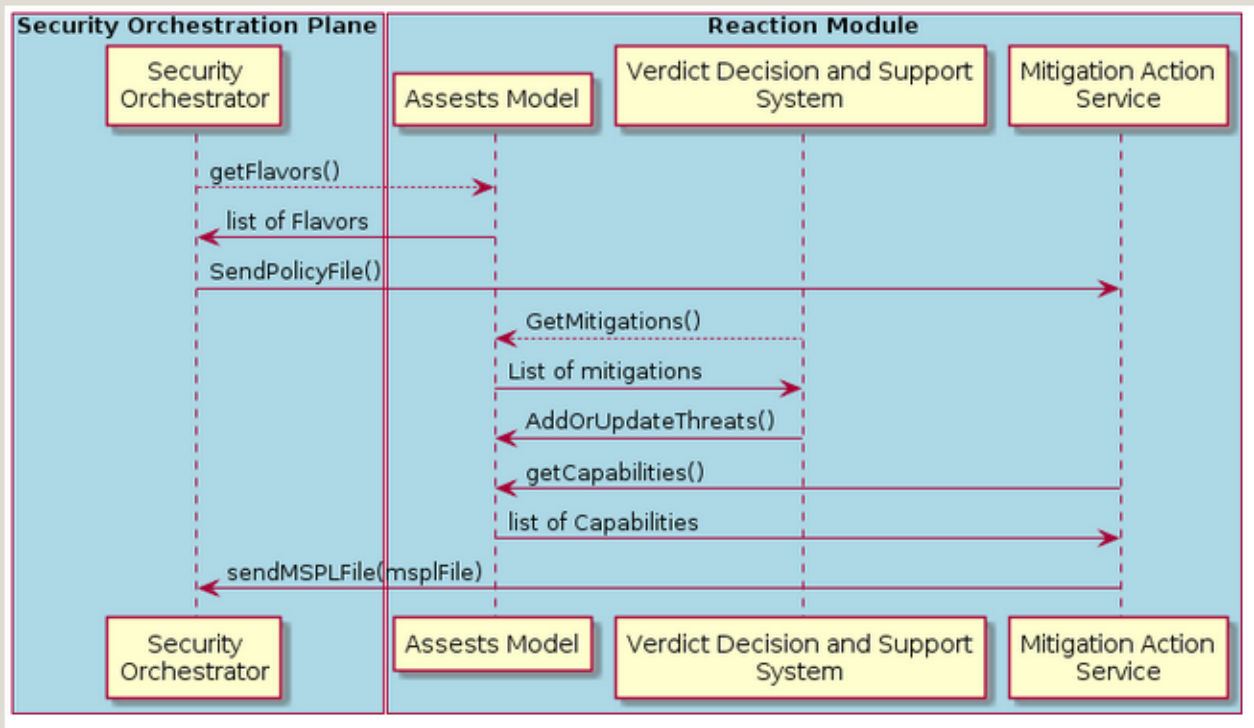

Figure 16 - The Security Model Analysis interactions

2.2.4.1 AS deployment

The component is deployed in wrapper fashion to simplify interactions between complex communication environment and computational part of AS. The data wrapper provides external connectivity to AALTO security orchestrator and the MAS via REST API, ATOS XL SIEM component using SSL secured RabbitMQ channel.

The operations of deployed AS component (Figure 17) are executed in four steps. During component initialization wrapper will request security models from SO component. Next AS component will start receiving analysing thread costs provided by ATOS XL SIEM component. After computing information AS will provide security model analysis that is then sent to ATOS XL-SIEM component on step 3. Lastly in step 4 ATOS based on the security model analysis and attack verdict messages will generate reactions that will be send to SO component for execution in SEP.

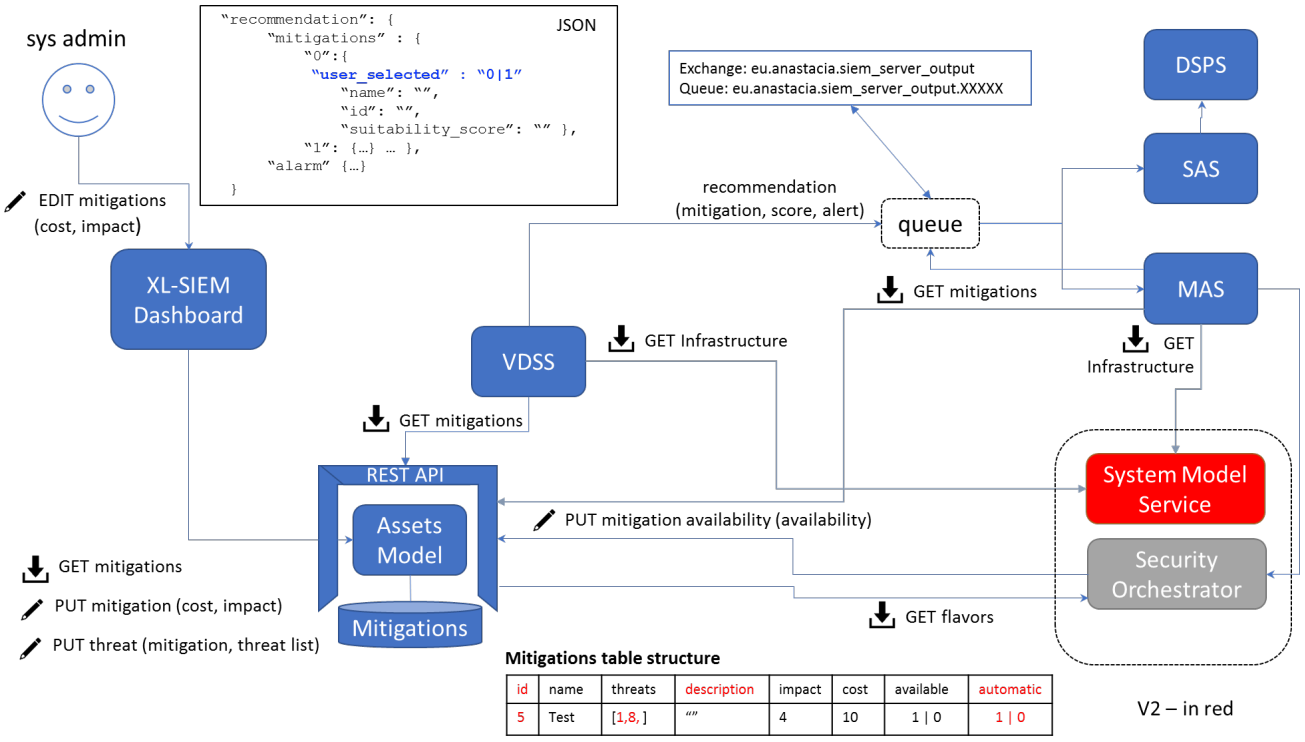

Figure 17 - Interaction between AS

In the following an example of VDSS – AS REST, and SO-AS API messages and the JSON file generated by the AS that aggregate the information provided by the other components.

GET available mitigations

```
Request:
GET https://212.101.173.75:5002/mitigations?available=1

Response:
[
    {
        "id": "",
        "name": "",
        "threats": "'1','2','3'",
        "description": "",
        "available": "1" ,
        "automatic": "0"
    },{ … }
]
```

GET all mitigations

```
Resquest:
GET https://212.101.173.75:5002/mitigations

Response:
[
   {
        "id": "",
        "name": "",
        "threats": "'1','2','3'",
        "description": "",
        "available": "0|1",
        "automatic": "0|1"
   },{ … }
]
```

GET single mitigation

```
Resquest:
GET https://212.101.173.75:5002/mitigations/{mitigation_id}

Response:
{
    "id": "",
    "name": "",
    "threats": "",
    "description": "",
    "available": "0|1",
    "automatic": "0|1",
}
```

GET mitigations for threat

```
Resquest:
GET https://212.101.173.75:5002/mitigations/threat/{threat_id}

Response:
[
    {
        "id": "",
        "name": "",
        "threats": "",
        "description": "",
        "available": "0|1",
        "automatic": "0|1"
    }, { ... }
]
```

UPDATE mitigation

```
Request:
PUT https://212.101.173.75:5002/mitigations/{mitigation_id}
Data: {
        "cost":5,
        "impact":10,
        "threats": "[list of threats that needs to be added]",
        "automatic":0
      }
Response:
200 (OK)
```

Figure 18: VDSS – AS REST API messages

UPDATE mitigation availability

```
Request:
PUT https://212.101.173.75:5002/mitigations/

Data:
[
      {
          "id": "0",
          "available": "1"
       }
]

Response:
200 (OK)
```

Figure 19: SO – AS REST API messages

2.2.5 Available Capabilities

Currently the ANASTACIA framework is able to implement a series of countermeasures to protect the IoT networks from threats of different kinds and with different objectives. The possible countermeasures are reported in the following table.

Threats	Capabilities
Insider attack (malware)	"Authentication", "AuthoriseAccess_resource", "DTLS_protocol", "Filtering_L3", "Filtering_L4", "Traffic_Divert", "IoT_honeynet", "IoT_control"
Man In The Middle	"Authentication", "DTLS_protocol" "Anonimity"
SQL Injection	"Filtering_L3", "Filtering_L4", "Traffic_Divert", "IoT_control"
Denial of Service	"Filtering_L3", "Filtering_L4", "Traffic_Divert", "network_slicing"

0-day	"Anonymity" "Authentication", "AuthoriseAccess_resource", "DTLS_protocol", "Filtering_L3", "Filtering_L4", "Traffic_Divert", "IoT_honeynet", "IoT_control", "network_slicing"
Traffic analysis/Sniffing	"Anonymity" "DTLS_protocol", "Filtering_L3", "Filtering_L4", "Traffic_Divert"

3 CONCLUSIONS AND FUTURE WORK

In this deliverable, we have described the functioning of the Reaction Module, by reporting in detail how it is contextualized on the ANASTACIA framework and that other modules are directly involved in the communication with it. We have then described in detail each component of the module, its behaviour and how it is programmed to communicate with external components or modules.

A summary of the internal communications of the Reaction Module is reported in the following.

From	To	Data	Format	Services
Verdict and Decision Support System	Security Alert Service	Information about detected threats in form of Alerts.	JSON	Dedicated RabbitMQ queue with SSL
Mitigation Action Service	Security Alert Service	Information on policy selected to mitigate the detected threat	HTTP POST request with JSON data	Web server, currently only HTTP connection
Verdict and Decision Support System	Mitigation Action Service	Information about detected threads in form of alerts.	JSON	Dedicated RabbitMQ queue with SSL
Assets model	Verdict and Decision Support System	Mitigation strategies, Suitable threats per mitigation, Capabilities per mitigation, Mitigation actions and flavours and Action targets	JSON	REST API

Table 6 - Summary of internal communications of the Reaction module