Co-funded by the Horizon 2020
Framework Programme of the European Union

Practical Autonomous Cyberhealth for resilient SMEs & Microenterprises

Grant Agreement No. 883335
Innovation Action (IA)

Deliverable 3.2 – PALANTIR Secure Services Ecosystem - Second release

Document Identification			
Status	Final	**Due Date**	28/02/2023
Version	1.0	**Submission Date**	28/02/2023

Related WP	WP3	**Document Reference**	1.0
Related Deliverable(s)	D3.1, D4.1, D5.1, D4.3, D5.2	**Dissemination Level (*)**	PU
Lead Participant	UMU	**Lead Author**	UMU
Contributors	i2CAT, UBI, UMU, ORION, NCSRD	**Reviewers**	i2CAT
			UBI

Keywords:
Security Capabilities, SecaaS, Security Orchestrator, Security Capabilities Catalogue, Risk Analysis Framework, design, specification, implementation, evaluation, data models, interfaces, GDPR compliance

This document is issued within the frame and for the purpose of the *PALANTIR* project. This project has received funding from the European Union's Horizon2020 Framework Programme under Grant Agreement No. 883335. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

Document Information

List of Contributors	
Name	Partner
Antonio López Martínez, Manuel Gil Pérez	UMU
Carolina Fernández, César Cajas Parra	i2CAT
Labros Katsikas, Nikolaos Gkioules	UBITECH
Akis Kourtis, George Xylouris	ORION
Andreas Oikonomakis	NCSRD

Document History			
Version	Date	Change editors	Changes
0.1	10/01/2023	Manuel Gil Pérez	Initial Table of Contents and initial assignments
0.2	30/01/2023	Antonio López Martínez	Design, specifications, implementation, and annexes for the SCs
0.3	02/02/2023	Carolina Fernández	Design, specifications for the SO
0.4	10/02/2023	Carolina Fernández	Implementation, annexes for the SO
0.5	10/02/2023	Labros Katsikas, Nikolaos Gkioules	Design, specifications, and implementation of the SCC
0.6	13/02/2023	Akis Kourtis, George Xylouris, Andreas Oikonomakis	Design, specifications, and implementation of the RAF
0.7	14/02/2023	César Cajas	Annexes for the SO
0.8	17/02/2023	Carolina Fernández	Evaluation, introduction, conclusion
0.9	23/02/2023	Manuel Gil Pérez	Final review and formatting

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	UMU	24/02/2023
Quality manager	INFILI	27/02/2023
Project Coordinator	DBC	28/02/2023

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
AAC	Authentication and Authorisation Control (SO module)
AE	Attestation Engine (TAR subcomponent)
API	Application Programming Interface (SO module)
ATR	Attestation and Remediation (SO module)
BUp	Backup (also a type of SC)
CFG	Configuration (SO module)
CNF	Cloud-based Network Function
CPU	Central Processing Unit
CRI	Container Runtime Interface
Dx.y	Deliverable number y, belonging to WP number x
DPI	Deep Packet Inspection (also a type of SC)
FBM	Fault and Breach Management (PALANTIR component)
FW	Firewall and Router (type of SC)
GB	Gigabyte
GDPR	General Data Protection Regulation
IAM	Identity and Access Management
IDS	Intrusion Detection System (also a type of SC)
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
L-SL	Lightweight Shipper for Logs (type of SC)
LCM	Life-Cycle Management (SO module)
LSPL	Low Level Security Policy Language (used by SC)
MON	Monitoring (SO module)
MSPL	Medium Level Security Policy Language (processed by SC)
NDS	Network Data Sniffer (type of SC)
NFV	Network Function Virtualisation
NFVI	NFV Infrastructure
NFVO	NFV Orchestrator
NNS	Network NetFlow Sniffer (type of SC)
NS	Network Service
OCI	Open Container Initiative
OS	Operating System
OSM	Open Source MANO
PKI	Public Key Infrastructure
PKG	Package (SO module)
POL	Policies (SO module)
RAF	Risk Analysis Framework (PALANTIR component)
RAM	Random Access Memory

Abbreviation / acronym	Description
RE	Recommendation Engine (PALANTIR subcomponent)
REST	ReprEsentational State Transfer
SB	Script-Based (type of SC)
SC	Security Capability (PALANTIR component)
SCC	Security Capabilities Catalogue (SCO subcomponent)
SCC-O	Security Capabilities Onboarding (SCC subcomponent)
SCDV	Security Capabilities and descriptors validation (SCC subcomponent)
SCMA	Security Capabilities Metadata Access (SCC subcomponent)
SCO	Security Capabilities Orchestrator (PALANTIR component)
SCPM	Security Capabilities Package Maker (SCC subcomponent)
SCR	Security Capabilities Registration (SCC subcomponent)
SCS	Security Capabilities Search (SCC subcomponent)
SDN	Software-Defined Networking
SEM	Security Element Manager (module within a SC)
SIEM	Security Information and Event Management (also a type of SC)
SM	Service Matching (PALANTIR subcomponent)
SO	Security Orchestrator (SCO subcomponent)
TAR	Trust, Attestation & Recommendation (PALANTIR component)
TI	Threat Intelligence (PALANTIR component)
URI	Uniform Resource Identifier
VDU	Virtual Deployment Unit
VIM	Virtualised Infrastructure Manager
VL	Virtual Link
VM	Virtual Machine
VNF	Virtual Network Function
VPN	Virtual Private Network (also a type of SC)
vTAP	Virtual Terminal Access Point (also a type of SC)
WP	Work Package
WTA	Web-based Traffic Analysis (also a type of SC)
xNF	(Container and Virtual) Network Function
YAML	YAML Ain’t Markup Language

Executive Summary

The current deliverable relies on the design, specification and implementations considerations already documented in D3.1, where all WP3-related subcomponents from the PALANTIR architecture were detailed. Specifically: (i) a number of Security Capabilities (SC), or virtualised security services; (ii) the Security Capabilities Orchestrator (SCO), in charge of (a) managing the SC's life-cycle management via the Security Orchestrator (SO), and (b) keeping a searchable Security Capabilities Catalogue (SCC); as well as (iii) the definition of the Risk Analysis Framework (RAF).

A first introduction to the intent of the deliverable is given in section 1. After this, the next sections provide a summary of the latest status per subcomponent and get into the details of the specific changes that occurred since the previous iteration of this deliverable (D3.1), where the first release of the Secure Services Ecosystem took place. The motivation for this structure is to avoid repeating content, focusing on which changes took place since D3.1.

Specifically, section 2 summarises the design and architecture of each subcomponent, and then provides the delta of the changes since D3.1. Section 3 lists the requirements and describes their fulfilment and changes, considering the latest status. The implementation is summarised in section 4, and the identified changes since the first release are also explained.

After all the above, section 5 provides evaluation results, which are extracted from the operation of the components. This is provided either in unitary or integrated fashion, according to the subcomponent Conclusions are introduced next.

This deliverable concludes with the annexes, which complement the explanations above with models used by or governing the logic of the WP3 subcomponents, descriptors and compliance; as well as documenting the up-to-date interfaces, whether used internally to each component or exposed to other PALANTIR components.

1. Introduction

This deliverable, "D3.2 PALANTIR Secure Services Ecosystem – Second Release", compiles the work done in tasks T3.1 to T3.4 in the whole lifetime of WP3; focusing on the specific changes introduced since the publication of D3.1 in M15. This is the final iteration and relies on the previous one to focus on the new changes introduced.

The document provides an introduction to the design, specifications and implementation; as well as some evaluation of the work carried out. First, the architecture and more specific design is recalled for each of the tasks. After this, the list of specifications which were modified or added since D3.1 are enumerated in the tables, and the changes are summarised. The implementation (i.e. specific technologies and tools in use, as well as other development- or deployment- related decisions) details are adjusted to the latest changes incurred by each of the WP3 components or subcomponents. All these sections highlight the modifications since D3.1 documented the first iteration (and release) of the secure services ecosystem. After that, a subset of the WP3 components is assessed, where measurements were taken on different operations and environments expected within the project.

Besides this, D3.2 gathers other details regarding data modelling that guides the design and implementation of different components and subcomponents, as well as low-level details on the exposed interfaces to deliver and expose the logic for operation within the PALANTIR framework; and finally, the regulatory compliance of each element regarding the General Data Protection Regulation (GDPR).

The primary audience of this document are all technical consortium members, i.e., those involved in the implementation and technical decision taking, who participate either in WP3 and/or in the other directly or indirectly related WPs (such as WP4 and WP5). This report provides design, implementation and evaluation details that can be of use both to any technical internal reader, to complement and extend knowledge on the upcoming deliverables; as well as to any external reader that is involved in the virtualised instance orchestration and cybersecurity environment.

2. Design

The following subsections provide a summary of the design presented in D3.1. This has been re-analysed for compliance with the stated requirements, where some were identified at that point, and some were modified during the last period. This entire section is also complemented in Annex A with the data modelling needed for the formal definition through ontologies of the different elements of the PALANTIR platform, together with their relationships; as well as with the multi-tenancy functionality to achieve more flexibility in the management of the resources when several tenants are involved.

2.1. Security Capabilities

As presented in D3.1, the standard design for the SCs is defined in figure 2.1-1. This corresponds to the final design since the internal structure of the Network Service (NS) is maintained from the previous deliverable. Summarising the SC architecture: the NS encompasses the Security Element Manager (SEM) and the Container and Virtual Network Function (xNF), being the SEM the input/output point with respect to the xNF internally deployed.

The SEM exposes two external interfaces to interact with the xNF: (i) the *VeNf-Vnfm* and (ii) the Message Broker (here, Kafka [1]). The former allows the SO to apply day0, day1 and day2 actions, which reconfigure the xNF lifecycle. The latter implements the collection of internal data generated in such xNF. Besides, the SC has another interface with the Network Function Virtualisation Infrastructure (NFVI) component, which exposes some relevant information for the SC deployment and lifecycle.

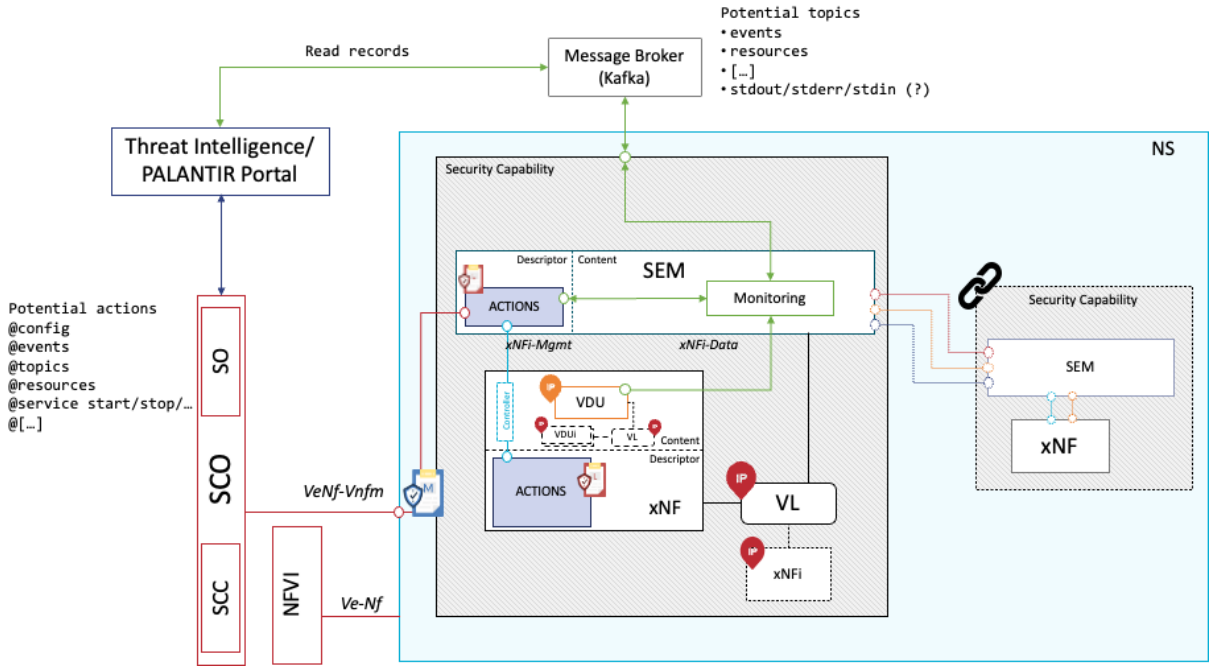

Figure 2.1-1: SC detailed architecture.

Internally, the xNF may include one or more Virtual Deployment Units (VDU), where the final security service is instantiated (i.e. a firewall), and the Virtual Link (VL) to connect the VDUs. To provide more granularity, service chaining is allowed; having only one internal channel for both management and data. Regarding the internal modules, interfaces and communication channels and guaranteed functionalities, the same design is maintained from D3.1, but some changes will be shown in the following section, especially in the types of SCs.

2.1.1. Changes from D3.1

The latest requirements in the PALANTIR project have originated some divergence since D3.1. To start with, the original types of SCs corresponding to Intrusion Detection System (IDS), Deep Packet Inspection (DPI), Network NetFlow Sniffer (NNS), Lightweight Shipper for Logs (L-SL), Firewall and Router (FW), Backup (BUp), and Web-based Traffic Analysis (WTA) have been evolved to cover more security services and provide affordable monitoring and remediation actions; as well as the incorporation of new types of SCs.

Virtual Terminal Access Point (vTAP) is added to the list of SC types in order to execute port (traffic) mirroring functionality to create a correct SC baseline deployment, as explained below.

Security Information and Event Management (SIEM) is incorporated as a new SC type to implement monitoring and remediation capabilities, from a host-based perspective; and in contrast to the network-based perspective presented in D3.1. This SC allows registering and managing agents installed in the SME hosts, monitoring data and triggering remediation actions. In addition, Virtual Private Network (VPN) is also developed as a new SC to provide suitable security to SME communications.

The L-SL SC has been redefined as SEM for all SCs because it offers the capability to collect the logs generated by the security service instantiated in the SC and forward them to the message broker. At this time, the SEM is composed of the L-SL functionality and the Juju controller, presented in D3.1. It triggers the day0, day1 and day2 actions through the SCO via the *VeNf-Vnfm* interface. The technologies selected to create the security services finally deployed in the SCs will be detailed in section 4.1. The IDS and DPI SCs have been unified in the same SC type, called IDS.

The original NNS SC did only encompass the NetFlow [2] data modality. With the new use case scenarios and threats to cover in the PALANTIR project, this SC has been renamed to Network Data Sniffer (NDS) to add other data modalities. At this time, a Zeek [3] (an open-source network traffic analyser) data modality has been incorporated into the project together with the NetFlow data modality. Therefore, the NDS SC includes the collection of Zeek and NetFlow data, depending on each client's needs and the contracted billing option. The FW SC and NDS SC are generally offered together to provide a monitoring and remediation SC.

The incorporation of a SIEM SC has also created a new type of SC, which is Script-Based SCs (SB SC). The potential of the SIEM SC, acting through the agents installed in the SME hosts, has allowed the provision of scripts to perform different host-based remediation actions; entirely implemented and offered by the SC developer. For instance, the SIEM SC allows the isolation or shutdown of a machine when a ransomware attack is detected by the Threat Intelligence (TI) component, thanks to the data collected by the agent running there. In this context, the BUp SC is transformed into an SCR SC allocated in the agent configuration, which is to be triggered on the client side. The possibilities of the SCR SCs are unlimited; as these can run in any Operating System (OS), provided there is an adaptation for the specific commands.

In accordance with specific requirements of the Fault and Breach Management (FBM) component, the health-check status has been implemented into each SC, exposing relevant information, such as the internal service status, the RAM usage, CPU usage, and the input/output network traffic load. FBM uses this mechanism to manage the SC status and inform the PALANTIR clients about the SCs that they have deployed in the infrastructure.

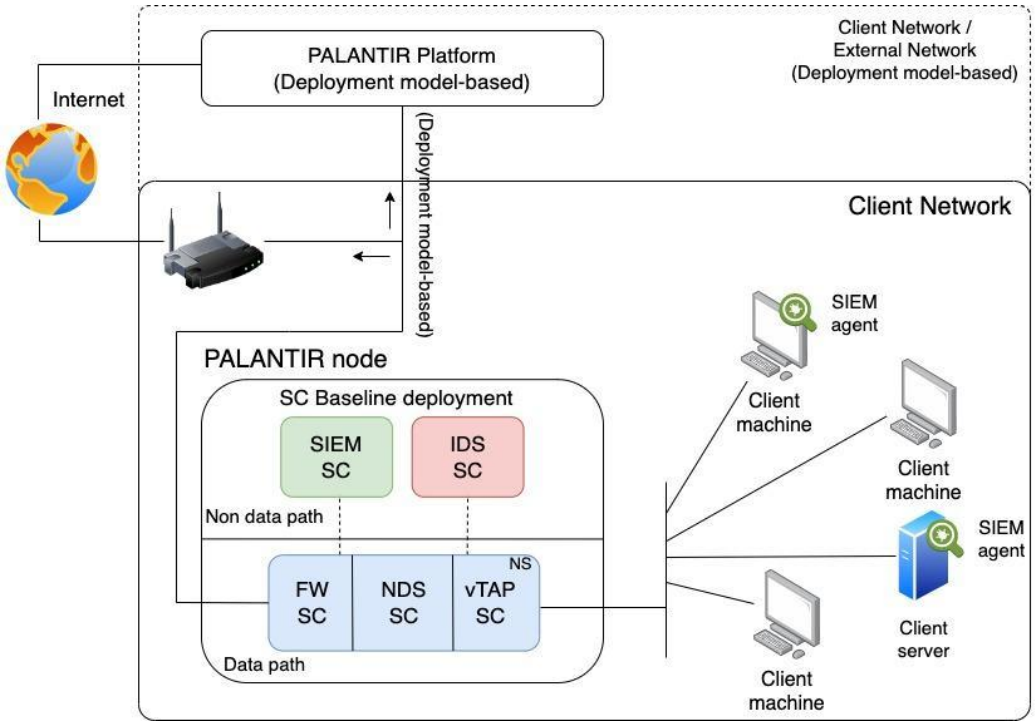

Figure 2.1-2: SC detailed architecture.

Finally, the PALANTIR project has defined the minimum set of SCs (i.e. the SC baseline) to deploy in each client. This SC baseline deployment is presented in figure 2.1-2 and comprises (i) an FW and NDS SC, (ii) an IDS SC, and (iii) a SIEM SC. Thus, the FW and NDS SC act as the client network's manager (allocated in the data path), the IDS SC monitors the network traffic, and the SIEM SC monitors the host logs (allocated in the non-data path); thanks to the SIEM agents installed in the client machines selected. To provide an affordable SC baseline, the vTAP SC is deployed to implement the port mirroring functionality, since the IDS SC should work with a copy of the actual network traffic. Regarding the three delivery modes implemented in the PALANTIR project, the SC baseline deployment should manage the data path in each network client independently of the delivery mode. The non-data path SCs could be allocated externally to the PALANTIR platform if the port mirroring functionality is correctly configured to reach the PALANTIR platform.

2.2. Security Orchestrator

The architecture of the Security Orchestrator (SO) was first introduced in D3.1 and underwent minor alterations. Yet, overall the same interactions and distribution of logic still apply.

This subcomponent contains the following modules: (i) Authentication and Authorisation Control (AAC); (ii) API; (iii) Attestation and Remediation (ATR); (iv) Configuration (CFG); (v) Life-Cycle Management (LCM); (vi) Monitoring (MON); (vii) Package (PKG); (viii) Policies (POL).

The SO interacts with both the PALANTIR components from WP3 (i.e. with the SCs and SCC) and from WP4, such as with the Service Matching (SM), TAR and Dashboard.

It also interacts with some third-party tools that are not considered part of the PALANTIR platform itself. This is the case of Open Source MANO (OSM), acting as the Network Function Virtualisation Orchestrator (NFVO); as well as sending notification events through the Kafka message broker and requesting low-level infrastructure information, directly requested to the Kubernetes cluster – which, in the end, act as the Virtualised Infrastructure Manager (VIM).

The NFVO is a key piece for the SC orchestration, as it allows onboarding or registering the SCs (or VNFs and NSs) and managing its life-cycle operations. The SO leverages such functionality and complements with logic not covered by the NFVO, such as keeping observability on the managed deployment and protected infrastructure, track specific monitoring information or nodes, determining the adequacy of deploying and configuring or reuse an instance to directly apply the configuration or bind actions and infrastructure to specific tenants, among others.

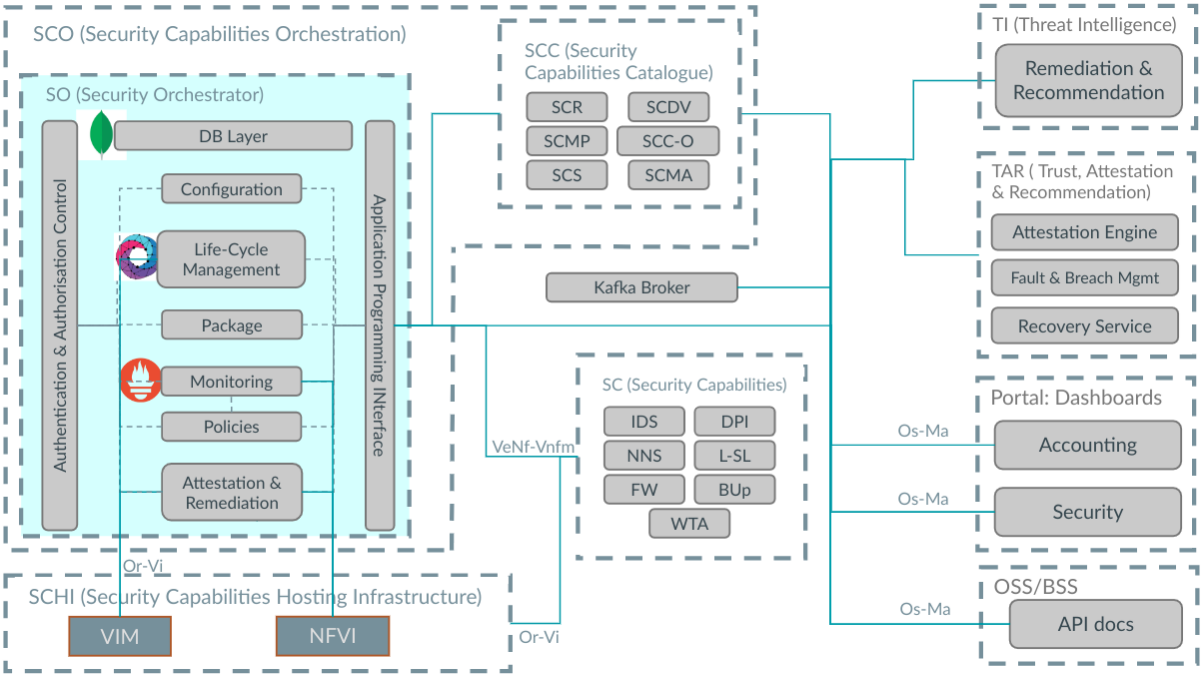

Figure 2.2-1: SO detailed architecture (left) with internal and external interactions.

Figure 2.2-1 covers the design of both the SO (left, centre side) and its relations with the infrastructure (left, bottom side), WP3 components (centre side) and WP4 components (right side).

2.2.1. Changes from D3.1

The most relevant change regarding the interactions between the SO and other components is that of the removal of the interaction with the Software-Defined Networking (SDN) controller in the NFVI, depicted in the design of the SO in D3.1. This is motivated by the discard of the SDN controller itself, and thus any requirement or logic that was derived from it is also withdrawn. Besides this, some other interactions were extended with more exchanged data, such as for the SM and TAR.

Internally to the SO, there were no changes in the architecture. Instead, reprioritisation of the visibility and importance of each module took place, as well as the rearrangement of the logic across such modules. In this regard, a part of the logic of the attestation-specific monitoring was centralised in the Monitoring module (MON) rather than in the Attestation & Remediation (ATR) itself, where such monitoring is queried by the LCM and notified to the Kafka topic consumed by the Attestation Engine (AE) in the Trust, Attestation & Recommendation (TAR) component.

At a more low-level fashion, the design of the SO (e.g. specific models, data exchange and workflows) was impacted by the introduction of the multi-tenancy approach; also required to cover the multiple delivery modes. Details on the changes that support the multi-tenancy are documented in Annex A.

2.3. Security Capabilities Catalogue

In D3.1, the SCC was introduced as a subcomponent of SCO, with interactions taking place between the SO, the dashboard (T4.1), Billing Dashboard and SM (T4.3) and TAR (T4.4). Of these interactions, the most significant ones involve the SO and the dashboard. In summary, the SCC offers several key functionalities, including the registration and validation of SCs, secure onboarding of service components to the SO, search functionality within the catalogue, and access to metadata for SCs by the dashboard and other PALANTIR components.

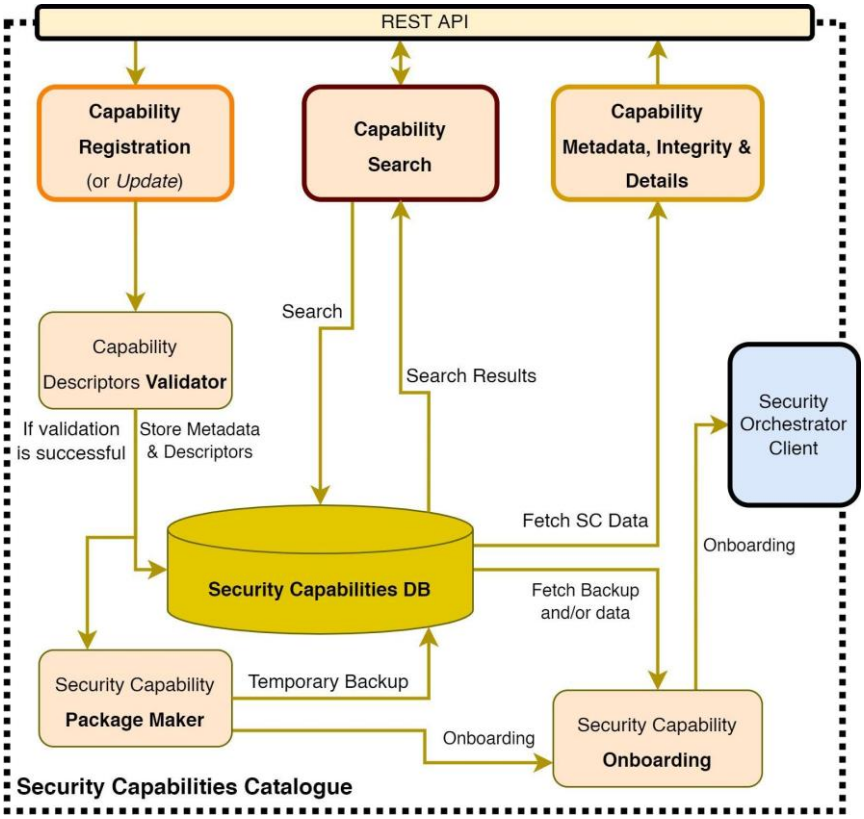

Figure 2.3-1: SCC architecture and interactions diagram.

Figure 2.3-1 depicts the internal SCC components, namely: (i) SC Registration (SCR), (ii) SC Search (SCS), (iii) SC Metadata Access (SCMA), (iv) SC and Descriptors Validator (SCDV), (v) SC Package Maker (SCPM), and (vi) SCC-O (SC Onboarding).

2.3.1. Changes from D3.1

Since D3.1, there have been some minor adjustments, primarily resulting from the integration with the dashboard and the introduction of multi-tenancy support. SCs can be designated as private if they are intended for internal use within the SC developer's organisation. When marked as private, the onboarding process is initiated automatically during registration; while, for public SCs, onboarding is initiated upon an authenticated dashboard user's subscription or purchase.

Furthermore, authenticated users have the ability to deploy or initiate the onboarding process for a SC through the SCC REST Application Programming Interface (API). To accommodate these changes, modifications to the authentication mechanism and SCC database schema have been implemented.

2.4. Risk Analysis Framework

The RAF component provides a risk calculation and analysis for SMEs based on business profiling, asset profiling and asset vulnerabilities assessment.

The external interface of RAF comes in the form of a user-friendly questionnaire where simple questions are filled from the SMEs. The questions are designed for collecting valuable and meaningful information even from non-experts that will allow the risk analysis workflow to take place, as proposed from the ENISA Risk Management process mentioned in D3.1.

RAF is designed to be a standalone framework which communicates with two external PALANTIR components: (i) the PALANTIR Portal and (ii) the Recommendation Engine (RE). In the image below the two interfaces are depicted, where the user has access to the questionnaire UI through the integrated iframe to the Portal. After the analysis is completed, the results are communicated to the RE.

The RE is largely based on the decision tree, supervised Machine Learning algorithm. A decision support tool known as a "decision tree" employs a tree-like model to represent options and their potential outcomes. It is a technique to present an algorithm that solely uses conditional control statements. RE creates and updates its tree-like model of potential outcomes, which is based on the answers collected from the questionnaire that the RAF component creates and sends to the RE. With the help of the mapping and different combinations of the questions-answers, RE makes decisions about which recommendation to execute or propose. As mentioned before, this requires some preliminary preparation and mapping of the existing questions with answers. The RE creates the recommendations and forwards them to the other component that requests the results.

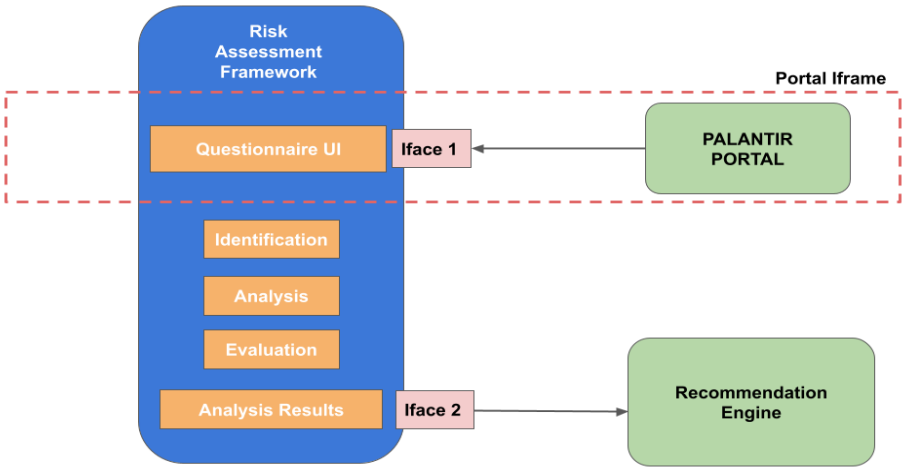

Figure 2.4-1: subcomponents of the RAF and interfaces with other PALANTIR functions.

In the PALANTIR platform, the user in the Portal activates the RAF questionnaire and answers it till the end. After submitting the questionnaire, RAF sends the results to the RE. RE determines the recommendations and based on each of them it asks other components (such as the SM) for the SCs, their costs and the billing results. RE returns the recommendations with SCs, costs and billing results to the Portal, which displays it to the user. This process is illustrated in figure 2.4-1.

2.4.1. Changes from D3.1

The design of the framework regarding the risk assessment and calculation has not changed. The main difference with the previous version is the definition and development of the interfaces used from the external components.

3. Specifications

This section reviews the requirements set out in D3.1 for compliance. It also proposes changes, if needed, with new technical specifications of the different components of the secure services ecosystem and adapted according to their initial version. It is also worth mentioning that the same nomenclature is used across all specifications-related tables, where the (+) indicates that a new module is added to satisfy a specification, and where the (-) indicates its removal.

3.1. Security Capabilities

The SCs encompass the security services implemented to deliver affordable Security-as-a-Service. Mainly, the SCs are divided into two families, monitoring and remediation capabilities, covering the detection and mitigation of identified threats. The final list of SCs targeted for implementation is an IDS, a FW, a NDS, a WTA, a SIEM, a VPN, a vTAP and an SB solution.

The following tables present the proposed requirements transcribed to technical specifications, providing the changes appeared from D3.1 regarding the SCs. For a proper interpretation, table 3.1-1 shows the new technical specifications appeared in the recent context of the PALANTIR project.

Table 3.1-1: new technical specifications for SecaaS.

Req. ID	Requirement description	IDS	SIEM	NDS	VPN	FW	vTAP	WTA	SCR
R1.3.10	The platform SHALL be able to deploy security capabilities from the Catalogue to operate with a copy of network data (off-the-path traffic)	✓	-	-	-	-	-	✓	-
SC_S19	The SCs shall work with a copy of traffic (off-the-path traffic)								
R1.3.11	The platform MAY be able to deploy security capabilities from the Catalogue to operate with online network data (on-the-path traffic).	-	✓	✓	✓	✓	✓	-	✓
SC_S20	The SCs shall work with online network data (on-the-path traffic).								

On the other hand, table 3.1-2 contains the requirements already presented in D3.1 but with some modifications. The requirements maintained in their original version shall be omitted.

Table 3.1-2: adapted technical specifications for SecaaS.

Req. ID	Requirement description	IDS	SIEM	NDS	VPN	FW	vTAP	WTA	SCR
R1.3.14	The platform SHALL be able to retrieve the basic status for the security capabilities instantiated or available (in the Catalogue).	✓	✓	✓	✓	✓	✓	✓	✓
SC_S9	The SCs shall give their basic status in form of health-check day2 action when needed.								

R1.3.19	The platform SHOULD deliver adaptive filtering and traffic control capabilities.	✓	✓	✓	✓	✓	✓	✓	✓
SC_S10	The SCs should deliver adaptive filtering and traffic control functionalities.								
R1.3.20	The platform SHOULD deliver port and service scanning capabilities.	✓	✓	-	-	-	-	-	✓
SC_S11	The SCs should deliver port and service scanning functionalities.								
R1.3.21	The platform SHOULD deliver remote attack detection capabilities.	✓	✓	-	-	✓	-	✓	✓
SC_S12	The SCs should deliver remote attack detection functionalities.								
R1.3.22	The platform MUST provide protection from data exfiltration attempts.	✓	✓	-	-	✓	-	✓	✓
SC_S13	The SCs must provide protection from data exfiltration attempts.								
R1.3.23	The platform SHOULD offer packet inspection capabilities.	✓	✓	-	-	-	-	-	✓
SC_S14	The SCs should provide DPI functionalities.								
R1.3.24	The platform SHOULD deliver intrusion detection and prevention capabilities.	✓	✓	-	-	✓	-	✓	✓
SC_S15	The SCs should deliver IDS functionalities.								
R1.3.25	The security capabilities MAY implement techniques such as exact data matching, structured data fingerprinting, statistical methods.	✓	✓	-	-	✓	-	✓	✓
SC_S16	The SCs may develop different mechanisms to provide data security and statistical methods.								
R1.3.29	The platform SHOULD prevent and react against Ransomware attacks.	✓	✓	-	-	✓	-	✓	✓
SC_S18	The SCs should deliver prevention and reaction ransomware functionalities.								

3.1.1. Changes from D3.1

Some specifications were added:

- SC_S19 and SC_S20 (related to R1.3.10 and R1.3.11, respectively): while initially these requirements were not contemplated in D2.1 and D3.1; the new traffic mirroring functionality, incorporated in the recent context of the PALANTIR project and contextualised in section 2.1.1, has required adopting such requirements for the SCs. Section 4.1 brings the low-level details of the port mirroring implementation.

Some other specifications were modified:

- The table columns have been updated by the changes produced in the list of the SC types implemented, where some SC types addressed in D3.1 have been reformulated in D3.2 and new types have also been incorporated.
- SC_S9 has been reformulated in order to indicate the name of the specific mechanism, i.e. the health-check day2 action.

3.2. Security Orchestrator

The SO comprises all modules related to the orchestration and life-cycle management of SCs, along with its monitoring. Given the central position it occupies in the architecture, it also provides ancillary functionality to other components and subcomponents within WP3 and WP4.

Table 3.2-1 describes how the requirements map to the technical specifications that were fulfilled in the SO subcomponent, as well as the internal modules that contribute to each specification. The next subsection explains in detail the changes on the specifications and/or requirements.

Table 3.2-1: adapted technical specifications for SO.

Req. ID	Requirement description	AAC	API	ATR	CFG	LCM	MON	PKG	POL
R1.3.2	The platform SHALL be able to configure security capabilities, whether already deployed or newly instantiated.	✓	✓	-	- (-)	✓	-	-	-
SO_S2	The SO shall be able to configure SC instances from the configuration action received within the PALANTIR platform. Furthermore, if the requested SC types are not running, previously instantiate them.								
R1.3.9	The platform SHOULD be able to monitor the deployed security capabilities and expose such data through programming interfaces for other internal components.	✓	✓	-	- (-)	-	✓	-	-
SO_S4	The SO should monitor the SC instances and expose the fetched metrics and telemetry data through programmable interfaces, accessible to other components.								
R1.3.10	The platform SHALL be able to deploy security capabilities from the Catalogue to operate with a copy of network data (off-the-path traffic).	✓	✓	-	- (-)	✓	-	✓	-
SO_S5	The SO shall instantiate SCs in a way they can process network data in an offline manner (e.g., through logs).								
R1.3.11	The platform MAY be able to deploy security capabilities from the Catalogue to operate with online network data (on-the-path traffic).	✓	✓	-	- (-)	✓	-	✓	-

SO_S6	The SO may instantiate SCs in a way they can process live network data (e.g., through mirrored interfaces).								
R1.3.14	The platform SHALL be able to retrieve the basic status for the security capabilities instantiated or available (in the Catalogue).	✓	✓	-	- (-)	✓	-	- (-)	-
SO_S7	The SO shall expose the list of both available SC packages and the SC running instances through programmable interfaces, accessible to other components. It shall also expose the particular status and useful details of the SC running instances.								
R1.3.15	The platform SHOULD be able to decide whether to reuse existing security capabilities or if new ones have to be instantiated, according to the received policy specifications.	-	-	-	✓	✓	✓	✓	-
SO_S8	The SO should identify appropriate running instances of SCs in order to use and configure according to policies, or otherwise instantiate new SCs.								
R1.4.1	PALANTIR SHOULD deploy mechanisms for the periodic attestation of the platform and the running applications', services' and configurations' integrity.	-	-	✓	-	✓	✓ (+)	-	-
SO_S10	The SO should provide the available specific runtime and environmental data, so it can be used during the attestation process.								

3.2.1. Changes from D3.1

A number of specifications have been reviewed or dropped. Regarding the reviewed ones, the changes are listed below:

- SO_S2 (related to R1.3.2) now indicates that the SO receives a configuration action, not the Medium Level Security Policy Language (MSPL) itself. Instead, the logic that would traditionally be obtained after translating the MSPL to Low Level Security Policy Language (LSPL) is covered inside of each SC and tailored to its specific behaviour.
- All specifications had the relation to the CFG module removed because its objective is to allow exposing and modifying general system-wide information rather than the specific information related to SCs or infrastructures. The exceptions to this review process are SO_S8 (related to R1.3.15); and SO_S10 (related to R1.4.1), which was already not related to the module in the prior iteration because the interaction with the attestation was already expected to differ slightly with respect to other components.

On the other hand, other specifications were removed:

- SO_S3 (related to R1.3.7) since, once all SCs are deployed via Docker runtime (with Open Container Initiative or OCI images), the Container Runtime Interface or CRI takes care of properly handling all the dependencies (the image). This is not the case for QEMU images used by OpenStack, which must be previously uploaded to the VIM and are not fetched on demand by the CRI. Therefore, this requirement (and its specification) is no longer relevant.
- SO_S9 (related to R1.3.16) with the usage of SDN technology to programmatically configure the networks that interconnect the infrastructures. The requirement was discarded, considering the needs of the infrastructure and the setup of the different delivery modes.

Finally, a potential requirement to be dropped is R1.3.16 (related to SO_S9), where both the SO and the SCHI were expected to fulfil it.

3.3. Security Capabilities Catalogue

The SCC is composed of modules related to the storage, onboarding, searching, and accessing data about SCs. The SCC provides functionality to other components and subcomponents within other WPs, and contributes to the actualisation of functionality related to subcomponents of the SCO.

Table 3.3-1 describes how each requirement maps to the technical specification that is to be fulfilled in the SCC subcomponent, as well as the internal modules that contribute to the specification.

Table 3.3-1: adapted technical specifications for SCC.

Req. ID	Requirement description	SCR	SCDV	SCPM	SCC-O	SCS	SCMA
R1.3.1	The platform SHALL be able to instantiate security capabilities.	✓	✓	- (-)	✓	-	-
SCC_S1	In order for the SO to instantiate SCs, the SC shall first need to be registered and onboarded to the SO, which is done through the SCC.						
R1.3.2	The platform SHALL be able to configure security capabilities, whether already deployed or newly instantiated.	-	-	-	-	✓	✓
SCC_S2	The platform shall provide the necessary metadata view and search for SCs, in order for the SO to provide the necessary configuration options.						
R1.2.6	Security mechanisms used in a complex cybersecurity eco-system SHALL be able to identify, distribute and allocate responsibilities between 5G ecosystem stakeholders.	✓	-	-	-	✓	✓
SCC_S3	The SCC shall be able to store relevant mitigation capabilities of SCs, as well as search for SCs matching them, and show them for a requested SC, disregarding their origin.						
R1.3.3	The platform SHALL provide a variety of SecaaS packages on the Catalogue.	✓	✓	✓	✓	✓	✓
SCC_S4	The SCC shall provide the ability to register and onboard SecaaS packages as SCs, search for appropriate SCs based on metadata, and view their properties.						
R1.3.7	The security capabilities SHALL be uploaded to the catalogue as a pre-packaged bundle containing its basic dependencies. Any external dependency SHALL be provided before its uploading to the Catalogue.	✓	✓	✓	✓	-	-
SCC_S5	The SCC shall provide a registration and onboarding endpoint, which is used by an SC developer through the Portal. There, the developer will be able to add all metadata for the SC, while the software and dependencies to be uploaded are all pre-packaged as an image for VMs or containers.						

R1.3.10	The platform SHALL be able to deploy security capabilities from the Catalogue to operate with a copy of network data (off-the-path traffic).	-	-	-	✓	-	✓
SCC_S6	The SCC shall provide the metadata and descriptors for SCs that may operate on off-the-path-traffic, and shall be able to onboard those SCs on the SO.						
R1.3.11	The platform MAY be able to deploy security capabilities from the Catalogue to operate with online network data (on-the-path traffic).	-	-	-	✓	-	✓
SCC_S7	The SCC shall provide the metadata and descriptors for SCs that may operate on on-the-path-traffic, and shall be able to onboard those SCs on the SO.						
R1.3.14	The platform SHALL be able to retrieve the basic status for the security capabilities instantiated or available (in the Catalogue).	-	-	-	✓ (+)	✓	✓
SCC_S8	The SCC shall provide all metadata and descriptors for any available SC, as well as search functionality based on some of these parameters.						
R1.5.4	The platform SHALL be able to analyse an attack report to produce an ordered set of suggested actions (e.g. VNFs configuration) for mitigation.	-	-	-	-	✓	✓
SCC_S9	The SCC shall provide the necessary security enhancing metadata for the SCs, as well as the necessary search capability based on these parameters, so that TI or any other component or user is able to find the appropriate SCs in order to mitigate attacks.						
R1.4.1	PALANTIR SHOULD deploy mechanisms for the periodic attestation of the platform and the running applications', services' and configurations' integrity.	-	-	✓ (+)	-	✓	✓
SCC_S10	The SCC should provide the necessary integrity metadata for SCs, as part of the search and metadata access functionalities; so that the AE can perform periodic attestations on the deployed SCs.						
R1.2.7	The PALANTIR eco-system SHALL be able to publish security KPI that measure compliance with stakeholder Security Level Commitments.	-	-	-	-	-	✓
SCC_S11	The SCC shall provide the security metadata of an SC, as part of the metadata access, in order to be able to measure the compliance KPIs.						

3.3.1. Changes from D3.1

The following list contains a set of minor changes that do not affect the scope of the requirements.

- SCC_S1 (related to R1.3.1): the invocation of the SCPM module is optional since the SC developer could choose either to directly upload the required files needed for the deployment; or specify the exact image to be retrieved from another registry.
- SCC_S8 (related to R1.3.14): the invocation of the SCC-O module is mandatory since it is the main responsible module, to communicate with the SO in order to retrieve the status of the deployed SC.
- SCC_S10 (related to R1.4.1): the SCDV module should be considered as mandatory in order to validate the integrity of the provided metadata.

3.4. Risk Analysis Framework

The RAF is developed as a standalone, lightweight application. The refined architecture design and implementation allows an easy deployment and integration with respect to the specifications and requirements introduced in D3.1.

3.4.1. Changes from D3.1

There are no perceived changes in the requirements for RAF with respect to those introduced in D3.1.

4. Implementation

This section summarises the more technical details based on the changes identified in the previous sections, considering the technologies and techniques applied in each component. Similarly, this section is extended by Annex B with the intra- and inter- component APIs used by each component and subcomponent.

4.1. Security Capabilities

This section presents the implementation details of the SCs identified in the PALANTIR context, as well as the main changes since D3.1. As commented in section 2.1.1, the IDS SC (combining IDS and DPI SCs), the FW SC, the NDS SC (introduced as NNS SC in D3.1), and the WTA SC are types of SCs still being delivered. However, four new types of SCs have been developed: SIEM and VPN, SCR and vTAP SCs, whose specific development and deployment details are explained below.

4.1.1. Changes from D3.1

The SCs have finally been developed as Docker images since the standard VIM selected for the project is *Kubernetes* (K8s) [4]. K8s supports the different needs of the delivery modes, adapting it to the specific requirements in each environment. The base SC image incorporates an Ubuntu 20.04 version (to be aligned with the particular software requirements) with the security service software installed there. Besides, the SC image contains the necessary configurations for its correct deployment. All SC images are uploaded to Dockerhub [5], the official public repository where many developers contribute their images.

Juju is the technology used to encapsulate the SC image in a Juju charm (through a Python library). The juju charm comprises two pods (i.e. the logical wrapper entity for a container): one containing the SC image (deployed with the *Podspec* mechanism) and the other containing the Juju operator. The latter pod is used to trigger the day0, day1 and day2 actions into the SC image, used to reconfigure the SC in real-time. The Juju charm is packaged using the *charmcraft* command, which creates a .charm file containing its programming logic. To allow more complex deployments, the Juju charm can also be composed by a Helm chart and a Juju operator. The Helm chart is a K8s-related technology that offers a way to package a collection of K8s resources. Adopting this technology improves the adaptability of SC developments and the possible implementation of any security service. In practice, this type of SC is implemented differently because the Helm chart needs to be defined and, in this case, the Juju charm is implemented as a proxy to communicate with the Helm chart deployment. However, this is transparent to the final user since it continues to use the day0, day1 and day2 actions mechanism to interact with the SC. The Helm chart technology has allowed the incorporation of the SIEM SC since it encompasses different Docker images and complex interactions between them.

Regarding the SC types, there are some differences since D3.1. On the one hand, the services used for NDS SC have been established as follows:

- NetFlow collectors:
  - *fprobe* [6] is a libpcap-based tool that collects network traffic data and transmits it as NetFlow flows. fprobe exports NetFlows in version 1, 5 and 7, and needs the specification of the specific collector where the NetFlows will be sent.

    Requirements: 64-bit amd64 (x86-64) compatible CPU, 1GB RAM, 8 GB or larger disk drive.
  - *softflowd* [7] is a software implementation of a flow-based network traffic monitor. In contrast with fprobe, softflowd exports the NetFlows in version 1, 5 and 9.

Requirements: 64-bit amd64 (x86-64) compatible CPU, 1GB RAM, 8 GB or larger disk drive.

- *Zeek* is an open-source network security monitoring tool. It acts as a sensor that observes the network traffic. Zeek creates compact, high-fidelity transaction logs, file content, and fully customised output to be used by the TI components.

  Requirements: 64-bit amd64 (x86-64) compatible CPU, 1GB RAM, 8 GB or larger disk drive.

On the other hand, the SIEM SC, VPN SC and vTAP SC have been incorporated as new SC types. The VPN SC is a new SC type created to comply with requirements that gained prevalence with the increase of the remote working modality. Here, the following software is used:

- *Wireguard* [8] is a fast VPN that utilises cryptography (Noise protocol framework, Curve25519, etc) to protect the communications being fast, simple and clean.

  Requirements: 64-bit amd64 (x86-64) compatible CPU, 1GB RAM, 8 GB or larger disk drive.

Concerning the SIEM SC, *Wazuh* [9] is the main software deployed. This was already introduced in D3.1, but previously placed inside the IDS SC and now promoted as a separate one, given the monitoring capabilities offered by Wazuh. However, the research and use of this software has revealed that Wazuh may offer monitoring and remediation capabilities at host level, besides its own definition as a SIEM service. The SIEM SC allows the creation of a new modality of SCs: the SCR SCs. Essentially, such kinds of SCs operate system-related scripts that can be implemented and injected in the Wazuh agents, installed in the client's machines. Therefore, the scripts are developed as .sh and .bat files for Linux and Windows operating systems, respectively. In this context, the scripts are executed through the API exposed by the Wazuh Manager, which is the manager component of the Wazuh SC. This component receives the script name, the agent_id where the script shall be executed, and the parameters needed for launching the script. In practice, the specific day2 action created in the SIEM SC does trigger the SCR SCs.

Finally, the vTAP SC is implemented. This SC is designated to provide a suitable port mirroring functionality for the IDS SC and WTA SC in order to receive a copy of the real traffic. The *nTap* software is allocated here, which includes the following characteristics:

- nTap [10] is comprised of two binary applications, *ntap_remote* and *ntap_collector*. The latter is started to listen to a port and an auxiliary interface created for this purpose, and the former sends the original packets to the destination indicated.

  Requirements: 64-bit amd64 (x86-64) compatible CPU, 1GB RAM, 8 GB or larger disk drive.

4.2. Security Orchestrator

The implementation of the SO continued from the stage in D3.1, both in terms of adopted technologies and regarding specific implementation choices per module. There were a number of changes from the landscape depicted in D3.1, which are described in the following section.

4.2.1. Changes from D3.1

On the one hand, the adopted technologies for the development and deployment of SO were revised and some of them incurred into changes.

In this regard, Python3 is still in use to develop all the logic in the modules.

Flask [11] and FastAPI [12] are both used: Flask is used for exposing the interfaces of the internal modules, whilst FastAPI (with uvicorn [13]) is used to expose and document all the interfaces supported by the API module, which is exposed to the PALANTIR clients. Compared to FastAPI, Flask provides non-production servers, which may be acceptable as long as these are internal; and also lacks the user-

friendly interface to run the different actions that OpenAPI/Swagger [14] provides. Therefore, FastAPI is favoured for the external API module. Also, more technical documentation is provided (for the PALANTIR operator) in Markdown [15] files.

Regarding configuration, whilst JSON and YAML were initially explored, YAML is widely favoured and is the approach finally followed by SO to host the static configuration files. As per accepted mimetypes, the default accepted content type is JSON, whereas explicit YAML requests shall also be served.

As per the options to deploy SO itself, D3.1 initially proposed venv [16], Docker [17] (via docker-compose [18]) and Kubernetes [19]. In the end, the deployment provides scripts that rely on Docker (through docker-compose). The original venv-based deployments are available, but not further extended or adapted. Regarding Kubernetes, the prospecting efforts were not continued within the project and are enhancements left to the continued development of the SO beyond the project.

On the other hand, and regarding the integration of SO with the infrastructure (SCHI) and its deployment of the SC instances, OpenStack [20] was dropped since D3.1 in favour of Kubernetes as the default VIM for the Network Function Virtualisation (NFV) architecture. This fact impacts the SO in multiple workflows, specifically those related to (i) the onboarding of packages and ancillary information, which is now simplified by the usage of Docker images as these are automatically managed by the CRI; (ii) the different means to extract runtime information related to the properties of the running SC instance and the image containing the logic, which are needed for the attestation process; (iii) similarly, the extraction of infrastructure-related information; and (iv) the monitoring of metrics related to the running SC instances, which initially related to the monitoring of Virtual Machines (VMs) deployed as Virtual Network Functions (VNFs) and now relate to containers, deployed as Cloud-based Network Functions (CNFs) or, simply put, xNFs.

Finally, some modules within SO have suffered some changes regarding its development. Those which were subject to some changes are enumerated and summarised below:

- The AAC module integrates with a Keycloak instance, which is the Identity and Access Management (IAM) provider and brings its own abstractions for realms, clients and/or users and implementation of specific authentication means to help deliver the multi-tenancy.
- The API module only considers FastAPI to expose the externally facing interfaces, as indicated above.
- The ATR module drops the OpenStack client because this is no longer used as VIM. On the other hand, whilst it can use the Kubernetes client instead, part of this logic (that takes specific runtime information) occurs inside the MON module; so, ATR is mostly a proxy to access such information.
- The CFG module has its main aim changed not to expose and allow dynamic reconfiguration of the static files of each other module, but to act instead as an entrypoint to the MongoDB [21] database, to both store and retrieve most of such static information and also to manage new information related to the multi-tenancy. Therefore, it uses both the mongoengine and pymongo clients.
- The LCM module drops the usage of clients against OpenStack and Docker, since the former is no longer used and the latter can be replaced by the Kubernetes client, already providing the expected degree of granularity – and which, on the other hand, should not specifically constrained as the CRI.

4.3. Security Capabilities Catalogue

The implementation has undergone some changes since D3.1. Some new functionalities have been added to the SCC. Also, the authentication mechanism has been changed. The rest of the technology stack is the same as D3.1. The new implementation is depicted in figure 4.3-1, and more details on the implementation changes are described in section 4.3.1.

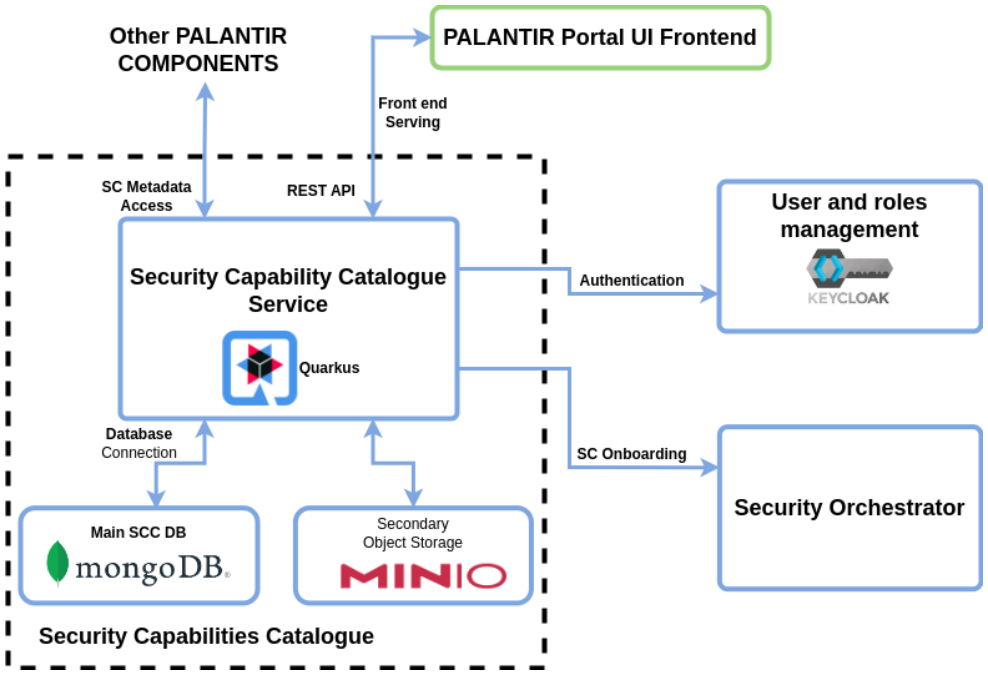

Figure 4.3-1: SCC technologies layout.

4.3.1. Changes from D3.1

The authentication between SCC and the dashboard is now established via Keycloak [22]. Furthermore, SCs can be marked as private, where a private SC is meant to be used only inside the SC developer organisation. Therefore, SCC has some new REST endpoints for the dashboard users to consume. A new endpoint to trigger the onboarding process has been added, along with an endpoint to trigger the deployment of a SC. Also, search functionalities of the SCC API are now restricted based on the user organisation who consumes the API. Regarding the SCC's database (MongoDB), changes were made to match the multi-tenancy requirements.

During the registration of a new SC, the related module of the SCC (in this case, SCC-R), is responsible for storing the SC developer's details; where a tenant or organisation ID is now stored in the DB, as this is mandatory for proper integration between the SCC and the SM and Billing Dashboard framework of WP4.

4.4. Risk Analysis Framework

The current version of RAF implements many functionalities that were under development when described in D.3.1. Many of the entities described were unified (e.g. control cards as part of the Risk Calculation & Assessment Engine). Also, the communication between the PALANTIR Portal and RE was defined and implemented.

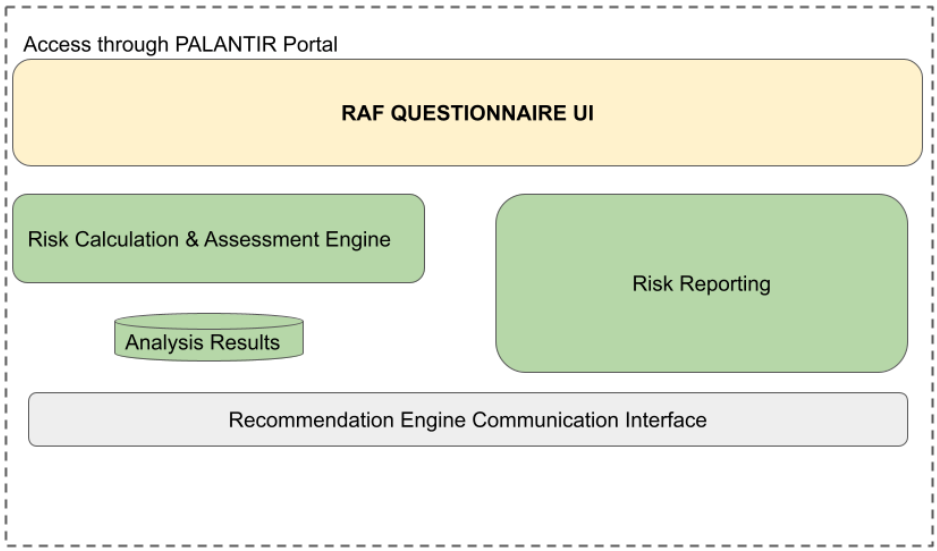

Figure 4.4-1: internal modules within the RAF.

The aim was to create a standalone framework, independent from other PALANTIR components where SMEs will have access and perform their risk analysis. The internal modules are depicted in figure 4.4-1. The framework backend is built using the Flask python which gives flexibility regarding the communication between the Portal and the Recommendation engine through its REST API. The database technology used for assets and the risk analysis results is MongoDB which integrates well with this Python-based framework. Finally, the front-end views of the RAF questionnaire are presented, which is implemented using SurveyJS libraries [23].

The RAF questionnaire UI, shown in figure 4.4-2, consists of three tabs.

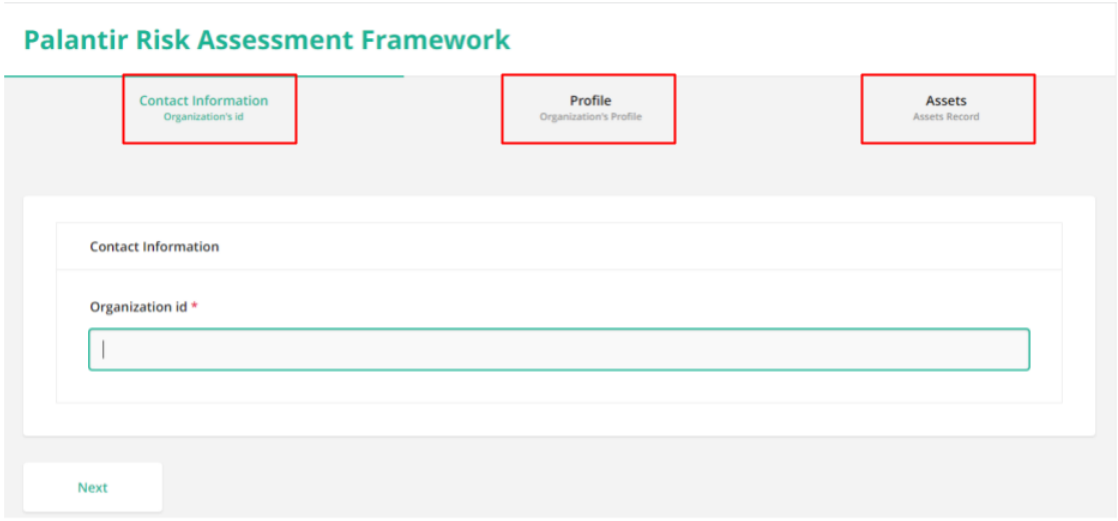

Figure 4.4-2: RAF questionnaire UI.

The contact information is filled in automatically from the PALANTIR Portal. The two remaining tabs (profile, assets) are actually the two phases mentioned in D.3.1:

- Phase1: questionnaire for risk profile.
- Phase2: asset identification.

In the Profile tab (see figure 4.4-3) the user must answer selected questions in order to determine the profile of the organisation by using 4 categories of risk profiles, as proposed from ENISA [24]:

- Reputation.
- Productivity.
- Financial stability.
- Legal and regulatory.

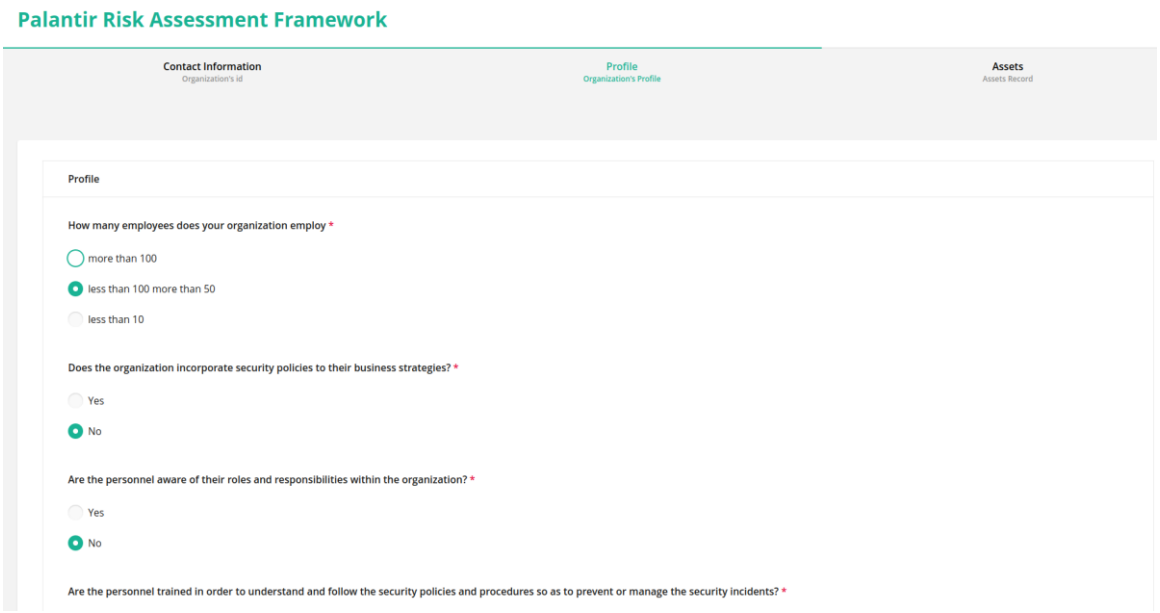

Figure 4.4-3: profile tab view within the RAF questionnaire UI.

After this phase is completed, a risk profile is calculated internally by scoring each risk profile.

By accessing the Assets tab (illustrated by figure 4.4-4), the user can list the organisation's assets and answer dedicated questions on each asset individually. The current version of RAF allows the user to add an arbitrary number of assets in a user-friendly way.

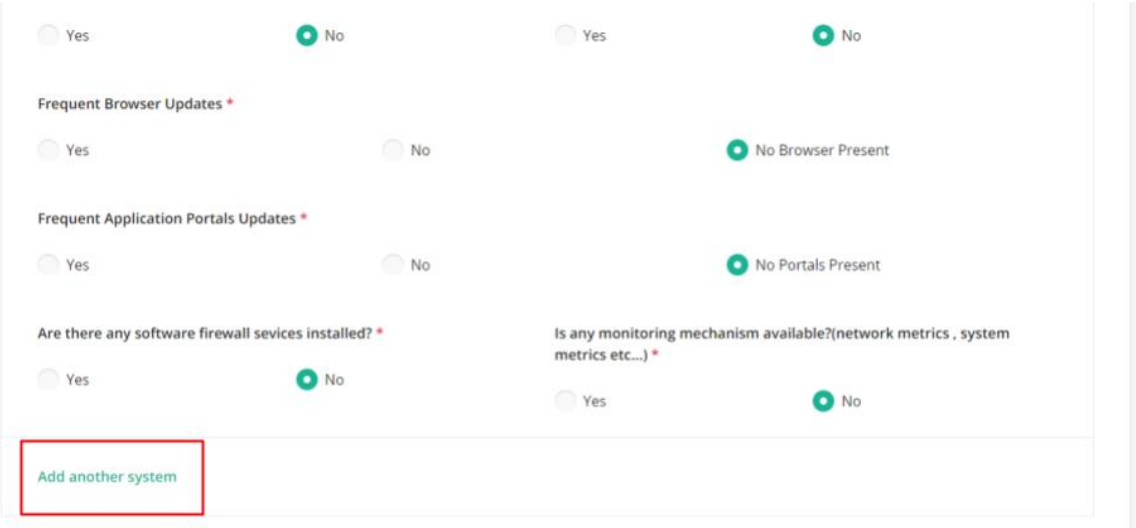

Figure 4.4-4: asset tab view within the RAF questionnaire UI.

Finally, when the user hits the submit button, the results of the analysis are sent to the Recommendation Engine. A sample output is provided below in JSON format.


```
{
    "results": [
        {
            "_id": "1",
            "results": {
                ##########################
                ### ASSET SCORES##########
                ##########################
                "assets_results": {
                    "Asset_0": {
                        "is_critical": true,
                        "potential_attacks": {
                            "Malware Attack": {
                                "estimated_vulnerability": "High"
                            },
                            "Password/Brute force Attack": {
                                "estimated_vulnerability": "High"
                            }
                        },
                        "specs_info": {
                            "os": "Windows",
                            "type": "Server",
                            "vendor": "DELL"
                        }
                    }
                },
                ##########################
                ### RISK PROFILE SCORES#####
                ##########################
                "org_id": "1",
                "profile_results": {
                    "financial_stability": "Medium",
                    "legal_and_regulatory": "High",
                    "productivity": "High",
                    "reputation": "High"
                }
            }
        }
    ]
}
```


4.4.1. Changes from D3.1

The previous version of RAF was a LimeSurvey-based [25] application.

In this version, the framework was rebuilt with custom Python code and well-known Python modules. As mentioned, the usage of the Flask module gives a lot of flexibility regarding the communication with other components. Since LimeSurvey had its own way to store information, a MongoDB database was now required to allow persisting data; and suitable connectors had to be developed for it.

Finally, the current version of RAF is a very lightweight application which can be either deployed as a Docker container or as a Kubernetes pod.

5. Evaluation

Following the design and implementation of the elements involved in WP3, measurements were taken to assess their behaviour, modelling it through repeated iteration and via stress-testing and extracting virtual resource consumption, on the one hand, and latency to perform operations, on the other.

5.1. Security Capabilities and Security Orchestrator

Given the tight relation between the SCs and the SO, carrying out its lifecycle management, a number of measurements were obtained from the operation of the SCs by the SO.

The environment used for this evaluation consists of a Kubernetes cluster with one control node and a number of workers, where the latter were used for deployment. Given that the motivation was the definition of a baseline benchmarking to consider per scenario, two types of workers were considered during the tests: (i) first, ones with full resources; (ii) then, others with a constrained subset of resources. This different setup relates to the different deployment scenarios or delivery modes: specifically, the scenario with full resources relates to the Edge and Cloud delivery models, given its larger number of resources; whereas the one with constrained resources resembles the Lightweight one, which is self-contained and limited.

The decision on how many resources to use for the full and constrained workers was motivated by the current availability of resources in the testbed, in the first case; and, in the second, by the economic adequacy of widespread computing equipment (in the range of 2k EUR) that can accommodate the deployment server for the SCs in the Lightweight mode.

Table 5.1-1 describes the specific resources assigned in each emulated scenario to the VMs hosting the Kubernetes workers.

Table 5.1-1: virtual resources per environment emulated (workers with full and limited resources).

	Sockets	**Cores**	**RAM (GB)**
Workers with full resources	3	4	20
Workers with constrained resources	1	4	8

The measurements considered the lifecycle managed by the SO; specifically, regarding the operations for instantiation, reinstantiation and configuration of SCs. These are illustrated in the boxplot charts from figures 5.1-1 to 5.1-3, respectively. A number of Python scripts were prepared so as to repeat each operation 100 times. Three of the available SCs were considered, from left to right in the figures: (a) iptnetflow, (b) snort and (c) suricata. It is worth noting that the environment was kept clean after every run, i.e. removing the deployed SC, which increased the testing time but ensured idempotent evaluation.

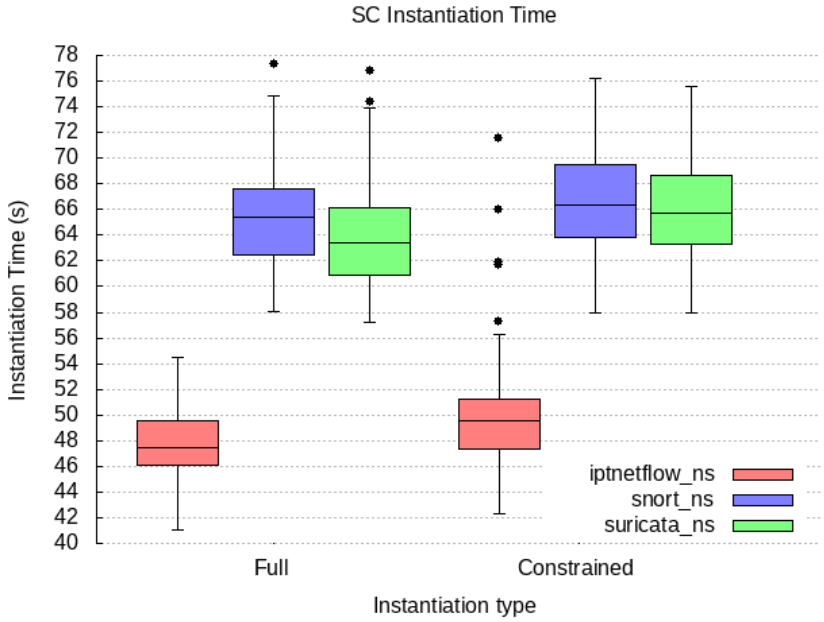

Figure 5.1-1: distribution of instantiation-related times for three SCs, considering full (left) and constrained (right) environments.

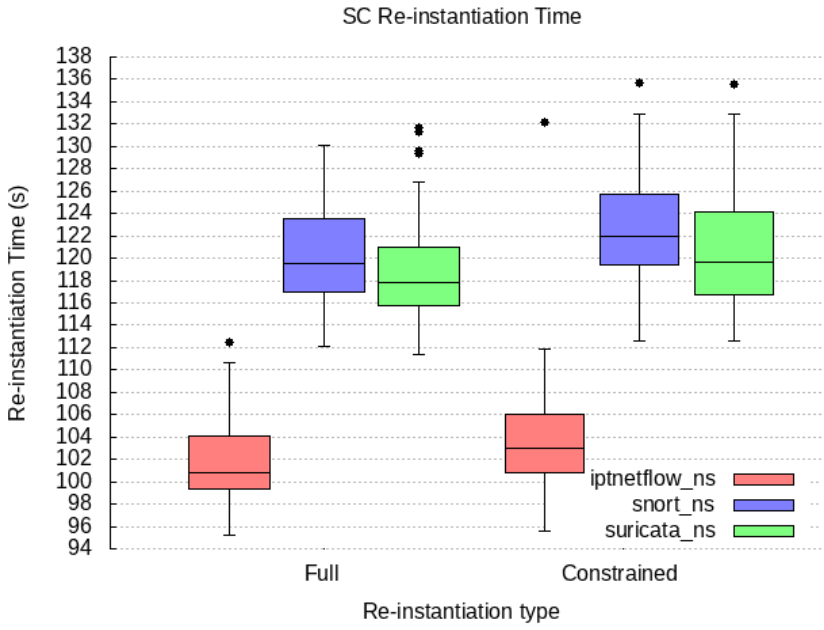

Figure 5.1-2: distribution of re-instantiation-related times for three SCs, considering full (left) and constrained (right) environments.

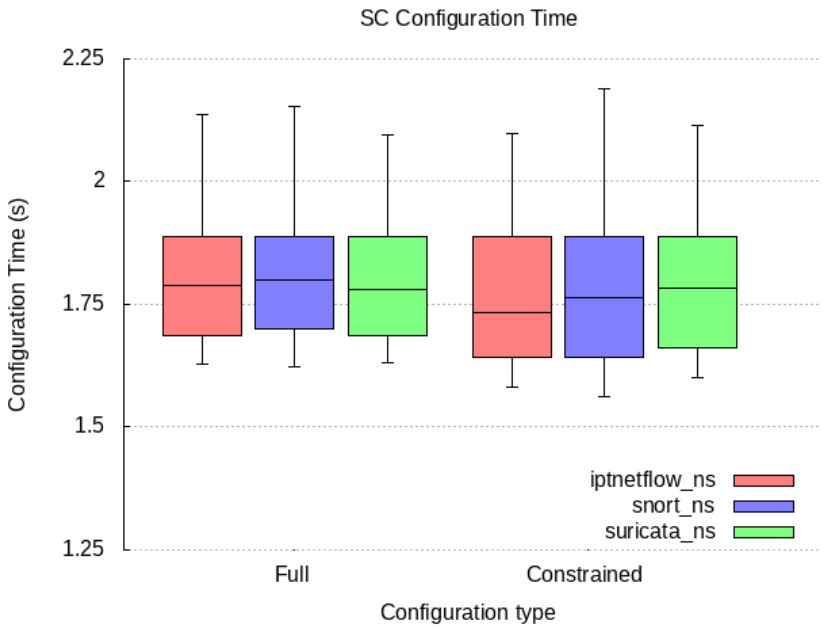

Figure 5.1-3: distribution of configuration-related times for three SCs, considering full (left) and constrained (right) environments.

The three different scripts generated logs with time measurements per operation, which were post-processed to extract CSV files that were input in the Gnuplot [26] for the generation of the plots defined above. The sources for the scripts and the data were used in the paper "PALANTIR: An NFV-Based Security-as-a-Service Approach for Automating Threat Mitigation" [27] and left into an open repository under the PALANTIR GitHub organisation [28].

The measurements indicate that the lifecycle management operations of the SCs do not suffer much when operating in a more constrained environment, not high-end or production-grade. This is more evident in the instantiation and reinstantiation times, where the mean time taken by each operation in a constrained environment increases in 4.2% and 2.2%, respectively.

5.2.Security Capabilities Catalogue

For the evaluation of the SCC module, a number of specific scenarios was considered to set the baseline regarding the performance metrics of this module. The execution of the scenarios took place in a dedicated environment with the resources described in table 5.2-1.

Table 5.2-1: available resources in the SCC testing environment.

VM Type	CPU Cores	RAM (GB)
Ubuntu 20.04.4 LTS	4	16

The testbed is based on Apache JMeter [29], an open-source tool for load testing and measuring performance. The evaluation process consists mainly of 2 scenarios of different workload. Table 5.2-2 gathers the details of each scenario.

Table 5.2-2: SCC performance scenario setup.

Name	Concurrent Users	Iterations	Ramp-up period
Scenario 1	25	10	0
Scenario 2	35	10	0

Both scenarios are based on the same API calls, executed in the same order. Table 5.2-3 shows the API invocations performed per user and iteration.

Table 5.2-3: endpoints invoked from the SCC APIs.

Name	HTTP Method	Path
get all deployments	GET	/api/v1/deploy
get all registrations	GET	/api/v1/register
get registration	GET	/api/v1/register/{id}
post registration	POST	/api/v1/register
search	POST	/api/v1/search
get deployment	GET	/api/v1/deploy/{id}

The metrics extraction from the application side was performed using a Prometheus server. The graphs shown in the following section are created with Grafana [30].

From the perspective of the resources consumed by the SCC application, figures 5.2-1 and 5.2-2 show the CPU and memory allocation, respectively for each scenario, during the performance tests.

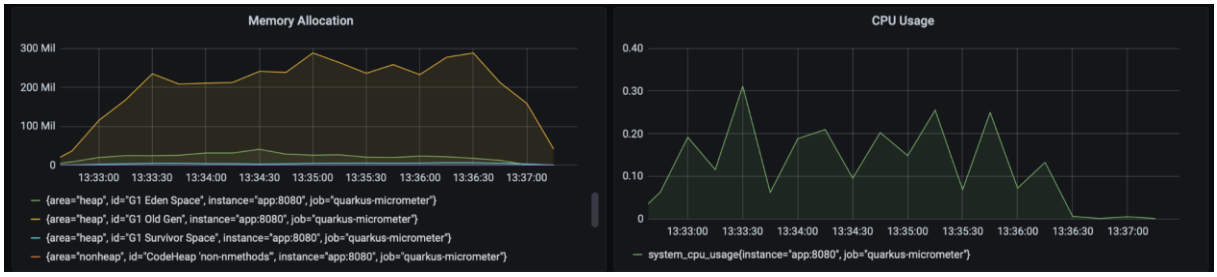

Figure 5.2-1: SCC CPU and memory allocation stats for scenario 1.

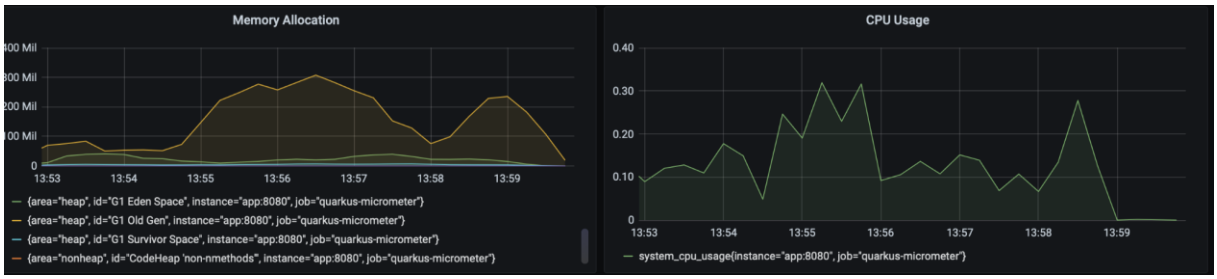

Figure 5.2-2: SCC CPU and memory allocation stats for scenario 2.

Figures 5.2-3 and 5.2-4 show the latency for the search-related operations for each of the two considered scenarios.

Figure 5.2-3: SCC search latency for scenario 1.

Figure 5.2-4: SCC search latency for scenario 2.

Figures 5.2-5 and 5.2-6 show the latency for the registration-related operations for each of the two considered scenarios.

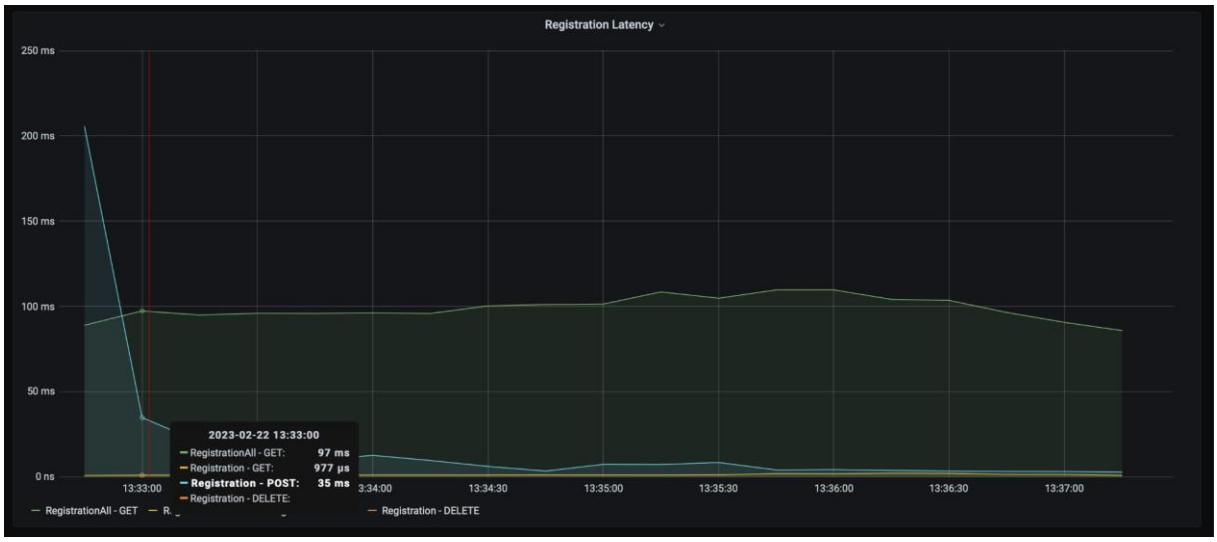

Figure 5.2-5: SCC registration latency for scenario 1.

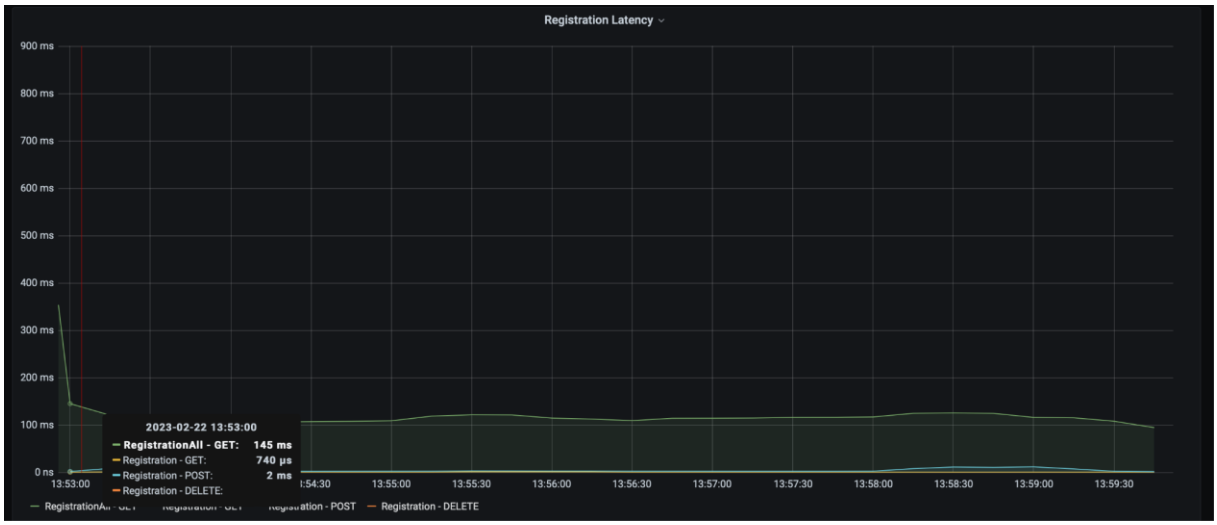

Figure 5.2-6: SCC Registration latency for scenario 2.

Finally, figures 5.2-7 and 5.2-8 show the latency for the deployment-related operations for each of the two considered scenarios.

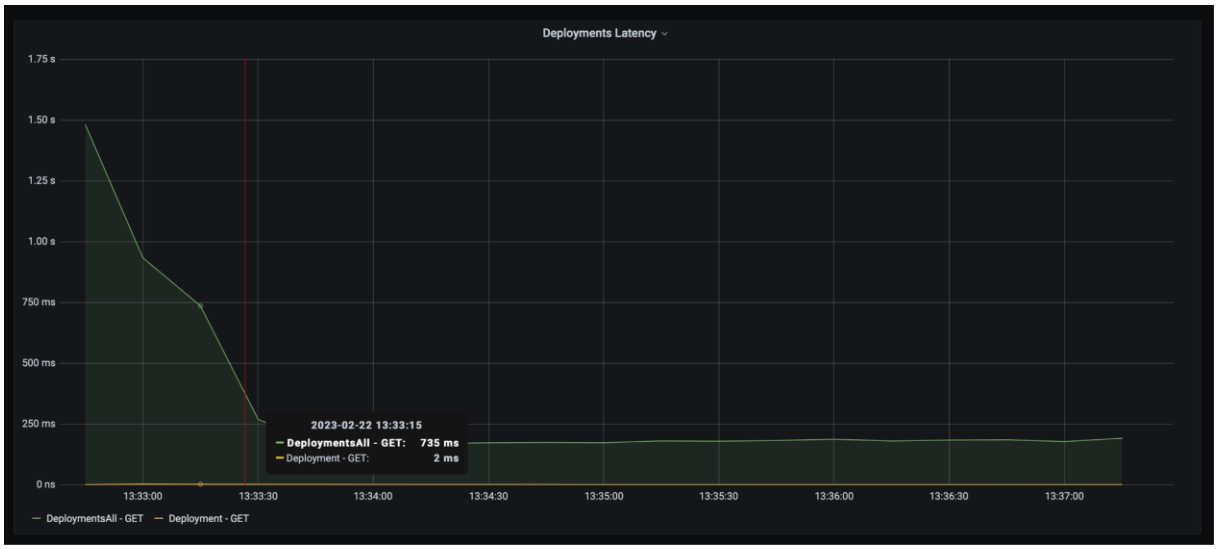

Figure 5.2-7: SCC deployment latency for scenario 1.

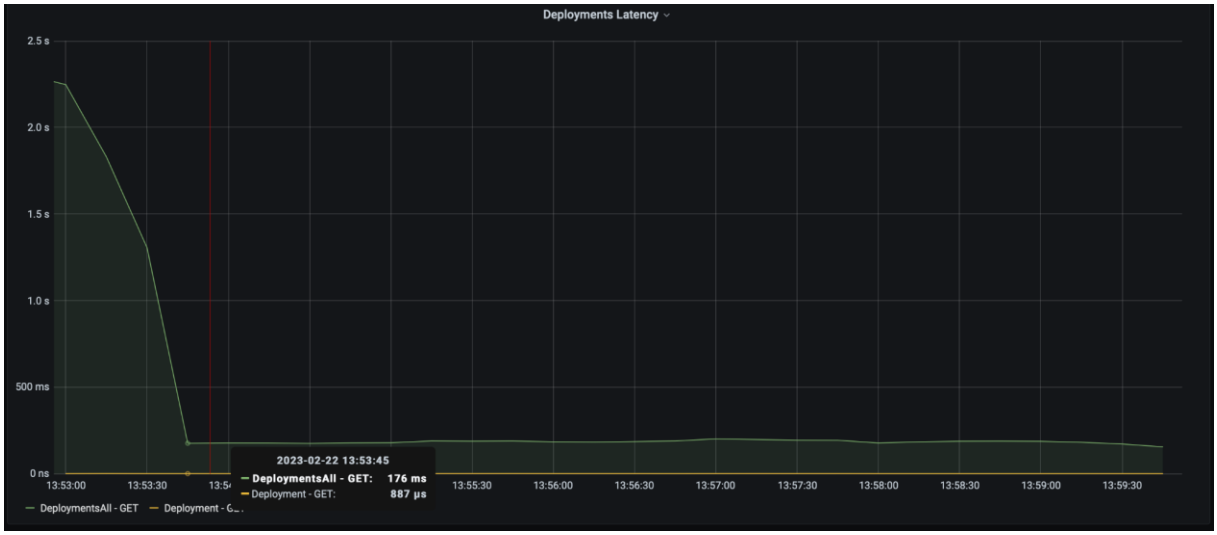

Figure 5.2-8: SCC deployment latency for scenario 2.

The measurements indicate that the Security Capabilities Catalogue performs well for both workloads (25 / 35 Concurrent Operating Users). The results are more than sufficient with a single instance of the SCC. The following list highlights the most important aspects of the performance evaluation:

- The memory consumption and the CPU workload of the application is very low.
- The information retrieval regarding the registration and the deployment operations of the SCC is below 200ms but the most important is that this time is approximately the same for both workloads, which shows that the application scales well in even greater workloads.
- Search API is the most resource consuming API of the SCC, however the average latency of the API is acceptable. Same as above, the API scales well for both workload scenarios.
- Of course, the SCC was fully responsive during all performance scenarios.

6. Conclusions

This deliverable concludes the description of the work carried out in WP3 and updates the decisions made in D3.1 at the different stages of work in WP3.

Specifically, it first summarises the design or architecture of each component and subcomponent and highlights any change incurred since D3.1. After that, the updates on the specifications are presented, showcasing the mapping of the requirements to the inner logic realised by each task. Their up-to-date relevance and adequacy in the context of the project is evaluated, and the delta is provided to indicate whether there are additions, modifications, or deletions regarding the specifications.

Similarly, this document reviews the selection of technologies to both develop and deploy the logic at the latest stage of implementation for each task; and follows with both integrated and unitary evaluation measurements regarding the behaviour of the involved elements.

Besides these considerations, D3.2 also compiles other up-to-date low-level details on the design and implementation governing the logic presented beforehand; and which relate to the modelling and APIs (internal or external) per component and subcomponent, as well as the assessed compliance of each element regarding the GDPR.

The outcome of this document serves as well to set a technical milestone, used to integrate with the other technical WPs (WP4 and WP5) in the context of WP6. The following steps tackle the finalisation of such integration, its evaluation and documentation in the upcoming WP6 deliverables.

7. References

[1] Apache, “Kafka: open-source distributed event streaming platform”, https://kafka.apache.org

[2] Cisco, “NetFlow: IP network traffic collector”, https://en.wikipedia.org/wiki/NetFlow

[3] Zeek, “Zeek: Open Source Network Security Monitoring Tool” https://zeek.org/

[4] Kubernetes, “Kubernetes” https://kubernetes.io/es/

[5] Docker, “Dockerhub” https://hub.docker.com/

[6] fprobe, “fprobe” https://manpages.ubuntu.com/manpages/bionic/man8/fprobe.8.html

[7] softflowd, “softflowd”, https://manpages.ubuntu.com/manpages/bionic/man8/softflowd.8.html

[8] Wireguard, “Wireguard” https://www.wireguard.com/

[9] Wazuh, “Wazuh” https://wazuh.com/start

[10] ntop, “ntop’s nTap” https://www.ntop.org/guides/ntap/installation.html

[11] Pallets, “Flask: web development, one drop at a time” https://flask.palletsprojects.com

[12] S. Ramírez, “FastAPI framework, high performance, easy to learn, fast to code, ready for production” https://fastapi.tiangolo.com

[13] Uvicorn, “Uvicorn: an ASGI web server for Python” https://www.uvicorn.org

[14] SmartBear Software, “OpenAPI: API description format for REST APIs” https://swagger.io

[15] Markdown, “The simple and easy-to-use markup language” https://www.markdownguide.org

[16] virtualenv, “Python Virtual Environments: A Primer” https://realpython.com/python-virtual-environments-a-primer

[17] Docker, Inc., “Docker: Empowering App Development for Developers” https://www.docker.com

[18] Docker, Inc., “Docker Compose: tool for defining and running multi-container Docker applications” https://docs.docker.com/compose

[19] Cloud Native Computing Foundation, “Kubernetes: open-source system for automating deployment, scaling, and management of containerised applications”, https://kubernetes.io

[20] Open Infrastructure Foundation, “OpenStack: Open Source Cloud Computing Infrastructure”, https://www.openstack.org

[21] MongoDB Inc., “MongoDB: the application data platform”, https://www.mongodb.com

[22] Keycloak, “Identity and access management”, https://www.keycloak.org/

[23] SurveyJS, https://surveyjs.io/

[24] ENISA, a simplified approach for SMEs, https://www.enisa.europa.eu/publications/archive/RMForSMEs

[25] LimeSurvey, “LimeSurvey: online survey tool”, https://www.limesurvey.org/

[26] Gnuplot, “Gnuplot website”, http://www.gnuplot.info

[27] Compastié, M.; López Martínez, A.; Fernández, C.; Gil Pérez, M.; Tsarsitalidis, S.; Xylouris, G.; Mlakar, I.; Kourtis, M.A.; Šafran, V. PALANTIR: An NFV-Based Security-as-a-Service Approach for Automating Threat Mitigation. Sensors 2023, 23, 1658. https://doi.org/10.3390/s23031658

[28] PALANTIR, “Open source scripts and data for evaluation of the NFV-based remediation paper”, https://github.com/palantir-h2020/paper-nfv-aas-threat-mitigation/tree/master/6_3_2-sc-orchestration

[29] Apache, “JMeter: load testing and performance measurement tool”, https://jmeter.apache.org

[30] Grafana Labs, “Grafana: observability stack”, https://grafana.com

8. Annex A: modelling

This annex complements the design with extra details on the entities used for modelling the behaviour of the WP3 components and subcomponents, through some specific assumptions, properties, and roles.

8.1. Ontology

A complete data modelling allows PALANTIR to act consistently and exhaustively against different threats and anomalies. In this context, the ontology defines the information used by the involved elements in different processes and tasks. The final intent of this ontology is to best understand which SC to leverage under certain conditions.

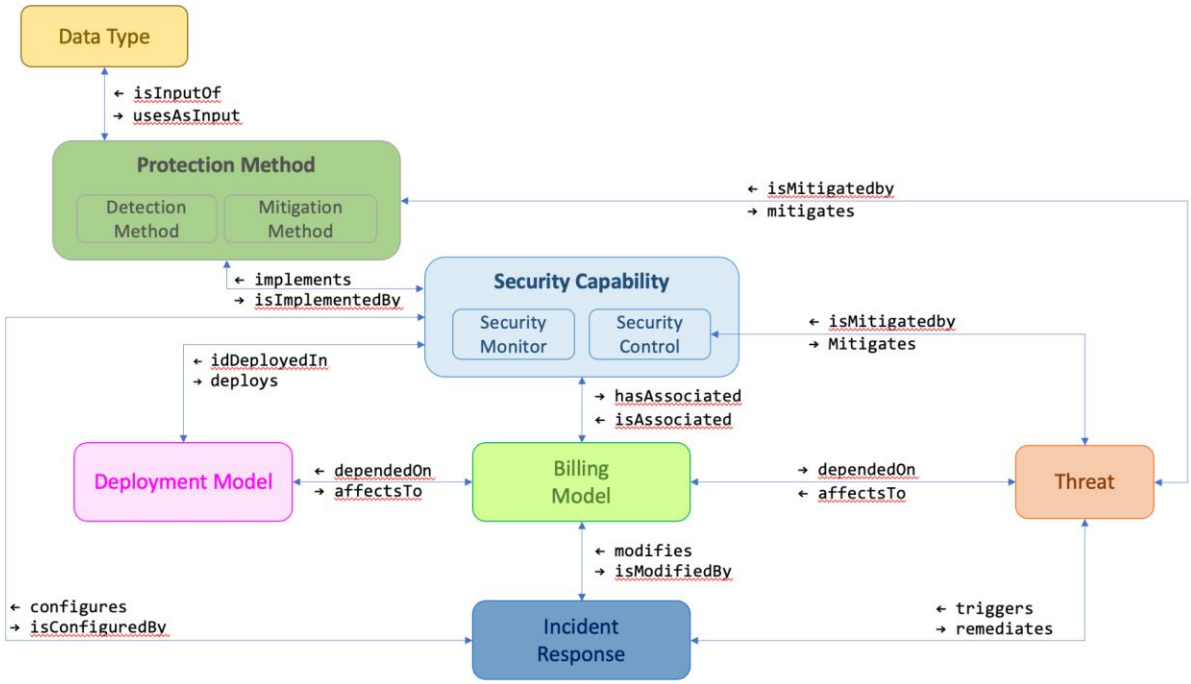

Figure 8.1-1: Ontology defined for the formalisation of data in PALANTIR.

Figure 8.1-1 depicts the ontology with its different classes and relations. It contains seven classes, which relate to the main PALANTIR components from WP3, WP4 and WP5 involved in the remediation procedures. Their significance and relationships are described below.

- The ***Data Type*** represents the different data modalities contemplated in PALANTIR. In this case, the NDS SC (NetFlow and Zeek logs) and SIEM SC (host-based logs) are providers of data. This class is connected to the Protection Method one and uses the latter as input.
- The ***Protection Method*** class defines the detection and mitigation methods (i.e. the two families of SCs) available in PALANTIR as its subclasses. It is connected to two specific classes: Security Capability and Threat.
- The ***Security Capability*** class does refer to the security service developed to be deployed in the client infrastructure. This class supposes a central part of the ontology because it has relation with a great part of the ontology classes. Also, a Security Capability can implement one or more protection methods and is related to the Threat class, contributing to mitigate the effects represented by the latter.
- The ***Threat*** class defines the threats/attacks classified by the WP5 components, where the categorization and remediation procedure are performed. This class is also connected to the

Billing Model and Incident Response classes because it directly affects the data allocated in both classes.

- The ***Billing Model*** class represents the different fees to be applied regarding the deployed SCs and the contracted characteristics. This class identifies the specific component allocated in WP4 in charge of this functionality. The connection between the Billing Model and Threat classes is due to a Threat generating a remediation procedure that deploys a new security service. This fact is also represented by the connection between the Billing Model, Incident Response, and Security Capability classes.
- Besides, the Billing Model depends on the ***Deployment Model***; where the latter defines the available deployment models (or delivery modes), which have associated different physical resources, operation modes, etc. The Security Capability class is also affected by this class since, for instance, the physical resources can impact the instantiation of the final security service.
- Finally, the ***Incident Response*** class models the element (belonging to WP4) selecting and performing the remediation procedures when a threat/attack is detected. Therefore, it connects with Threat and Billing Model classes, already commented, but also with Security Capability class, since the remediation procedure (initiated by the WP5 components) to perform can result in a reconfiguration of a security service deployed in the client infrastructure.

8.2. Multi-tenancy model

The model that allows multi-tenancy in the SO was designed following integration guidelines to comply with the different delivery modes, after D3.1 was submitted. The multi-tenancy approach introduces more flexibility in the management of the resources and allows the SO to keep track of specific tenants; where they are organisations as a whole or even specific users that manage a given infrastructure, wish to protect another, do some kind of operation or management and/or subscribe to the SCs. Tenancy is relevant to all operations that require a filtered view of managed resources, as well as for those related to accountability, e.g. to keep track of subscribed SCs and their instantiation to compute consumption.

Initially, a simple model without multi-tenancy was enforced in SO, as depicted in figure 8.2-1; in such a manner that only one tenant was considered at the time (i.e. the PALANTIR project itself). This implies that a single *tenant* sets up a single OSM (the NFVO) to take care of multiple infrastructures (or Kubernetes clusters), depending on the Use Case and/or delivery mode involved.

Figure 8.2-1: initial model to manage multiple infrastructures in SO.

On the other hand, the new modelling incorporates more entities to support the operation with multiple tenants. Besides allowing filtered views and accountability, this new model introduces a more flexible assumption where a tenant may have multiple NFVO instances (like OSM) to control different segments of their network. This is shown in figure 8.2-2.

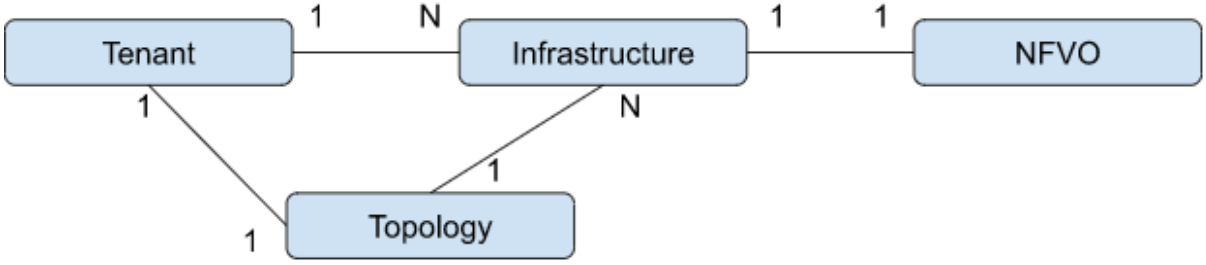

Figure 8.2-2: current model to manage multiple tenants and infrastructures in SO.

Besides the above mentioned, this model has other implications on the workflows defined within the SO; which now requires first to register a ***Tenant***, then one or more ***NFVO*** (OSM) instances and one or more ***Infrastructure***, and finally a ***Topology*** that holds a view of all the managed network.

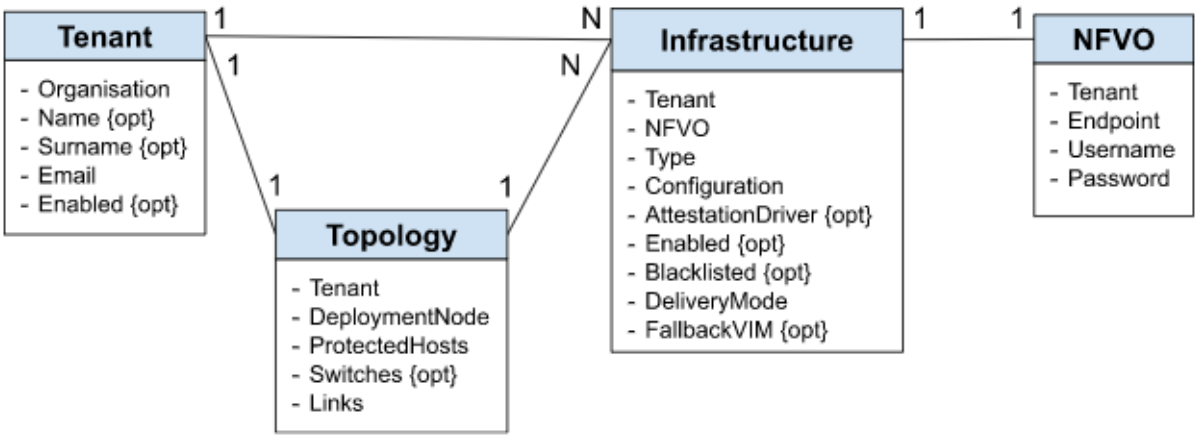

Figure 8.2-3: involved entities on the new model and their attributes.

The Topology entity helps provide a fine-grained view on both the protected infrastructure (i.e. the hosts of the organisation to be protected) and the deployment nodes where the SC instances are deployed (i.e. the VIM), as well as how these are connected. The properties of each entity, as well as whether these are mandatory or optional and their relation are shown in figure 8.2-3.

8.3. SCC descriptors

The original set of descriptors for the SCC were presented in D3.1. Table 8.3-1 introduces two new entries to extend such descriptors.

Table 8.3-1: new entries for the SCC descriptors.

Name	Technical name	Purpose, description	Contributors	Ext. availability	Searchability	Example
Internal use	internal_use	Explains if the SC is private (for use only inside the organisation of the developer) or public	Dev	Y	N	true
Developer's organisation ID	dev_org_id	A unique ID for the developer, corresponds to Keycloak ID of the organisation	Dev	Y	Y	<unique id>

9. Annex B: APIs

9.1. Intra-component APIs

The following section provides the changes introduced in D3.2 regarding the intra-component APIs that were initially provided in the annexes of D3.1.

9.1.1. Intra-component APIs for SCs

The SCs does not implement an intra-component API as usual to expose the day2 actions available. In contrast, the SCs use a *juju controller* to perform this task. This component is deployed as a pod together with the SC deployment, and it establishes a SSH connection to trigger the day2 action specified. This mechanism is the unique one allowed in the SC to interact with the security service instantiated via SO.

9.1.2. Intra-component APIs for SO

The internal communication between the modules in the SO is carried out fully via REST APIs, described below per each of them.

The AAC module (table 9.1.2-1) has focused on the authentication process, reshaping its logic to both validate and generate a token, adapting both endpoints, query parameters and payloads.

Table 9.1.2-1: description of the SO's internal API exposed by the AAC module.

Endpoint	Path query	Method	Request body
/token	id	GET	-
Retrieves information on the token, such as its temporal validity and the operations that are allowed or authorised on it.			
/token	-	POST	{"tenant-id": "...", "source-entity": "...", "credentials": {...}}
Creates a new authentication token, based on the requestor's identity. Credentials can accept both basic auth (with "user" and "password" fields) or Public Key Infrastructure (PKI)-based approaches ("certificate" field).			

In the ATR module (table 9.1.2-2), the runtime endpoint has increased the granularity to obtain metrics from different entities related to the infrastructure, whether the infrastructures or clusters, their nodes, or the running instances of the SCs. The attestation endpoint has a slight simplification on its payload.

Table 9.1.2-2: description of the SO's internal API exposed by the ATR module.

Endpoint	Path query	Method	Request body
/runtime	- infra-id node-id service-id	GET	-
Requests all available information for different levels of granularity (the whole infrastructure/cluster, a specific node on it or a specific service or running SC).			

/attestation	node-id	POST	{“status”: “...”, “reason”: “...”}
Communicates a new attestation incident report to the SO, by uniquely identifying the ID of the failed node that was first attested by TAR, as well as the reported attestation status and the reason.			

The CFG module (table 9.1.2-3) is assigned some endpoints to cover some system-wide new logic, related to the registration of tenants, infrastructure (or clusters), NFVO instances and the overseen network topology that is bound to each tenant.

Table 9.1.2-3: description of the SO’s internal API exposed by the CFG module.

Endpoint	Path query	Method	Request body
/infrastructure	- id	GET	- -
Requests all available information for the totality or a specific node from the infrastructure.			
/infrastructure	-	POST PUT	{“type”: “kubernetes”, “config_file”: “...”, “delivery_modes”: “[...]”, “fallback”: “false|true”, “tenant_id”: “...”, “nfvo_id”: “...”}
/infrastructure	id	DELETE	-
Deletes the record of the infrastructure identified by the specific ID.			
Registration (or update) of a new infrastructure node(s) (e.g. typically a Kubernetes cluster with 1+ servers), associated with a specific tenant and NFVO (i.e. OSM).			
/nfvo	- id	GET	- -
Requests all available information for the totality or a specific NFVO (i.e. OSM).			
/nfvo	-	POST PUT	{“endpoint”: “...”, “username”: “...”, “password”: …”, “tenant_id”: “”}
Registration (or update) of a new NFVO (i.e. OSM), associated with a specific tenant.			
/nfvo	id	DELETE	-
Deletes the record of the NFVO identified by the specific ID.			
/tenant	- id	GET	- -
Requests all available information for the totality or a specific tenant/organisation.			
/tenant	-	POST	{“name”: “...”, “surname”: “...”, “email”: …”, “organisation”: “”}

Registration of a new tenant, where only the organisation field is required.			
/tenant	id	DELETE	-
Deletes the record of the tenant identified by the specific ID.			
/topology	- tenant-id	GET	- -
Requests all available network topologies, or that associated to a specific tenant.			
/topology	tenant-id	POST PUT	{“topology”: “...”}
Registration (or update) of a new topology for the specified tenant.			
/topology	tenant-id	DELETE	-
Deletes the record of the topology associated with the specific tenant.			

LCM (table 9.1.2-4), as one of the most central modules to SO (since it manages the lifecycle of the running SCs), there were a number of changes.

The most cosmetic ones relate to the “ns/” endpoint changing the path query parameter (i.e. the parameter passed along with the endpoint) changing slightly, thus shown here; as well the previous “/vnf” endpoint being renamed to “/xnf” to provide more abstraction to the concept of the instance being deployed, while aligning with the fact that VNFs (and thus, VMs) are no longer being the default, and instead any VNF or CNF would be treated homogeneously.

The rest of the changes were the introduction of multiple other functionalities, both for requesting different levels of details on the running SCs (which are later displayed by the Dashboard or used by the AE, among others) and also for submitting requests to instantiate a new SC, configure an already existing SC instance or a combination of both.

Table 9.1.2-4: description of the SO’s internal API exposed by the LCM module.

Endpoint	Path query	Method	Request body
/ns /ns/	- id	GET	- -
Requests all available information for the totality or a specific running instance of the SCs (deployed as NSs) that are already instantiated and running. This provides information like the specific packaged used, the operational and configuration status, the VIM (i.e. infrastructure) where it is deployed or the contained xNF ID.			
/ns	ns_pkg_id	POST	{“action_name”: “...”, “action_params”: {“...”: “...”}, “description”: “...”, “name”: “...”, “ssh_key”: [“...”], “vim_id”: “...”}
Instantiates a new SC (as a Network Service), where the name field is the only required.			
/ns/	id, force	DELETE	-

Deletes a specific running instance of the SC (as a NS). Optionally, it can immediately remove it from the system via the force parameter.			
/ns/action/	ns_id, action_id	GET	-
Requests all available information on the list of executed actions on a given running instance of the SC (as a NS). Optionally, it can filter a specific action if its action_id is provided.			
/ns/action/	id, wait_for	POST	{“action_name”: “...”, “action_params”: {“...”: “...”}}
Submits an action to a specific running instance of the SC (as a NS). The action_name is the only required field in the body. Optionally, the action_params field in the body encodes in a dictionary the number of expected parameters and values. Also optionally, the wait_for parameter helps determine which kind of information is expected (which is directly related with the time the operation will take).			
/ns/health/	id	GET	-
Requests all available information on the health calculated for a given SC. This health check is performed internally to the running SC instance and exposed via a REST API, here proxied.			
/xnf /xnf/	- xnf-id	GET	- -
Requests all available information for the totality or a specific building block of the running SC (i.e. its internal xNF). Compared to the /ns endpoint, this provides more infrastructure, low-level details, such as its IP, the Kubernetes cluster where it is deployed and the assigned namespace; as well as again the relation to the NS and its operational and configuration status.			

The MON module (table 9.1.2-5) is another component that has suffered a number of changes, since it retrieves some of the details requested by the LCM module itself and propagated to other PALANTIR components. There are also some cosmetic changes here, where the “vim/” endpoint from D3.1 was renamed to “infra/”, and the “vnf/” endpoint followed the same principles as above. Besides this, other endpoints were added to request or query metrics from the running SC instances.

Table 9.1.2-5: description of the SO’s internal API exposed by the MON module.

Endpoint	Path query	Method	Request body
/infra /infra/	- id	GET	- -
Requests all available specifications and other technical details for the totality or a specific infrastructure. This returns both the initially registered information per infrastructure (or Kubernetes cluster) and details on their available resources, per node belonging to the cluster; as well as the VIM ID generated by OSM, which can be used to instantiate NSs in LCM.			
/infra/service /infra/service/	- infra_id, id	GET	- -

Requests all available information on the running services (i.e. SCs) for the totality or a specific infrastructure, as well as for the totality or a specific running SC instance. This returns the list of running pods (i.e. the minimal unit managed by Kubernetes) that conform the running SC instances, with low-level details such as the IP of the specific Kubernetes cluster node where the SC instance is deployed (as a pod), its name and internal IP within that cluster node, as well as information on the container ID and image it is using. This latter information is especially relevant for the AE, so to determine their integrity.			
/metrics	xnf-id metric-name	GET	-
Requests details on the generic metrics returned from one or all registered Prometheus targets, as well as for one or all metrics.			
/metrics/alerts	xnf-id metric-name	GET	-
Retrieves a list of registered alerts registered in one or all registered Prometheus targets, as well as relating these to any specific metric.			
/metrics/background	-	POST	{“xnf-id”: “...”, “metric-name”: “...”, “metric-command”: “...”}
Starts a background monitoring process in a specific Prometheus target, fetching the provided custom metric with a specific command to run.			
/metrics/node	-	POST	{“xnf-id”: [“...”]}
Manually install a Prometheus instance in the target. One or more targets’ IPs are provided, as a list, in the payload.			
/metrics/node	xnf-id	DELETE	-
Manually uninstall the Prometheus instance from the target, identified by the query parameter.			
/metrics/xnf	-	POST	{“xnf-id”: “...”, “metric-name”: “...”, “metric-command”: “...”}
Registers a new metric and obtains a specific custom metric from a specific Prometheus target.			
/targets	-	GET	-
Retrieves the complete list of registered Prometheus targets.			
/targets	-	POST	{“url”: “...”, “force”: “...”}
Registers a new Prometheus target. It is possible to register a target with the same ip as an already registered target but with different port by setting the “force” parameter to “true”			
/targets	-	PUT	{“current-url”: “...”, “new-url”: “...”}
Replaces an already registered Prometheus target with a new one, by changing the IP of the original			

node to the IP of another one.			
/targets	url	DELETE	-
Deletes a specific Prometheus target from the registry or all of them.			
/targets/metrics	xnf-id metric-name	GET	-
Requests details on the metrics (both generic or custom) returned from one or all registered Prometheus targets, as well as for one or all metrics.			

The PKG module (table 9.1.2-6) is used for the onboarding of the SC definition or specification, consisting of both an xNF and an NS package. There were also a number of changes on this front. The most cosmetic changes relate to the renaming of the "/ns-pkg" to the "/ns" endpoint, and similarly, from "/vnf-pkg" to "/vnf". In the same manner, the "pkg-id" path query parameter (for each of these endpoints) was renamed to "id". Besides this, the "name" path query parameter was also added to filter also in an easier way (per package name). Finally, the DELETE method is also introduced to support removal of specific packages; so, to avoid future instantiation of such SCs.

Table 9.1.2-6: description of the SO's internal API exposed by the PKG module.

Endpoint	Path query	Method	Request body
/ns /ns/	- id, name	GET	-
Requests all available information on the NS package belonging to the SC specification. Optionally, this can provide details on a specific NS if either (or both) the "id" or "name" path query parameters are passed.			
/ns	-	POST	package (multipart/form-data)
Submits the binary package (.tar.gz file) that contains the NS package belonging to the SC specification and records it in the NFVO. This package is the last one to be uploaded, as it depends on the xNF package.			
/ns/	id	DELETE	-
Deletes the NFVO record of the specific NS package belonging to the SC specification. This record is the first one to be removed, as the xNF record is referenced by it.			
/xnf /xnf/	- id	GET	-
Requests all available information on the xNF package belonging to the SC specification. Optionally, this can provide details on a specific xNF if either (or both) the "id" or "name" path query parameters are passed.			
/xnf	-	POST	package (multipart/form-data)
Submits the binary package (.tar.gz file) that contains the xNF package belonging to the SC			

specification and records it in the NFVO. This package is the first one to be uploaded, as the NS package depends on it.			
/xnf/	id	DELETE	-
Deletes the NFVO record of the specific xNF package belonging to the SC specification. This record is the last one to be removed, as the NS record references it.is referenced by it.			

Finally, the POL module (table 9.1.2-7) had its main goal changed to be an ancillary element, related to the MON module. Endpoints were added to retrieve related metrics and to set alerts based on specific conditions.

Table 9.1.2-7: description of the SO's internal API exposed by the POL module.

Endpoint	Path query	Method	Request body
/alerts	-	GET	-
Retrieves a complete list of registered alerts			
/alerts	-	POST	{"alert-name": "...", "threshold": "...", "operator": "...", "time-validity": "...", "hook-type": "webhook", "hook-endpoint": "..."}
Registers a new alert by setting a name, a threshold (int), operator (<, >, <=, >=, ==), time validity (int), hook type (webhook) and hook endpoint.			
/alerts	-	PUT	{"alert-name": "...", "threshold": "...", "operator": "...", "time-validity": "...", "hook-type": "webhook", "hook-endpoint": "..."}
Replaces an existing alert by setting the new data			
/alerts	-	DELETE	-
Deletes all registered alerts.			
/events	-	POST	{"alert-name": "...", "metric-name": "..."}
Triggers a comparison between the current metric obtained from the last background monitoring and against the threshold set in the alert related to that metric.			
/metrics	xnf-id metric-name	GET	-
Retrieves a list of all metrics (associated with the alerts) from the Prometheus targets.			
/metrics	-	POST	{"xnf-id": "...", "metric-name": "...", "metric-command": "..."}
Activates background monitoring on the Prometheus target to fetch metrics associated to the alert			

9.1.3. Intra-component APIs for SCC

The SCC is implemented as a standalone component, following a monolithic approach. Therefore, it does not require intra-component APIs; but rather load the specific logic directly.

9.1.4. Intra-component APIs for RAF

Table 9.1.4-1 lists the internal communications that the RAF needs to operate, as a REST API.

Table 9.1.4-1: description of the RAF's internal APIs.

Endpoint	Path query	Method	Request body
/list_org_ids	-	GET	-
Returns a list of all organisation ids that have completed the questionnaire			
/results	org_id	GET	-
Returns the risk analysis results of a specific organisation with the given id			
/compute_results	-	POST	{"survey_data": "..."}
This endpoint is used to communicate the data of the questionnaire to the back-end engine of RAF.			
/exists	org_id	GET	-
This endpoint returns a boolean value if the org_id exists in the results database.			

9.2. Inter-component APIs

The following section leverages the inter-component APIs which enable communication across the PALANTIR components. These are provided in full in this section, since there is no previous iteration of this content provided in D3.1.

9.2.1. Inter-component APIs for SCs

The juju controller deployed for each SC exposes the needed interface to provide affordable communication with SO. This communication is used to trigger day2 actions in the security service deployed. The list of day2 actions available is listed in the xNF descriptor, where the day2 action details (name, parameters, etc.) are presented. Moreover, the SC uses the SEM (L-SL functionality), as commented in section 3.1, to send the logs and monitoring data collected into it. This information is delivered into the specific Kafka topic used for each SC type. The Kafka topic name format for each SC is *sc.generated_log.x*, being *x* the name of the security service software used. An example is *sc.generated_log.suricata* for the logs created by the IDS SC implemented with Suricata software.

9.2.2. Inter-component APIs for SO

On the one hand, the SO exposes the vast majority of its logic via REST APIs for other PALANTIR components to contact it. The API module is the one publicly exposing this logic to other PALANTIR components, thereby offering a subset of the functionality provided by the rest of the SO modules.

First of all, endpoints are publicly documented and accessible via OpenAPI in the following Uniform Resource Identifier (URI):

```
<so_ip>:50101/api/docs
```

Endpoints are publicly exposed via URIs that implement the following naming scheme:

```
<so_ip>:50101/<so_module_name>/<so_module_endpoint_name>
```

Therefore, the intra-component APIs provided in 8.3.2 would be reachable, for instance, as follows:

```
<so_ip>:50101/mon/infra/service
```

On the other hand, the communication with other components in PALANTIR is carried out through Kafka topics, where each of them is consumed by their specific component. This type of communication is typically restricted to notification events with relevant information (e.g., for the Dashboard and AE), but also to some information request or exchange (e.g., with SM).

9.2.3. Inter-component APIs for SCC

All the APIs from the SC Catalogue (SCC) are publicly documented and accessible via OpenAPI in the following URI:

```
<scc_ip>:8080/q/swagger-ui
```

Table 9.2.3-1 lists the external REST API endpoints offered by the SCC. It is worth noting that the payload in some endpoints (i.e., /api/v1/register and /api/v1/search) were trimmed for readability.

Table 9.2.3-1: description of the SCC's inter-component API.

Endpoint	Path query	Method	Request body
/api/v1/deploy /api/v1/deploy/	- id	GET	-
Retrieve SC deployment status.			
/api/v1/deploy /api/v1/deploy/	- id	POST	{"id": "...", "name": "..."}
Deploy a registered SC.			
/api/vi/deploy/	id	DELETE	
Terminate and remove a SC.			
/api/v1/register	-	POST	{{"vnf": {...}, "security": {...}, "billingSLA": {...}, "xnfId": "...", "nsId": "...", "internal_use": "..."}
Register a new SC.			
/api/v1/register	-	GET	

/api/v1/register/	id		
Get registered security capability information.			
/api/v1/register/	id	DELETE	
Delete a security capability from the SCC.			
/api/v1/register/	id	PUT	
Update a registered security capability.			
/api/v1/search	-	POST	{{“vnf”: {...}, “security”: {...}, “billingSLA”: {...}, “xnfId”: “...”, “nsId”: “...”}
Search SCs based on metadata and descriptors.			
/api/v1/health/	id	GET	
Retrieve deployed security capability health status.			
/api/v1/onboard /api/v1/onboard/	- id	GET	
Get status of an onboarding job			
/api/v1/onboard		POST	{“id”: “...”}
Onboard a registered security capability to the SO.			

9.2.4. Inter-component APIs for RAF

RAF allows an easy integration with the PALANTIR Portal by exposing a REST endpoint, as described in Table 9.2.4-1.

Table 9.2.4-1: description of the RAF’s external APIs.

Endpoint	Path query	Method	Request body
/dashboard/	-	POST	{“org_id”: “...”}
Provides a connection interface with the PALANTIR Dashboard.			

10. Annex C: GDPR compliance

This annex reviews the regulatory compliance, taking into account the GDPR regulation for each component or subcomponent. This formalisation has considered the data needed for analysis and processing, as well as the location from where the data is taken and where it has been processed.

Whilst some details vary per component or subcomponent, others can be grouped and explained as a whole, as these affect the whole PALANTIR platform rather than specific elements. This is the case of the Data Protection Officer (DPO), where the same person or organisation operating the PALANTIR platform should be appointed as DPO. Also, the consent on the processing of the data subject information shall be explicitly informed and collected during the user registration process in the PALANTIR platform (i.e. through the Dashboard). This will aggregate the consent to all further personal data processing for the SCC, the SCs and the SO.

10.1. GDPR compliance for SCs

Of all the elements that make up the PALANTIR platform, the Security as a Service (SecaaS) enablers are the ones that are constantly accessing the SME/ME information for analysis and protection. The following reviews the compliance for each of the SecaaS enablers, which constitute the different SC offered by PALANTIR as protection and security mechanisms.

Table 10.1-1: GDPR compliance assessment for the IDS SC.

Security Capability	**Intrusion Detection System (IDS)**
xNF name	Snort / Suricata
xNF description	Network traffic analysis to identify possible malicious activities according to rules based on security policies.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to infrastructures and third-party software)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Outputs	Alerts and detected events through Kafka.
Formats	XML, JSON.
Data STORAGE and USABILITY	
Personally Identifiable Information	IPs from client hosts.

Sensitive data	N/A. No sensitive data is stored.
Data Retention	No data is retained in the xNF.
Data Encryption	N/A. There is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	IPs from client hosts.
Sensitive data	N/A. No sensitive data is processed.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of IDSs includes cybersecurity to ensure local detection of anomalies that could subvert the interests of service providers.
Data SHARING	
Transfer and sharing	The xNF only shares alerts and events with the TI pipeline, internally in the platform.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

Table 10.1-2: GDPR compliance assessment for the NDS SC.

Security Capability	**Network Data Sniffer (NDS)**
xNF name	NetFlow / Zeek
xNF description	Network traffic sniffer to act as a collector for other PALANTIR components and TI functioning.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to infrastructures and third-party software)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Outputs	NetFlow and Zeek data.
Formats	JSON.
Data STORAGE and USABILITY	
Personally Identifiable Information	IPs from client hosts.
Sensitive data	N/A. No sensitive data is stored.
Data Retention	No data is retained in the xNF.
Data Encryption	Not applicable because there is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	IPs from client hosts.
Sensitive data	N/A. No sensitive data is processed.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of NDSs includes cybersecurity to ensure local collection of data that could subvert the interests of service providers.

Data SHARING	
Transfer and sharing	The xNF only shares data with the TI pipeline, internally in the platform.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

Table 10.1-3: GDPR compliance assessment for the FW SC.

Security Capability	**Firewall and Router (FW)**
xNF name	iptables
xNF description	Network traffic management to filter and control traffic when attacks/threats are detected.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to infrastructures and third-party software)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Outputs	N/A. No output data is generated.
Formats	N/A. No output data is generated.

Data STORAGE and USABILITY	
Personally Identifiable Information	N/A. No personal data is stored.
Sensitive data	N/A. No sensitive data is stored.
Data Retention	No data is retained in the xNF.
Data Encryption	Not applicable because there is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	N/A. No sensitive data is processed.
Sensitive data	Network traffic is managed to filter and allow traffic flows regarding possible threats detected.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of FWs includes cybersecurity to ensure local filtering of network anomalies that could subvert the interests of service providers.
Data SHARING	
Transfer and sharing	No data is sent or shared.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.

Portability	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.

Table 10.1-4: GDPR compliance assessment for the VPN SC.

Security Capability	**Virtual Private Network (VPN)**
xNF name	Wireguard
xNF description	Network traffic protection to secure the communications of PALANTIR clients.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to infrastructures and third-party software)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Outputs	N/A. No output data is generated.
Formats	N/A. No output data is generated.
Data STORAGE and USABILITY	
Personally Identifiable Information	Identifiers of the VPN users.
Sensitive data	N/A. No personal data is stored.
Data Retention	Information about connections allowed.
Data Encryption	Information secured in a private scope not accessible outside.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	N/A. No sensitive data is processed.
Sensitive data	N/A. No sensitive data is processed.

Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of VPNs includes cybersecurity to protect the communications.
Data SHARING	
Transfer and sharing	The xNF does not share information except if needed.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

Table 10.1-5: GDPR compliance assessment for the SIEM SC.

Security Capability	**Security Information and Event Management (SIEM)**
xNF name	Wazuh
xNF description	Host traffic and behaviour analysis to identify and mitigate possible malicious activities according to rules based on security policies.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to	The SC uses the day2 actions through the SO to receive input

infrastructures and third-party software)	regarding configuration.
Outputs	Alerts, detection events.
Formats	JSON.
Data STORAGE and USABILITY	
Personally Identifiable Information	IPs and user behaviour logs.
Sensitive data	No sensitive data is stored. The SIEM stores system logs, tracking basic data that induced changes in the OS' file system.
Data Retention	No data is retained in the xNF.
Data Encryption	Not applicable because there is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	IPs and users' behaviour logs used in client hosts.
Sensitive data	No sensitive data is processed. The SIEM operates on metadata extracted from operations carried out in the file system.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of IDSs includes cybersecurity to ensure local detection and protection of anomalies that could subvert the interests of service providers.
Data SHARING	
Transfer and sharing	The xNF only shares alerts and events with the TI pipeline, internally in the platform.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.

Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

Table 10.1-6: GDPR compliance assessment for the vTAP SC.

Security Capability	**Virtual Terminal Access Point (vTAP)**
xNF name	nTap
xNF description	Traffic mirroring to allow IDS to work with a copy of traffic.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to infrastructures and third-party software)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Outputs	Copy of traffic.
Formats	Network traffic.
Data STORAGE and USABILITY	
Personally Identifiable Information	IPs of client hosts.
Sensitive data	N/A. No sensitive data is stored.
Data Retention	No data is retained in the xNF.
Data Encryption	Not applicable because there is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.

Data PROCESSING	
Personally Identifiable Information	IPs of client hosts.
Sensitive data	N/A. No sensitive data is processed.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of vTAPs includes cybersecurity to ensure traffic mirroring that could subvert the interests of service providers.
Data SHARING	
Transfer and sharing	The xNF does not share information except if needed.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

Table 10.1-7: GDPR compliance assessment for the SCR SC.

Security Capability	**Script-based (SB)**
xNF name	Custom OS scripts.
xNF description	Scripts to perform monitoring and mitigation tasks into the client hosts.
On data, interfaces and formats	
Inputs (interfaces exposed	SIEM SC is able to execute the scripts with possible parameters

to the operator)	established.
Inputs (interfaces exposed to other PALANTIR components)	SIEM SC is able to execute the scripts with possible parameters established.
Inputs (connectors to infrastructures and third-party software)	SIEM SC is able to execute the scripts with possible parameters established.
Outputs	Internal log.
Formats	Plain text.
Data STORAGE and USABILITY	
Personally Identifiable Information	N/A. No personal data is stored.
Sensitive data	N/A. No sensitive data is stored.
Data Retention	No data is retained in the xNF.
Data Encryption	Not applicable because there is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	N/A. No personal data is stored.
Sensitive data	N/A. No sensitive data is stored.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of SCRs includes cybersecurity to ensure local mitigation of anomalies that could subvert the interests of service providers.
Data SHARING	
Transfer and sharing	The xNF does not share information with other PALANTIR components.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.

Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

Table 10.1-8: GDPR compliance assessment for the WTA SC.

Security Capability	Web-based Traffic Analysis (WTA)
xNF name	Ntopng
xNF description	Web traffic analysis to identify possible malicious activities according to rules based on security policies.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (interfaces exposed to other PALANTIR components)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Inputs (connectors to infrastructures and third-party software)	The SC uses the day2 actions through the SO to receive input regarding configuration.
Outputs	Alerts and detection events.
Formats	XML.
Data STORAGE and USABILITY	
Personally Identifiable Information	Web traffic, potentially accessing IPs of client hosts and/or unsecured payloads.
Sensitive data	N/A. No sensitive data is stored.
Data Retention	No data is retained in the xNF.

Data Encryption	Not applicable because there is no retention.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	Network packets are checked against the detection rules configured in the xNF for the detection of possible deviations or anomalies.
Sensitive data	N/A. No sensitive data is processed.
Data Processor	The platform operator can act as a data processor.
Data Controller	The platform service provider can act as a data controller.
Lawful Processing	The legal use of WTAs includes cybersecurity to ensure the detection of anomalies that may subvert the interests of service providers.
Data SHARING	
Transfer and sharing	The xNF only shares alerts and events with the TI pipeline, internally in the platform.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. No personal data is retained.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained.

10.2. GDPR compliance for SO

The SO, as one of the central elements in the architecture, finds itself in a crossroad where multiple data concur, whether being transmitted or internally processed. Table 10.2-1 provides an up-to-date assessment of the GDPR compliance for this subcomponent.

Table 10.2-1: GDPR compliance assessment for the SO subcomponent.

Component name	**Security Capabilities Orchestrator (SCO)**
Subcomponent name	Security Orchestrator (SO)
Description	Orchestrator for the SCs, handling its lifecycle (start/stop/deletion) and actions (day2), as well as an overseeing tool for the infrastructure visualisation, management and monitoring.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	Configuration of the system: ● Infrastructure-related data: nodes, Kubernetes, etc. ● Tenancy: organisation, email and; optionally, representative's data. ● Third-party software: IPs to OSM, Kafka, etc. Details on SCs: ● Private key to inject in running SCs.
Inputs (interfaces exposed to other PALANTIR components)	Details on SCs: ● Packages for SCs (via SCC).
Inputs (connectors to infrastructures and third-party software)	Runtime information: ● Running SC instances (via OSM and Kubernetes). ● Telemetry data from nodes (Kubernetes). ● Telemetry data from SC instances (Prometheus, others).
Outputs	Summaries of all data ingested above.
Formats	Plain text (JSON, YAML, files) and SC packages (.tar.gz files) for inputs. Plain text (JSON and YAML) for outputs.
Data STORAGE and USABILITY	
Personally Identifiable Information	Configuration of the system (by operator): ● Infrastructure-related data: IPs from protected and managed hosts. ● Tenancy: email. ● Tenanty: name and surname (optional data). ● Third-party software: IPs from managed SW. Details on SCs (by operator): ● Private key to inject in running SCs for telemetry. Runtime information (by infrastructure and 3rd party SW): ● Running SC instances: IPs. ● Telemetry data from nodes: IPs.
Sensitive data	N/A. No sensitive data is stored.

Data Retention	Data is kept during the operation of the SCs to protect the infrastructure and, in general, during the operation of the tenant with the PALANTIR platform. No history is kept beyond the operation lifetime.
Data Encryption	No technique applied. Yet, the storage runs in a private scope that is not accessible externally.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Personally Identifiable Information	Same data as in "Data STORAGE and USABILITY".
Sensitive data	N/A. No sensitive data is processed.
Data Processor	The SO is usually processing the data it outputs. This data is sometimes retrieved from third-party software and sometimes received already post-processed by other PALANTIR components. The data is also output to other PALANTIR components, which will process it further. Also, when it comes to telemetry, the operator or service provider also acts as the data controller.
Data Controller	The operator should be the data controller, indicating which data to fill for the SO to properly function.
Lawful Processing	Consent by the operator and/or the service provider shall be obtained before using the PALANTIR platform, and the granularity of the provided data shall be decided during the configuration of the system. Also, the contract requires the minimum necessary information to operate, leaving others (such as Personally Identifiable Information) to be optionally input by the person interfacing with the platform.
Data SHARING	
Transfer and sharing	Data is transferred within other PALANTIR components and subcomponents, as well as with third-party software that is hosted along with the PALANTIR infrastructure.
Third-party disclosures	N/A. No APIs are established for use by third parties.
Cross-border data transfers	N/A. No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	Interacting users are not requested to introduce personal data for the SO to operate. Instead, such data is optional. As per IPs, each tenant shall only be returned data regarding the IPs they oversee and manage.

Access	APIs implemented for the retrieval of the information persisted both for the specific tenant (or organisation) requesting access, as well as other information on their managed and protected infrastructure and the running SCs in the latter.
Rectification	APIs implemented for the modification of the information persisted for the processing of the data on each tenant (or organisation) and its managed and protected infrastructure.
Erasure	APIs implemented for the deletion of some or all information persisted for the processing of the data on each tenant (or organisation), its managed and protected infrastructure and the running SCs.
Notification	The requested email per tenant (i.e. organisation) allows the notification to reach them if there was a data breach on the persisted and/or processed data.
Restriction of processing	The APIs used to rectify the data can be used to fully restrict (disable) the processing (while keeping the stored data), by disabling the tenant or organisation itself.
Portability	The APIs used to access the data already provide this data in typical machine-readable formats, being JSON by default.
Objection	N/A. Data is not processed in terms of public interest, legitimate interest or scientific, historical or statistical purposes; nor is marketing associated. Objection can be achieved anyway by (total) restriction of processing or by erasure of such stored data.

10.3. GDPR compliance for SCC

The SCC stores information on the SCs and interacts with multiple other components that require such data. Table 10.3-1 provides an up-to-date assessment of the GDPR compliance for this subcomponent.

Table 10.3-1: GDPR compliance assessment for the SCC subcomponent.

Component name	**Security Capabilities Orchestrator (SCO)**
Subcomponent name	Security Capabilities Catalogue (SCC)
Description	SCO sub-component in charge of secure registration and onboarding of SCs, access their metadata by the Dashboard, as well as providing search functions within the catalogue.
On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	N/A. The SCC interfaces are only interacted with by other PALANTIR components.
Inputs (interfaces exposed	Details on SCs:

to other PALANTIR components)	• Registration/deployment capabilities (via Portal). • SC health-check (via Portal). • Search capabilities (via Portal). • Registration metadata management (via Portal).
Inputs (connectors to infrastructures and third-party software)	Authorise incoming Portal requests (via Keycloak).
Outputs	Summaries of all data ingested above.
Formats	Multipart data for input (APPLICATION_JSON format / APPLICATION_OCTET for .tar.gz files). APPLICATION_JSON format for all outputs.
Data STORAGE and USABILITY	
Personally Identifiable Information	Registration flow data: • Developer email. • Developer Certificate (X.509 format). • Developer name (optionally).
Sensitive data	N/A. No sensitive data is stored.
Data Retention	Data is kept as long as the tenant is active. No history data is kept.
Data Encryption	No technique applied. Yet, the registration information is stored in a private scope that is not accessible externally.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Data Processor	SCC requests are originated by the user via portal as well as from other PALANTIR components. The data is output to the portal user and to other PALANTIR components.
Data Controller	The operator should be the data controller, indicating which data to fill for the SCC to properly function.
Lawful Processing	Consent by the operator and/or the service provider shall be obtained before using the PALANTIR platform, and the granularity of the provided data shall be decided during the configuration of the system. Also, the contract requires the minimum necessary information to operate, leaving others (such as Personally Identifiable Information) to be optionally input by the person interfacing with the platform.
Data SHARING	
Transfer and sharing	Data is transferred within other PALANTIR components and

	subcomponents.
Third-party disclosures	No APIs are established for use by third parties.
Cross-border data transfers	No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	Consent is provided during the registration of the future SCC users (i.e. SC developers).
Access	No API implemented. Personal data is used for internal matching of a given tenant and the SCs.
Rectification	No API is implemented. The original developer data is used throughout all operations.
Erasure	APIs implemented for the deletion of all information, occurring when all SCs are deleted from the SCC.
Notification	The requested email per SC developer allows the notification to reach them if there was a data breach on the persisted and/or processed data.
Restriction of processing	No API is implemented. The SCC functionality does not directly manage the user's representations in a way that its processing can be restricted.
Portability	The APIs used to access the data already provide this data in typical machine-readable formats, being JSON by default and providing also some content in OCTET format.
Objection	N/A. Data is not processed in terms of public interest, legitimate interest or scientific, historical or statistical purposes; nor is marketing associated. Objection can be achieved anyway by erasure of such stored data.

10.4. GDPR compliance for RAF

The RAF provides a standalone form for risk assessment. Table 10.4-1 provides an up-to-date assessment of the GDPR compliance for this component.

Table 10.4-1: GDPR compliance assessment for the RAF.

Component name	**Risk Analysis Framework (RAF)**
Description	The Risk Analysis Framework is a standalone component that allows SME's to calculate and analyse their risk in a simple manner. Input is given through a user-friendly user interface in the form of a questionnaire and the analysis engine is implemented as proposed by ENISA.

On data, interfaces and formats	
Inputs (interfaces exposed to the operator)	Questionnaire UI exposed to the operator.
Inputs (interfaces exposed to other PALANTIR components)	The RAF framework communicates with the (i) PALANTIR Portal and (ii) the RE. The first interface is the integrated iframe of the questionnaire that also serves as the interface that the SMEs will interact with. The second interface is used when the risk analysis is performed and the results are sent to the RE through HTTP calls in JSON format.
Inputs (connectors to infrastructures and third-party software)	N/A. No interaction with other infrastructures or third-party components.
Outputs	The output contains scores regarding the 4 risk profiles: ● Reputation ● Productivity ● Financial stability ● Legal and regulatory and also the list of the assets that need protection.
Formats	JSON.
Data STORAGE and USABILITY	
Personally Identifiable Information	N/A. No personal data is stored.
Sensitive data	N/A. No sensitive data is stored.
Data Retention	The analysis results are retained in the local database. Since no personal or sensitive data are stored, there is no specific retention period.
Data Encryption	No technique applied. Encryption is not needed since there is no personal data and the results stored in the local DB are only visible to RAF.
Pseudonymisation	No technique applied.
Anonymisation	No technique applied.
Data PROCESSING	
Data Processor	The input provided from the questionnaire is processed from the Risk Analysis Engine.
Data Controller	The operator should be the data controller, indicating which data to fill for the RAF to properly function.
Lawful Processing	Consent by the operator and/or the service provider shall be obtained

	before using the PALANTIR platform, and the granularity of the provided data shall be decided during the configuration of the system.
Data SHARING	
Transfer and sharing	Data is transferred within other PALANTIR components and subcomponents.
Third-party disclosures	N/A. No APIs are established for use by third parties.
Cross-border data transfers	N/A. No APIs are defined for cross-border data transfers and sharing.
Data Subject RIGHTS	
Consent	N/A. Interacting users are not requested to introduce personal data for the RAF to operate.
Access	N/A. No personal data is retained.
Rectification	N/A. No personal data is retained.
Erasure	N/A. No personal data is retained.
Notification	N/A. No personal data is retained.
Restriction of processing	N/A. No personal data is retained.
Portability	N/A. No personal data is retained.
Objection	N/A. No personal data is retained. Plus, data is not processed in terms of public interest, legitimate interest or scientific, historical or statistical purposes; nor is marketing associated.