



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)

D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release

Document Identification			
Status	Final	Due Date	31/12/2021
Version	1.0	Submission Date	30/12/2021

Related WP	WP4	Document Reference	1.0
Related Deliverable(s)	D3.1, D5.1, D3.2, D4.3, D5.2	Dissemination Level (*)	PU
Lead Participant	UBI	Lead Author	UBI
Contributors	UBI	Reviewers	DBC
	I2CAT		UMU

Keywords:

Portal, Security Dashboard, Service Matching, billing framework, Service level agreement, constraint programming, Threat sharing, User Interface, Billing framework, Constraint programming

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.

This document and its content are the property of the *PALANTIR* Consortium. All rights relevant to this document are determined by the applicable laws. Access to this document does not grant any right or license on the document or its contents. This document or its contents are not to be used or treated in any manner inconsistent with the rights or interests of the *PALANTIR* Consortium or the Partners detriment and are not to be disclosed externally without prior written consent from the *PALANTIR* Partners.

Each *PALANTIR* Partner may use this document in conformity with the *PALANTIR* Consortium Grant Agreement provisions.

(*) Dissemination level: **PU**: Public, fully open, e.g., web; **CO**: Confidential, restricted under conditions set out in Model Grant Agreement; **CI**: Classified, **Int** = Internal Working Document, information as referred to in Commission Decision 2001/844/EC.

Document name:		D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release			Page:	2 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

Document Information

List of Contributors	
Name	Partner
Stelios Tsarsitalidis	UBITECH
Maxime Compastié	i2CAT
Carolina Fernández	i2CAT

Document History			
Version	Date	Change editors	Changes
0.1	10/11/21	Stelios Tsarsitalidis	Initial Table of Contents
0.2	9/12/21	Stelios Tsarsitalidis	UBITECH first contribution
0.3	15/12/21	Maxime Compastié	i2CAT contribution
0.4	22/12/21	Stelios Tsarsitalidis	UBITECH second contribution
0.5	28/12/21	Stelios Tsarsitalidis	Changes after UMU review
0.6	29/12/21	Stelios Tsarsitalidis	Changes after INFILI review

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	UBITECH	22/12/2021
Quality manager	INFILI	...
Project Coordinator	DBC	...

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	3 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

Table of Contents

Document Information	3
Table of Contents	4
List of Tables.....	6
List of Figures	7
List of Acronyms.....	8
Executive Summary	9
1. Introduction	10
1.1. Objectives and goals of the deliverable	10
1.2. Relation to D2.1, D2.2 and other WPs.....	10
1.3. Specification Methodology	11
1.4. Structure of the document	11
2. Dashboard and Service Matching Design	13
2.1. Portal and Dashboard Design.....	13
2.1.1. PALANTIR User Roles.....	14
2.1.2. Portal Backend	15
2.1.3. Message Aggregation and Classification	15
2.1.4. User Management.....	16
2.1.5. REST API.....	16
2.1.6. Live Notifications.....	16
2.1.7. Dashboard Web App – Portal Frontend	16
2.1.8. Involvement in PALANTIR usage scenarios	17
2.2. Threat Sharing Design	17
2.3. Service Matching Design	19
2.3.1. Initial modelling of the constrain-satisfaction problem.....	19
2.3.2. Functional architecture	24
2.3.3. Involvement in PALANTIR Usage Scenarios.....	25
2.4. Billing Framework and SLA.....	26
2.4.1. Billing models	26
2.4.2. Involved PALANTIR components.....	27
2.4.3. SLA and billing extension for the Security Capability Catalogue (SCC)	28
2.4.4. Involvement in PALANTIR usage scenarios	30
3. Specifications	31
3.1. Dashboard and Portal Specifications	31
3.2. Threat Sharing Specifications	33
3.3. Service Matching Specifications.....	33
3.4. Billing and SLA Specifications.....	34
4. Implementation.....	36
4.1. PALANTIR Portal and Dashboard implementation	36
4.1.1. Portal Backend Implementation	37
4.1.2. Dashboard Web Application Implementation	37

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release			Page:	4 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0
				Status:	Final

4.1.3. Code availability.....	38
4.2. Service Matching implementation	38
4.2.1. Code availability and dependencies	38
4.2.2. Service matching execution.....	39
4.2.3. Used data formats for communication	39
4.2.4. Development Status.....	40
4.3. Billing and SLA implementation	40
4.4. Threat Sharing implementation.....	41
4.5. Dashboard - Actors' views.....	42
4.5.1. User management	45
4.5.2. SME Manager views	45
4.5.3. Network Operator views.....	48
4.5.4. Security Capabilities Developer views.....	50
5. Conclusions & future work	52
References	53
Annex A: PALANTIR functional workflows	55
SC registration & onboarding	55
Initial deployment	55
Event handling	56
Periodic attestation.....	56
Fault Management.....	57
Annex B: Use Case to PALANTIR user role mappings	58

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	5 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

List of Tables

<i>Table 1: Billing and SLA properties involved in the security capabilities entrees of the catalogue</i>	<i>29</i>
<i>Table 2: Technical specifications for Dashboard and Portal</i>	<i>31</i>
<i>Table 3: Rechnical specifications for Threat Sharing.....</i>	<i>33</i>
<i>Table 4: Technical specifications for Service Matching</i>	<i>33</i>
<i>Table 5: Technical specifications for Billing and SLA.....</i>	<i>34</i>
<i>Table 6: UC roles to PALANTIR role mapping</i>	<i>58</i>

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	6 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

List of Figures

<i>Figure 1: PALANTIR Portal architectural design, with respect to the whole platform.....</i>	<i>14</i>
<i>Figure 2: Threat sharing service architecture and interaction with related components.....</i>	<i>18</i>
<i>Figure 3: Service Matching and interacting components</i>	<i>25</i>
<i>Figure 4: Billing framework architecture overview.....</i>	<i>28</i>
<i>Figure 5: PALANTIR Portal and Dashboard technologies layout.</i>	<i>36</i>
<i>Figure 6: Screenshot of the Service Matching usage.....</i>	<i>39</i>
<i>Figure 7: PALANTIR Dashboard login Screenshot.....</i>	<i>43</i>
<i>Figure 8: Dashboard menu sidebar per user type, retracted (left) and expanded (right).....</i>	<i>44</i>
<i>Figure 9: PALANTIR Threats Dashboard UI Sample</i>	<i>46</i>
<i>Figure 10: Dashboard Security Incident Reports view (Network Operator)</i>	<i>49</i>
<i>Figure 11: SC Registration and Onboarding Workflow</i>	<i>55</i>
<i>Figure 12: Initial Deployment Workflow.....</i>	<i>56</i>
<i>Figure 13: Event Handling Workflow.....</i>	<i>56</i>
<i>Figure 14: Periodic Attestation Workflow</i>	<i>57</i>
<i>Figure 15: Fault Management Workflow.....</i>	<i>57</i>

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	7 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

List of Acronyms

Abbreviation / acronym	Description
CERT	Computer Emergency Response Team
C.P.	Constraint Programming
CPE	Consumer Premises Equipment
CSIRT	Computer Security Incident Response Team
CSP	Constraint Satisfaction Problem
CTI	Cyber-Threat Intelligence
Cybox	Cyber Observable eXpression
ETSI	European Telecommunications Standards Institute
GUI	Graphical User Interface
HTTPS	HyperText Transfer Protocol - Secure
IoC	Indicator of Compromise
ISP	Internet Service Provider
JSON	JavaScript Object Notation
OASIS	Organization for the Advancement of Structured Information Standards
RAF	Risk Analysis Framework (PALANTIR component)
REST	Representational State Transfer
SC	Security Capability
SCC	Security Capabilities Catalogue (SCO subcomponent)
SCHI	Security Capabilities Hosting Infrastructure
SCO	Security Capabilities Orchestrator (PALANTIR component)
SecaaS	Security as a Service
SLA	Service Level Agreement
SM	Service Matching (PALANTIR component)
SO	Security Orchestrator (SCO subcomponent)
STIX	Structured Threat Information eXpression
TAR	Trust, Attestations and Recovery (PALANTIR component)
TAXII	Trusted Automated Exchange of Intelligence Information
TI	Threat Intelligence (PALANTIR component)
UI	User Interface
VNF	Virtual Network Function
vCPE	Virtual Consumer Premises Equipment

Executive Summary

In this demonstrator, the architecture that was provided in D2.1 is elaborated upon, and the dashboard-related subcomponents are detailed. This report dives deeper into the following components or subcomponents from the PALANTIR architecture: (i) Central Portal and Security Dashboard User Interface (UI) containing mainly (a) cybersecurity and (b) accounting dashboard elements; (ii) User Management; (iii) Security Capability Matching; (iv) Billing and Service Level Agreement (SLA) ; and the (v) Threat Sharing mechanisms. This report contextualizes all aforementioned elements as PALANTIR components, accompanies the delivery of components' source code for the first iteration, and elaborates on design, specifications and implementation of each component.

This report proceeds to describe the overall design of the aforementioned PALANTIR components and provides some key ideas about them. It also outlines the requirements that pertain to these components, having taken both D2.1 and D2.2 into account. The technical specifications are then derived by all the gathered requirements for all the subcomponents. Finally, the implementation of all the aforementioned components is presented, by explaining both their implemented and yet-to-be-implemented aspects, including the logic that is followed and the technologies utilized to build them. This is the first iteration of the deliverable, and D4.3 will be the next. D4.3 will showcase the final versions of the implementation of the Portal and Security Dashboard and the Security Capability Matching, and will include the implementation of the Billing and SLA and Threat Sharing components.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	9 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

1. Introduction

1.1. Objectives and goals of the deliverable

This deliverable, “D4.1 PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release”, highlights the current work performed in tasks T4.1 and T4.3 based on the requirements and the design of each of the related subcomponents. The document provides their technical specs (obtained, iterated and updated from the general requirements agreed in D2.1 and the use-cases of D2.2), as well as the implementation details, including the specific dependencies, libraries and tools, either in use or expected to be in use in the following stages.

The primary audience of this document are all technical consortium members; *i.e.*, those involved in the implementation in addition to technical decision making, who participate either in WP4 and/or in the other related WPs (mainly WP3, WP5 and WP6), since the Portal/Dashboard is the visual gateway to the entire project. This report also provides an overview of the design and implementation, and can be of great use to any technical external reader involved in the overall cybersecurity environment, who would focus on visualization, management and sharing aspects of the platform.

This deliverable is the first iteration, out of two, and represents half the WP4 tasks. The next iteration of the deliverable is due March 2023, where the final design and implementation are to be provided, as well as other interesting technical details such as the finalized interactions with other components, the inter-component APIs, and any other interface that is used to enforce the functional workflows within the PALANTIR platform. The next iteration will also include the complete showcase of the final User Interface and the functionalities it will provide. Given the state of affairs at the time of this writing, many specifics should be considered subject to change after December 2021, in which the integration of all components will be the primary focus and will justify any adaptation needed to ensure the components' integrability with the rest of PALANTIR platform.

1.2. Relation to D2.1, D2.2 and other WPs

This document uses D2.1 as a starting point to dive into lower-level technical details about the Dashboard and related components and subcomponents. It draws further direction from D2.2 use-cases in order to provide the complete set of requirements, and further refines the original PALANTIR architecture into the design of each component and subcomponent. This work also leverages the latest version of the internally agreed workflows that span across different architectural components. The expected behaviour described in the aforementioned workflows is taken into account, and is expanded upon by the design and interactions described here. Secondly, the list of requirements derived from D2.1 and D2.2 are mapped to technical specifications in this deliverable, explicitly stating the need for a specific subcomponent and how it is impacted by the requirement (and sometimes use case), and defining a specification per requirement.

Furthermore, other WPs and the remaining tasks of WP4 are strongly related to the work outlined here:

- The work exposed here relates to T4.2 and T4.4, whose components interact with the portal. Data breaches and faults found by the work of T4.2 are meant to be sent to the portal and routed to the appropriate stakeholders. Moreover, the work of T4.4 ensures the integrity of the components outlined here, as it covers every deployed PALANTIR component. T4.4 is also meant to provide integrity status and tampering-related alerts to the appropriate stakeholders through the dashboard.
- WP3 relates strongly to WP4, as components and subcomponents of both WPs directly interact with one another. Components of WP3 account for the Security Capabilities (SC) themselves, the risk-based analysis that is enacted on inputs by a client by the Risk Analysis Framework (RAF), and the overall Security Capabilities Orchestrator (SCO) including both the Security Orchestration (SO) and the Security Capabilities Catalogue (SCC), as the SC lifecycle management is implemented by these components. The Dashboard acts as the main visualisation

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	10 of 58	
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

component for all the aforementioned, and Service Matching is a crucial intermediary that eases SC selection, based on the situation.

- Moreover, there is direct relation with the Threat Intelligence (TI) and all related subcomponents, undertaken by the WP5. WP5 defines the identified threats and allows for monitoring and management wherever human intervention is required by working in tandem with the Portal. Anomalies and threats are found and sent to the Portal in order to be shown in the cybersecurity dashboards, along with recommendations and remediation information.
- Finally, integration actions and uses-cases of WP6 help to qualitatively validate the functionality expected by the subcomponents outlined in this report, and the integration that will be enacted after December 2021 will also help complete the implementation of several functionalities.

1.3. Specification Methodology

The specification methodology consists of gathering the functional and non-functional requirements of the Portal-related components and the UI, with their priorities, from: (1) Use case-specific requirements derived from partners who will be using and validating the PALANTIR platform and (2) requirements that pertain to the UI and the related subcomponents. The latter requirements are elicited by the questionnaires, the internal requirements analysis, having taken laws and regulations into account, and have been published in D2.1.

Moreover, UI and overall Portal requirements are classified into the categories that represent the main functionalities of modules or submodules. Each requirement is then translated into detailed technical and functional specifications. The UI specifications are illustrated visually, with the “UI Designs” that show an overview of the UIs, as expected at the end of development and are created in Figma [1]. These designs are validated by both the PALANTIR Portal developers and then by the entire consortium. The state of the UI is evaluated, focussing on the integration of the PALANTIR platform and providing access to features of individual components on the Portal UI / Dashboard accordingly. The assets and layouts of the UI Designs are made so that they can be exported directly into the Portal implementation, once deemed ready by involved partners. UI elements and components in the implementation are prepared in advance, so that entire views can be moved from design into implementation quickly. After validation of specifications and Designs, the implementation moves forward, and main focus is given on integrating with the internal components of PALANTIR, and completing the underlying functionality accordingly. The final version of the UIs and the integration logic with other components are to be reported on deliverable “D4.3 - Dashboard, Reporting and Threat Sharing Platform - Second release”.

1.4. Structure of the document

The structure of the rest of the document in the following sections is explained here:

In **section 2**, the overall design of the dashboard is explained, based on the architecture of the project. Every subcomponent has its design described, including key ideas, kinds of data that is communicated or stored, and foreseen interfaces and communications channels with other PALANTIR components. Afterwards, any internal sub-module is also described in its own subsection, which includes the explicit goal, behaviour and relations in the internal interactions or workflows devised for the subcomponent. Finally, in the last subsection, further UI requirements are extracted from the use-cases, identifying the actor, priority and expected behaviour of the UI for each.

Section 3 gathers all the requirements from D2.1 and the use cases, and revisits their applicability within WP4. To do so, it maps the requirements to implementation specifications. This effectively refines more generic requirements into specific technical needs to be fulfilled, and is aimed towards a more clear and concise definition of the expected features of each subcomponent. There are also some cases where specifications are derived directly from the use case definitions, as are shown in D2.2.

Section 4 compiles the ensemble of technologies used in each module and submodule. Specifics on how a technology is to be used, its benefits and its adequacy to the expected environment are all provided,

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	11 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

each in dedicated subsection per submodule. It also contains characteristics and technologies expected to be included, as the project moves forward.

Section 5 is the conclusion and includes some remarks on what has been achieved and the next steps.

Finally, the **annexes** that are provided include the PALANTIR functional workflows, as they have been thought of by the consortium, as well as a mapping of use case roles into PALANTIR platform roles.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	12 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

2. Dashboard and Service Matching Design

This section outlines the high-level design of the PALANTIR Portal, as derived from requirements that were first introduced in D2.1, as well as discussion points raised by the consortium, up until the time of writing this deliverable. This section also details the architectural design choices for the Portal-related components, and the subcomponents or modules they are composed of. Specifically, the Dashboard is decomposed in terms of views, and the related requirements and architecture choices are outlined for each.

2.1. Portal and Dashboard Design

The PALANTIR Portal, in general, is a set of components delivered by T4.1, that provide visual access to features of various components the PALANTIR platform comprises. The main goals of the dashboard components are to enable access to different functionalities, depending on the user role, and to present real-time and historical data. The overall security status and statistics of the SME, along with real-time alerts and notifications, are all accessible through the Portal. In essence, the Dashboard itself is a Web Application, the “frontend” of the portal. This web application is served by the backend, while the quintessential security data and reports, as well as the real-time notifications stream become instantly available to the dashboard web app upon the proper authentication. Essentially the Portal backend supports the dashboard by (a) providing access to a temporarily stored security data meant for the platform’s main users, mainly incidents and analytics, (b) routing events, alerts and notifications to the appropriate users, and temporarily storing reports regarding recent events, and (c) allowing for the creation and authentication of users.

The Dashboard UI frontend, however, provides access to even more functionalities than the Portal Backend provides. The Portal backend, other than the functionalities already mentioned, is only considered with providing the user role and ID to other PALANTIR components upon request, so that a user can access the exposed functionality of other components through the Dashboard web application, according to their role. This is done, having taken the overall PALANTIR architecture into consideration, as well as:

- The fact that events and data are shared between components through data and event streams.
- The fact that interactions between the UI and other components are done through the REST API of each component.

As such, depending on the situation, the dashboard can provide more interactivity with the whole platform, and enable different functionalities by directly integrating with them, and avoiding further entities between them. For that, there only needs to be a reverse proxy server for a whole platform deployment, that enables access of the Dashboard UI to all the REST APIs.

The complete visual security management environment the PALANTIR dashboard provides is essentially completed by fusing multiple APIs, one per component, into one web-based GUI, namely the dashboard. In order to do this securely and in a performant way, two generic rules are followed; (a) the web app has clearly segregated functionalities as different views, so that interactions do not happen with too many components at once; (b) access to the exposed APIs of the platform components and the SecaaS is aggregated through, and controlled by, a Reverse Proxy server. The reverse proxy can block unwanted access from the outside of the platform, and allow access only to paths designated by each component, filtered by user role and id at the component API level. This overall design enables the creator of each component and SC to expose certain functionalities based on their design and targets, and any further integration steps that may be required for a new feature are only taken at the web application level. This also allows for the quicker implementation of changes in the Dashboard, and the easy access to other supplementary or niche UI elements that are implemented within other components or SCs.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	13 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

In Figure 1 shown below, a complete overview of the architectural design of the PALANTIR Portal is shown, complete with the main user types, basic subcomponents of the Portal backend, and the interactions between the Portal subcomponents and other PALANTIR components.

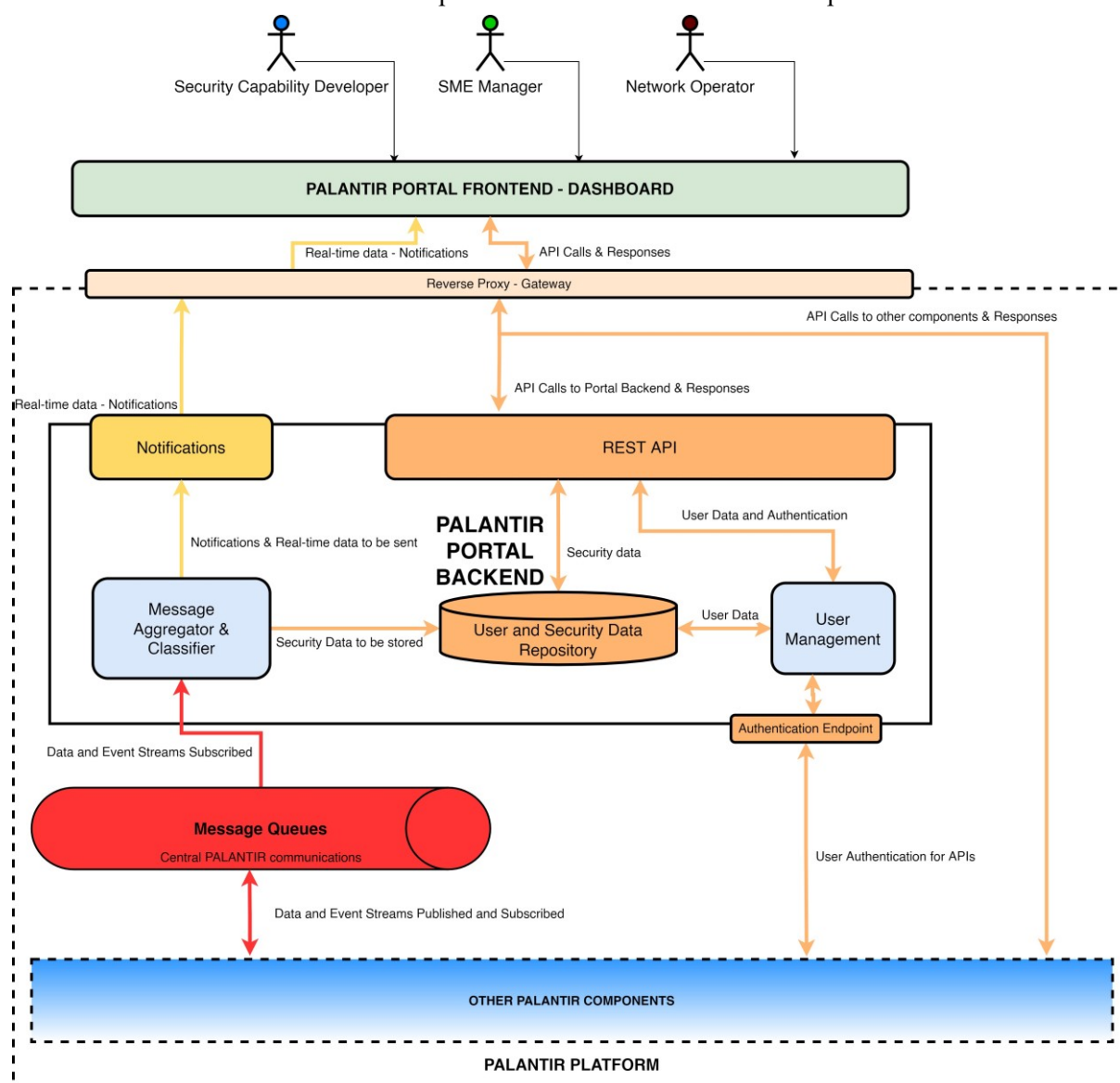


Figure 1: PALANTIR Portal architectural design, with respect to the whole platform

The following subsections dive deeper into the important aspects of components and subcomponents shown in Figure 1, the role of each in the PALANTIR platform, and high-level functionalities to be expected by the users.

2.1.1. PALANTIR User Roles

Through the process of analysing all the use cases of PALANTIR, and the cross-work-package discussions, the list of user roles has been refined. Two roles have arisen from the main users of the PALANTIR platform, as identified by each of the three use-cases, and three roles have arisen out of necessity, due to the architecture that the consortium has come up with for the platform, and having taken the DoA into consideration. It should be noted that it is possible that the definition of roles may change until the end of the project, either with variations of existing ones being introduced, or with slight

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	14 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

changes to the access rights and the responsibilities they bear. Any change that may arise, will be highlighted in deliverable D4.3.

The currently defined user roles are the following:

- **Network Operator:** The main users from the PALANTIR service provider organisation. Operators that either have IT support duties within the SME, or are generally given technical authority over the SME network, and can make some technical decisions that the PALANTIR platform leaves to human intervention. They have access to many views of the Dashboard, and can view many security details, and the overall status of the network. They also receive alerts and notifications and view real-time data. Access to relevant threats, propagated by threat sharing, is also given, and technical details of threats and indicators of compromise can also be viewed by this role.
- **SME Manager:** Executives from PALANTIR subscribers. The managers of the SME who are given access to the platform. The SME manager has access to various views of the platform. They can monitor alerts and notifications, can view real-time data, but are given fewer technical details than the Network Operator. They do, however, have access to more business-oriented functions, viewing financial projections, and setting privacy and threat sharing policies.
- **SC Developer:** This type of user is only meant to register SCs to the Catalogue, and monitor their overall performance, in terms of financial income and of activity when deployed by clients. This role is given access to only a specific set of views of the Dashboard that closely integrate with the SCC, and by extension to the SCO. Network Operators and SME Managers are, in a sense, customers/clients of this role.
- **Platform Administrator:** An executive from the PALANTIR service provider organization, who is also the initial user role, and the one that can register users of the platform. The Platform Administrator is only meant to access the user management subcomponent in order to define access of a role to certain parts of the platform and assign roles to users.
- **Integrity Assurance Operator:** An Operator that can only access and configure parameters of the Trust, Attestation and Recovery (TAR) framework. This operator can be defined by the administrator, and is restricted to only operate the TAR internals directly. This user role is **not** allowed in any view of the dashboard, and is only stored in the user management subcomponent for purposes of the TAR.

2.1.2. Portal Backend

The backend is meant to accommodate the immediate needs of the dashboard/frontend. User management is one major part of it, and access to its capabilities, as well as dashboard views is filtered by it. The Portal backend however is mostly concerned with aggregating and classifying the real-time data incoming by other PALANTIR components. This function essentially serves to route notifications to the correct users, as well as compute the latest reports. The overall security reports that are shown in the dashboard are prepared and stored in the backend so that the user can quickly view the current status of the network and the platform. The backend is complete with a database for the storage of all such data, and a connection to the message bus of the PALANTIR platform. The design of the backend is made in such a way, due to the overall platform design that has been agreed by the consortium. The following subsections proceed to show the design of each subcomponent of the portal backend, as seen in Figure 1.

2.1.3. Message Aggregation and Classification

A very crucial subcomponent of the backend is tasked with storing and feeding real-time data, based on the origin and the type of data, as well as the target users/tenants of the PALANTIR platform. The message aggregation and classification subcomponent, first and foremost, subscribes to various message queues, in which other PALANTIR components, such as the TI, SO, TAR and others, publish events and data to. Based on the origin, and the type of information in the feed, and the directly affected stakeholders, i.e., users of PALANTIR that are meant to be informed, this subcomponent sends the data

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	15 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

to the live notifications subcomponent, and stores security data in the database. The data is either stored as is, or is processed along with some other relevant stored data, perhaps even aggregated periodically, in order to have statistics and reports ready to be viewed by the users. The specifics of this stream processing are determined in coordination with the authors of the PALANTIR components that feed the data to the portal.

2.1.4. User Management

The user management component is where user authentication, creation and verification occurs. The Platform Administrator is the operator that is essentially given access to change the data stored by this subcomponent. User credentials are managed and user roles are assigned through it, and it also is responsible for the authentication of users who log in the platform. The authentication through the REST API of the Portal Backend is directly handled by this component, as well. For the rest of the components, the portal backend provides an authentication endpoint that simplifies access of logged-in users to other PALANTIR components. Based on the user data, any PALANTIR component can view the user's ID, role and their related enterprise/affiliation ID, in order to aid in its processes and ensure proper access control across the platform.

2.1.5. REST API

The REST API of the Portal Backend is meant for use by the Dashboard web app, and provides access to the security report data stored for the user's enterprise/affiliation and role. Because of this, most of its functionality is reserved for the Network Operator and SME Manager roles, while other parts of it are more generally available, such as user authentication. Historic incident reports, past alerts and notifications and basic statistics that are stored in the database are accessed through the API, and its definition depends on the needs of the Dashboard. Finally, any status update and any human intervention or action in the platform is recorded and validated through this subcomponent, in conjunction with the user management. The REST API provides an abstraction from the dashboard web app, as well as potential third party applications that may need to interact with the PALANTIR platform.

2.1.6. Live Notifications

The live notifications subcomponent handles active and continuous data stream connections with dashboard instances. Real-time data that has been ingested filtered and transformed by the message aggregator and classifier is sent as alert or push notification to any dashboard instance meant to receive it. This routing is constantly done in, as a reaction to events received through the message queues, and the messages are routed to the appropriate web-app instances based on the user ID, affiliation/enterprise ID and user role. A third-party app could also receive the live notifications in the same manner, once user authentication and data stream connection are properly made.

2.1.7. Dashboard Web App – Portal Frontend

The Dashboard is a web application that provides an appealing and intuitive user interface which exposes the security-related features of PALANTIR to end users. Based on the role attributed to each connected user, the graphical user interface adapts the provided features accordingly, in order to show what is of use to them. The different capabilities of the dashboard, which are also based on the role, are always visible, and the logged-in user can switch between them. By means of a web browser, PALANTIR users are able to see the monitoring system of PALANTIR as a whole, and perceive an overview of the security status of their network and services. Moreover, the dashboard interface also displays detected vulnerabilities, threats, and remediation suggestions that allow mitigation of each detected vulnerability, when human intervention is necessary. Network Operators are also able to analyse the proposed actions of a remediation and decide whether to apply the suggestion or not. Billing features are also present in the security dashboard graphical user interface of the SME Managers, allowing for viewing the financial status and setting up quotas. On another note, the Security Capability Developers are able to add new Security Capabilities by means of the registration and onboarding process, through the dashboard, as well as monitor their performance and the profits gained through

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	16 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

them, if monetized. SME Managers and SC Developers can also set SLA-related parameters for security capabilities, that also affect billing. Security Capabilities become available through a marketplace that is accessible by SME Managers and Network Operators, which is populated by SC Developers. Finally, Threats and Indicators of Compromise (IoC) can be shared between PALANTIR customers and CERTS and CSIRTS in the platform, with the help of the dashboard. SME Managers can set sharing policies, while everyone else with the appropriate access can inspect the published threats and IoCs. All the aforementioned are achieved through integration with multiple components of the platform, by receiving real-time data that is routed through the Portal Backend and by consuming the combination of the multiple APIs through the platform's gateway.

2.1.8. Involvement in PALANTIR usage scenarios

The PALANTIR dashboard is the main interaction focal point between the several users and the platform. Functionally speaking, it is involved in the following usage scenarios:

- *SC registration and on-boarding*: The SC Developer uses the portal to specify and register the capabilities he/she developed into the platform and make it available to the customers.
- *Initial deployment*: The Network Operator chooses to deploy security capabilities in response to a risk to mitigate. In return, he/she is notified when a SC has been instantiated, and, at a later stage, the SME Manager can inspect the billing status.
- *Event handling*: The Network Operator is alerted when a threat is detected by PALANTIR. Afterwards, the user is notified when a mitigation is available for deployment and confirms its application.
- *Periodic attestation*: The Network Operator is notified when a SC is failing its attestation and can select among the multiple remediation options to enforce. Later, the SME Manager can inspect the billing status to check if the SLA violation has impacted the billing.
- *Fault Management*: The Network Operator is warned when a SC is failing its internal health-check and is later prompted with multiple mitigation options. Later, the SME Manager can inspect the billing status to check if the SLA violation has impacted the billing.

2.2. Threat Sharing Design

Threat sharing is now separate from the Portal, as opposed to how it was viewed in D2.1, in the sense that there is a dedicated service that is concerned with it. The decision to turn this into a separate service, which integrates with the dashboard in a fashion similar to other components, is derived from the following considerations: First of all, threat sharing is meant to follow a specific set of standards, so that external parties can interface easily with it. This also means that the way threat and vulnerability data is received and stored follows a model that is not geared towards the specific PALANTIR dashboard visualisation, making the format here differ from the one in the Portal Backend. Secondly, the published threats, vulnerabilities and IoCs are meant to be represented in a way that is different from the ones that affect just the network of one client organization. The data stored and shared have more generic applicability and are not bound to the PALANTIR dashboard. Thirdly, the correlation between IoCs from different client organizations and their assets is a concern adjacent to the previous, and has to follow the same standards. In addition to all the aforementioned, the new separation of concerns makes sure that the Portal Backend is much more streamlined, while the Dashboard web app communicates with both the Portal Backend and the Threat Sharing components in the same manner. Since outsourcing of threat monitoring and sharing of Cyber-Threat Intelligence (CTI) are required, particularly in Use Cases two and three (as seen in D2.2), the related set of concerns should be handled separately from the rest, in tighter integration with the Threat Intelligence. The threat sharing component, therefore is concerned specifically with the following:

- Allowing for setting policies regarding the sharing and publishing of threats that have occurred in a per-organisation basis.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	17 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

- Sharing Threats, IoCs and even theoretical remediation schemes in a standard format with third parties, such as response teams that operate outside of the same PALANTIR deployment, whenever sharing is allowed and said data is available.
- Creating notifications about threats that happen to be relevant for similar PALANTIR client enterprises, or warning about potential vulnerabilities pre-emptively.

To achieve those goals, a service has been conceived, which is placed between the Threat Intelligence, the Portal Backend and the Dashboard itself, in the view of the entire PALANTIR platform. It facilitates the above concerns by receiving CTI prepared by the threat intelligence, storing and publishing it appropriately, while also warning other potentially affected stakeholders in PALANTIR, through the same real-time data streams flowing through the Portal backend to the Dashboard instances. The overall architecture and interactions of threat sharing are visually presented in Figure 2 below.

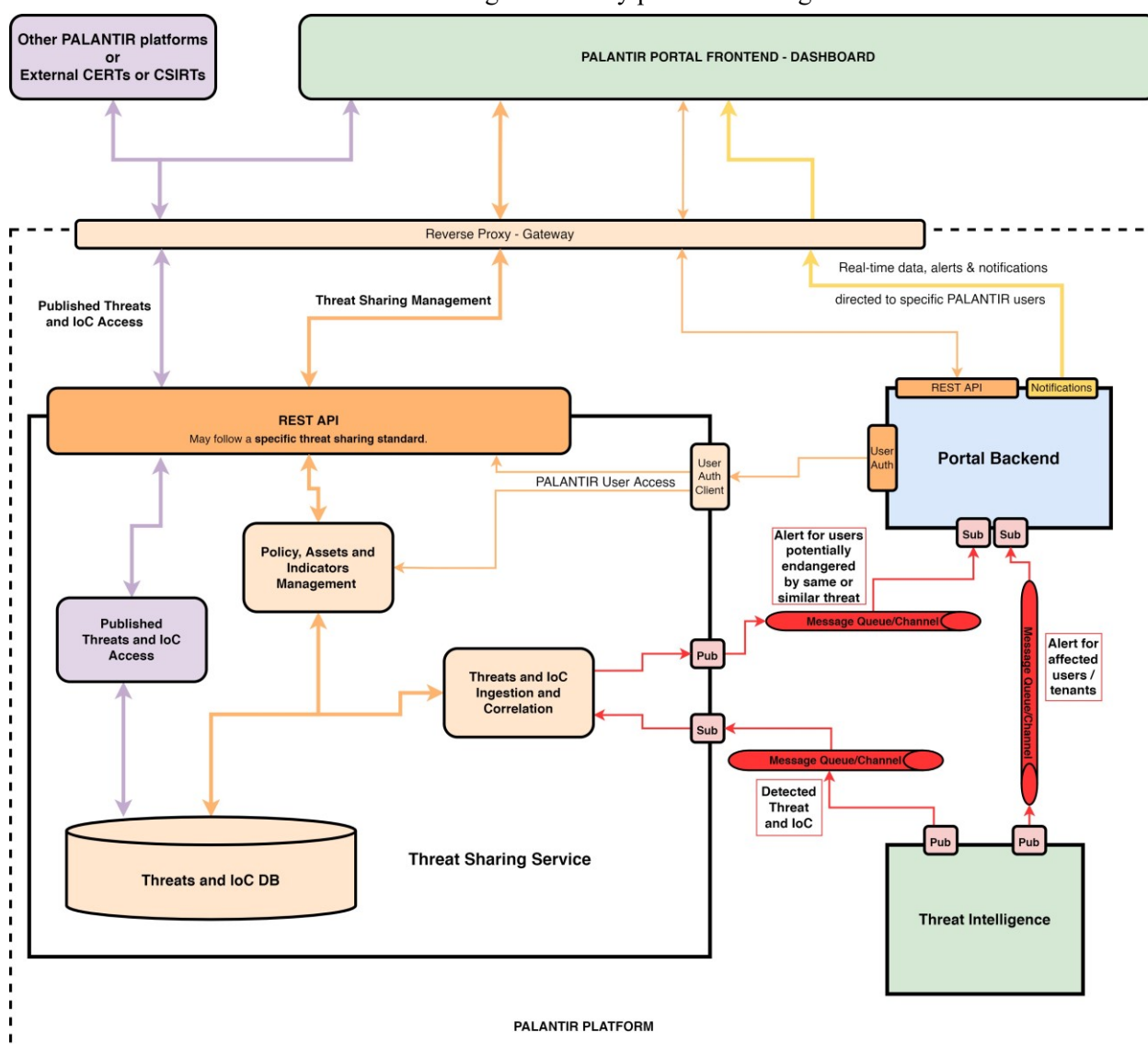


Figure 2: Threat sharing service architecture and interaction with related components

Once again, the integration of the components shown relies on the user authentication exposed by the portal backend to internal components, and the message bus of PALANTIR. The internal communications here are done through the message bus of PALANTIR entirely.

When TI makes a detection it sends it directly to the affected clients through the Portal backend, as seen before. However, the TI also sends a more processed and generically applicable finding, which is

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release	Page:	18 of 58
Reference:	1.0	Dissemination:	PU
Version:	1.0	Status:	Final

formatted based on the Threat Sharing standard to be followed, to the threat sharing component. A subcomponent of the Threat Sharing is tasked with (1) receiving these findings, (2) making correlations with known facts about client organizations and IoCs, (3) sending alert to other potentially vulnerable parties to be routed through the Portal, and (4) storing it in the Threats and IoC database.

The SME Manager is capable of setting policies about which CTI should be shared and when, and both they and the Network Operator are capable of adding more specifics about assets, based the standard that is followed, so that the correlation mechanism can operate. They can also provide or revoke access of third parties to the data stored in the Threats and IoC database. A subcomponent is meant to implement the overall management of threat sharing, based on the user's role.

Users of the PALANTIR platform, as well as allowed third parties, are able to view CTI through the threat sharing component, as a RESTful API that follows the threat sharing standard to be used is in place. This API is accessible by the Dashboard web app, which uses it to materialise its threat sharing views meant for SME Managers, Network Operators and SC Developers. Apart from the aforementioned management of threat sharing, all users can access a basic visualization of the CTI data in order to inspect it. In the meantime, the third-parties, such as other partner organizations, CERTs and CSIRTs, as well as other external PALANTIR deployments, can view the CTI data they have access to, by integrating with this API by simply following the same threat sharing standard. The access to shared threats and IoC also includes a client for other similar threat sharing components, as long as they use a compatible standard, possibly supporting a hub-and-spoke communication scheme between threat sharing components.

The overall architecture of the Threat Sharing component should be considered as possibly subject to changes in the future, as integration with the Threat Intelligence is crucial to its operation. However, this first draft is comprehensive enough to drive the implementation and integration activities of the project, after December 2021. The final version of this design will be shown in D4.3.

In terms of usage scenarios, the threat sharing is involved in the *event handling* usage scenarios, where the Network Operator is able to share the intelligence of the newly detected threat with other PALANTIR customers, based on the sharing policies set by the SME Manager.

2.3. Service Matching Design

The Service matching is a PALANTIR component delivered by T4.3 to determine relevant security appliances to be deployed to enforce a certain set of security properties with respect to constraints issued from the deployment environment. In this context, the requirements are structured by

- The security dashboard (after an interactive confirmation from the human operator),
- The recovery service (in case of breach mitigation or fault management),
- Or the risk management framework (in case of initial deployment),

and describe both the types of mechanisms to be deployed and their configuration. An approach based on constraint programming has been selected to drive the design and the implementation of this component. This supposes eliciting the SM features as a constraint satisfaction problem (CSP) addressable via C.P. and modelling as a set of equations indicating (i) a set of variables to solve with their interval definition (ii) a set of constraints restricting the set of solutions and (iii) the selection of an objective variable to identify the most optimal solution.

2.3.1. Initial modelling of the constraint-satisfaction problem

2.3.1.1. Overview

In this section, we detail our reasoning driving the elaboration of the CSP problem addressing the SM's objectives.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	19 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

2.3.1.2. Functional requirements on the problem

The SM must support four core features when selecting mechanisms to deploy:

- *REQ1*: The component must deploy mechanisms implementing the specified security capabilities.
- *REQ2*: The mechanism to be deployed must be compatible with the deployment model supported by the infrastructure.
- *REQ3*: The deployment of a security mechanism mustn't require more computing resources (namely CPU, RAM, and storage) than provided by infrastructure.
- *REQ4*: In case if the component identifies multiple deployment solutions, the component should select the cheapest one.

Multiple mechanisms may implement the same security capabilities but may also come with different billing models and infrastructure compatibility, making some of them more relevant than the other according to the (i) available infrastructure environments and (ii) pricing constraints expressed by the user.

The supported pricing model aligns with the three billing parameters envisioned in PALANTIR:

- *Fixed licence cost*: The first deployment of a mechanism results in the subscription of a general licence at a fixed cost. Further deployments won't appeal for further payment.
- *Fixed instantiation cost*: Each time a mechanism is scheduled for deployment, the fixed cost is charged.
- *Operation cost*: When a mechanism is deployed, a horary cost is charged all along its operation.

Therefore, to compute a reference cost to satisfy *REQ4*, it is not only necessary to rely on the cost parameter expressed as properties of mechanisms to be deployed, but also relies on a reference period to compute an operation cost. This implies the usage a reference duration, that should be specified by the user:

- *REQ5*: The user should specify a reference duration to enable the SM to compute a comparable cost from the horary cost specified by the component properties.

2.3.1.3. Modelling as a CSP

For convenience, we define B as the Boolean set, defined as follows:

$$B = \{0,1\}$$

We also define the set \mathcal{M} as the set of the n available deployment configuration for a security mechanism. Naturally:

$$\#(\mathcal{M}) = n$$

This set can be enumerated, as following:

$$\mathcal{M} = (m_i)_{0 \leq i < n}$$

To generalize this problem not only to one but to multiple security mechanisms, we consider

$$\mathcal{M}' = (m_i)_{0 \leq i < n'}$$

with $\#(\mathcal{M}') = n'$ and $\mathcal{M} \subseteq \mathcal{M}'$ where \mathcal{M}' represents all the deployment configurations of all security mechanisms.

To define a valid, constraint-satisfaction problem, we must define three main parameters [2], namely:

- A set of variables to be affected as a solution of the CSP,
- A set of interval domains for the variable needing affectation,
- A set of constraints that the affected variable should satisfy to declare a solution acceptable.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	20 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

As the core of the problem is to determine which deployment configuration should be enabled, we define the set of variables to be affected as:

$$X = (x_i)_{0 \leq i < n'} \quad (\text{EQ0})$$

Where x_i denotes if the deployment configuration m_i is to be enabled or kept disabled.

It is therefore trivial to infer the definition domain of those variables: They can either be affected by a positive value (1) to express a requested deployment, or (0) to express its denegation. We can formalize this assertion as following:

$$D = (D_i)_{0 \leq i < n'} = B^{n'} \quad (\text{EQ1})$$

We must now express the different requirements on the SM component as a set of constraints to be satisfied by each affectation of X , and construct accordingly a cost function to order all the valid affectations.

2.3.1.4. Infrastructure-issued constraints

REQ2 and REQ3 drive the selection of infrastructure on which a mechanism should be deployed. The first aspect we detail is related to the consumption of computing resources on an infrastructure, which is REQ3.

We model an infrastructure I_q belonging to a set of Infrastructures as a tuple of its computing characteristics and a set of booleans reflecting the supported deployment model. Currently, our infrastructure model accounts for the following characteristics:

- The number of CPU cores available,
- The amount RAM available,
- The persistent storage space available.

We also account for different deployment models for mechanisms. In real life scenarios, we want to discriminate the infrastructures from cloud computing, edge computing and on-premises location as their different technical natures will encourage the developers of the mechanisms to produce different versions of their component, adapted to each specific type of infrastructure.

Reasoning in terms of resource limitation permits us to implement an approach featuring the knapsack problem, known to be a NP-hard and representing a suiting case for constraints programming usage.

One straightforward but erroneous way to model infrastructure resource constraints consists of ensuring each components resource requirement do not exceed what an infrastructure can provide. Therefore, we may think that for blocking the allocation of a mechanism to an unsupported infrastructure type, we may affect its resource consumption to unlimited to ensure the constraint is systematically violated when the solver will consider such deployment. However, some infrastructures (such as public clouds) may be modelled with an unlimited number of resources as well, driving the solver to an unpredictable state when evaluating the constraints.

We overcome this pitfall by adopting a two-fold approach:

- We model infrastructure and components reflecting their resources offers and demands,
- We model their compliance and compatibility with the deployment models.

We can now write:

$$I_q = \langle r_q^{cpu}, r_q^{ram}, r_q^{stg} \rangle \in N^3 \quad (\text{EQ2})$$

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	21 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

Where:

- r_q^{cpu} stands for the number of CPUs provided by the infrastructure q
- r_q^{ram} for its amount of RAM,
- r_q^{stg} for its storage capacity,

Similarly, if for all $(m_i)_{0 \leq i < n}$, we can adopt the following notation:

- $r_{m_i,q}^{cpu}$ indicates the amount of CPU needed on infrastructure if deployment configuration m_i is enacted,
- $r_{m_i,q}^{ram}$ states the amount of RAM needed,
- $r_{m_i,q}^{stg}$ stands for the needed storage capacity,

These values can be null if the deployment configuration m_i do not operate infrastructure q. Therefore, if m_i is mono infrastructure, there is only one infrastructure I_q such as $r_{m_i,q}^{cpu}$, $r_{m_i,q}^{ram}$ and $r_{m_i,q}^{stg}$ are non-null.

Therefore, we can formalize a first set of constraints to assess which mechanisms can I_q support for deployments.

The first equations relate to resource consumption:

$$\begin{aligned} \sum_{0 \leq i < n} x_i r_{m_i,q}^{cpu} &< r_q^{cpu} \\ \sum_{0 \leq i < n} x_i r_{m_i,q}^{ram} &< r_q^{ram} \\ \sum_{0 \leq i < n} x_i r_{m_i,q}^{stg} &< r_q^{stg} \end{aligned} \quad (EQ3)$$

They can be read as the sum of resource requirements for one deployed mechanism have to be lesser than what the infrastructure can provide.

We can generalize (EQ3) to all security mechanisms by considering not the sole set of deployment configuration of a single mechanism but to all of them:

$$\begin{aligned} \sum_{0 \leq i < n'} x_i r_{m_i,q}^{cpu} &< r_q^{cpu} \\ \sum_{0 \leq i < n'} x_i r_{m_i,q}^{ram} &< r_q^{ram} \\ \sum_{0 \leq i < n'} x_i r_{m_i,q}^{stg} &< r_q^{stg} \end{aligned} \quad (EQ4)$$

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	22 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

This gives us the first set of constraints for a given infrastructure. They must be evaluated for every single infrastructure I_q considered for deployment.

To ensure the unicity of the deployment of a security mechanism and prevent multiple deployment configuration from being enacted multiple times on different infrastructure, we pose the following constraint:

$$\sum_{0 \leq i < n} x_i \leq 1 \quad (\text{EQ5})$$

$\sum_{0 \leq i < n} x_i$ is valued 0 if the security mechanism has not been retained for deployment, 1 otherwise.

In line with the PALANTIR requirements, we consider three deployment models:

- **Cloud**, addressing the deployment in a cloud region,
- **MEC**, focusing on the deployment of mechanisms in a mobile-edge environment,
- **vCPE**, supporting the deployment on virtualized CPE, located on the premise of a PALANTIR customer.

Each infrastructure supports one deployment model.

As some mechanisms may be supported by a subset of deployment mechanisms, they may not feature deployment strategies for all available infrastructures but only for the ones they are compatible with.

2.3.1.5. Security capabilities-issued constraints

REQ1 drives the constraints on the security capability to be deployed and the mechanisms that will support them.

To model how a configuration deployment of a security mechanism contributes to a security property to be accepted, we introduced a set of coefficients, where $p \in N$ denotes the number of security properties to be satisfied.

We can therefore introduce:

$$\left((c_{i,j})_{0 \leq i < n'} \right)_{0 \leq j < p}$$

with

$$c_{i,j} \in B$$

And for each of the p security properties to be satisfied, we identify the following constraint:

$$\begin{aligned} \sum_{0 \leq i < n'} x_i c_{i,1} &\geq 1 \\ &\vdots \\ \sum_{0 \leq i < n'} x_i c_{i,j} &\geq 1 \\ &\vdots \end{aligned}$$

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	23 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

$$\sum_{0 \leq i < n'} x_i c_{i,p} \geq 1 \quad (\text{EQ6})$$

This equation signifies that, for each one of the p properties to be satisfied, there must be at least one deployment configuration contributing to it.

2.3.1.6. Construction of a cost function

The construction of the cost function is framed by REQ4 and REQ5.

We want a cost function to minimize that reflect the cost of enacting a deployment of security mechanisms.

Let t be the reference period set by the use. if we denote

- $e_i^{license}$ the license for the i-th deployment configuration,
- $e_i^{instantiation}$ the instantiation cost,
- $e_i^{hourlycost}$ the hourly cost of exploitation,

The reference cost for a deployment configuration is:

$$e_{i,t} = e_i^{license} + e_i^{instantiation} + t e_i^{hourlycost} \quad (\text{EQ7})$$

Hence the proposed cost function:

$$C_t(\mathcal{X}) = \sum_{0 \leq i < n'} x_i e_{i,t} \quad (\text{EQ8})$$

Our cost approach is to select the solution providing the minimum value of the cost function.

2.3.2. Functional architecture

In this subsection, we detail and justify the design choices driving the implementation of the CSP as the SM components.

2.3.2.1. Overview

The SM will implement two main features:

- *FEAT1 Selection of security mechanisms*: From a list of security requirements, the SM will issue a list of security mechanisms to be deployed to match those requirements.
- *FEAT2 Requirement imputation*: The SM identifies which security requirement is responsible for the instantiation of a specified security policy.

In this section, we detail the SM technical framework, and explain how it supports the two features.

2.3.2.2. Technical Components

Figure 3 details the internal subcomponent of the SM, and the component interacting with it:

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	24 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

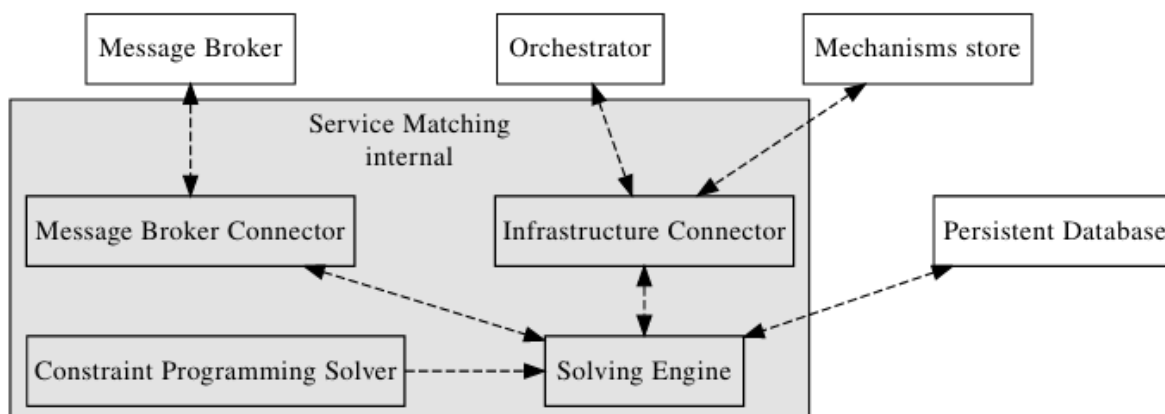


Figure 3: Service Matching and interacting components

The SM interacts with four external components, acting as dependencies:

1. *Message Broker*: This component communicates to the SM the security requirement to be resolved and return the result of the solving and the list of mechanisms to be instantiated and their pricing, and their characteristics. This component also receives requests for identifying security requirements from a mechanism name. We support Apache Kafka [3] to integrate with the rest of PALANTIR components (delivered by WP6).
2. *Orchestrator*: The orchestrator oversees collecting intelligence on infrastructures to receive mechanisms and manage the lifecycle of the mechanism's instances. This component is invoked by the SM. We envision the support the PALANTIR Security Orchestrator as a first orchestrator (delivered by WP3).
3. *Mechanism's store*: This component oversees listing all the security mechanisms available for deployment and their characteristics. In a first time, we plan to support PALANTIR Security Capabilities Catalogue Components (delivered by WP3).
4. *Persistent Database*: This component oversees registering (i) the ontology used for mechanism selection for FEAT1 and (ii) the security requirements involved in the selection of a component to enable FEAT2. We currently support PostgreSQL and MariaDB for this purpose, as suggested by WP6.

Internally, the SM have multiple subsystems dedicated to specific purposes:

1. *Message Broker Connector*: This subsystem permits to the SM to operate with the message broker. It is technically coupled with the broker it supports.
2. *Infrastructure Connector*: Similarly, the infrastructure connector is charged in interacting with orchestrator to retrieve the status and characteristic of the infrastructure.
3. *Solving Engine*: This subcomponent is tasked in populating the CSP model, solving it, and interpreting the solutions.
4. *Constraint Programming (C.P.) Solver*: This subcomponent provides the solving capability for CSP problems. We plan to integrate choco-solver [4], [5] to fulfil this role.

2.3.3. Involvement in PALANTIR Usage Scenarios

From a functional perspective, the SM interacts with PALANTIR components within the four following usage scenarios:

- *Initial deployment*: The SM is input with the risk management action provided by the risk management framework (RAF/RFM). Based on the specification of the deployment

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	25 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

environment provided by the security orchestrator (SCO/SO) and the list of available security capabilities provided by the Catalogue (SCO/SCC), the SM identifies a list of security capabilities implementing the needed mitigation actions and send it to initiate the instantiation. The association between the mitigated risks and the loaded security capabilities is registered by the SM for future SLA imputation purpose. This corresponds to FEAT1.

- *Event handling*: The SM only intervenes to select and record the SC deployment validated by the user in the portal (security dashboard) and send the deployment configuration to the orchestrator (SCO/SO). This corresponds to FEAT1.
- *Periodic attestation*: The security matching intervenes at two levels:
 - During the enactment of a remediation solution, the recovery service (TAR/RS) generates a remediation playbook possibly containing the instantiation of new security capabilities. In this case, based on the information provided by the catalogue (SCO/SCC) and the orchestrator (SCO/SO), the SM pinpoints the precise security capabilities to enact and output their list to the orchestrator. This is implemented by FEAT1.
 - During the integrity check (resp. after the SC deployment by the orchestrator), the SM is input with the list of failing (resp. restored) security capabilities to identifies subscription is impacted. The output is used by the accounting dashboard backend to register a failing (resp. a restored) SLA event and impact the billing accordingly. This corresponds to FEAT2.
- *Fault Management*: The SM is involved in two different roles:
 - During the enactment of fault mitigation solution, the SM identifies which security capabilities should be instantiated to match the specification of the playbook emitted by the recovery service and send the result to the orchestrator. This matches FEAT1.
 - During the detection of a fault and the deployment of a mitigation capability, the SM identifies which subscription is impacted by failing/restored security capabilities and submits the list to the accounting dashboard. This corresponds to FEAT2.

The complete workflows formalizing the usage scenarios are provided in Annex A.

2.4. Billing Framework and SLA

The PALANTIR billing framework is a set of tooling distributed across multiple PALANTIR components whose main purposes are to ensure the financial follow-up and assessing the SLA respect of the security capabilities operated by the PALANTIR platform.

2.4.1. Billing models

The billing framework operates based on the (i) lifecycles and operation time of security capabilities and (ii) the service level agreement (SLA) specified by the developers of the security capabilities:

- **Lifecycle of the security capabilities**: PALANTIR follows the operation of security capabilities to enforce the *measure service* billing as promoted by the as-a-service approach adopted by cloud computing services.
 - *The initial deployment* of a security capability includes the purchase of a specific license for it, if specified by the developer. Therefore, each time a security capability is instantiated anew by a PALANTIR customer, specific fix billing is applied to the customer subscription.
 - *The start of a new security capability instance* may induce a fixed charge on the customer. This permits to incorporate specific fees for security capabilities expected to scale up to deliver a protection if the developer wishes to support a licensing model per instance.
 - *The nominal operation duration of a security capability*, enabling the developer to enforce a billing model proportional to the usage of its security capability. PALANTIR

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	26 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

can follow up the duration of operation of a security capability. In this case, the deallocation of a security capability by the orchestrator closes the billing window.

- **Service level agreement:** PALANTIR not only allows the retribution of developers for the usage of their software, but also wishes to compensate the customer if the deployed security capabilities do not provide the protective power they are meant to deliver. To that extent, the SC Developers need to specify service-level agreements to protect themselves against potential malfunction of their software and PALANTIR will communicate these clauses to potential customer during the subscription.
 - *A duration of tolerated downtime:* if the health-check of a security capability fails and is found unable to deliver the expected protection (e.g., bug occurrence, infrastructure downtime due to maintenance), the PALANTIR customer won't receive the service he/she has subscribed to, opening the way to a legal prejudice. PALANTIR enables the developer to specify tolerated downtime per year during which the billing is suspended but does not call for compensation.
 - *The compensation fees:* if the downtime of a security capability exceeds the tolerated period, PALANTIR permits the customer to automatically receive a refund proportionate the incriminated period.

With the following approach, PALANTIR can deliver four distinct billing models, by combining the different parameters:

- **Subscription-based licensing:** Fees are retrieved only when a security capability is deployed for the first by a customer.
- **Instantiation-based licensing:** Fees are charged each time a new security capability instance is deployed.
- **Operation-duration-based licensing:** Fees are continuously charged during all along the operation of the security capability.
- **Complex licensing:** Fees are charges at each instantiation of a security capabilities and all along their operation.

2.4.2. Involved PALANTIR components

The billing framework comprises the following components, each of them contributing to a specific feature of the framework:

- *Deployed security capabilities (from T3.1):* each security capability implements a health-check interface permitting an external agent to assess if an instance works as expected: each time the orchestrator contacts this interface, an internal logic implemented by the developer is triggered and issues a positive or negative feedback. This result is conveyed back to the security orchestrator through the *VeNf-Vnfm* interface, as detailed in deliverable D3.1.
- *Monitoring interface of the security orchestrator (from T3.2):* The security orchestrator oversees managing the lifecycle of the security capability instances and provides a monitoring interface to access their state. The monitoring framework plans to use these interfaces to assess (i) if a security capability is running, (ii) when it was started, (iii) stopped, and (iv) if they are running as specified by the SLA through health check. These interfaces will be programmed to report security capabilities usage to the accounting dashboard.
- *Attestation engine and fault management (from T4.2 and T4.4):* These two components serve to report alarms on the SLA: the attestation engine notifies the accounting dashboard about node failing their attestation, while the fault management periodically queries the security orchestrator to proceed with the health-check on instantiated security capabilities emits an alarm if one is found failing.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	27 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

- *Security capabilities catalogue (from T3.4 and T4.3)*: The security capability catalogue is delivered by T3.4, but T4.3 extended its specification to integrate billing and SLA specifications in the description model of stored security capabilities. These properties are queried by the billing framework to let the accounting dashboard the right operation fees and SLA compensation fees of each operated security capabilities. They are precisely described in the following subsection.
- *Service matching (from T4.3)*: FEAT2 of service matching (requirement imputation) contributes to the billing framework. It provides an abstraction between the technical status of a PALANTIR protection, and the billing agreed during the subscription. It serves to associate a security capability whose an alarm was raised with the security requirements subscribed by the customer containing the billing condition and the service level agreement.
- *Accounting dashboard (from T4.1)*: The accounting dashboard is the recipient of billing events affecting PALANTIR customer subscriptions and oversees the computation of the fees to charge to the customers and presents the results to the SME Manager through a dedicated interface in the portal.

Figure 4 exposes how the components composing the billing framework are expected to interact.

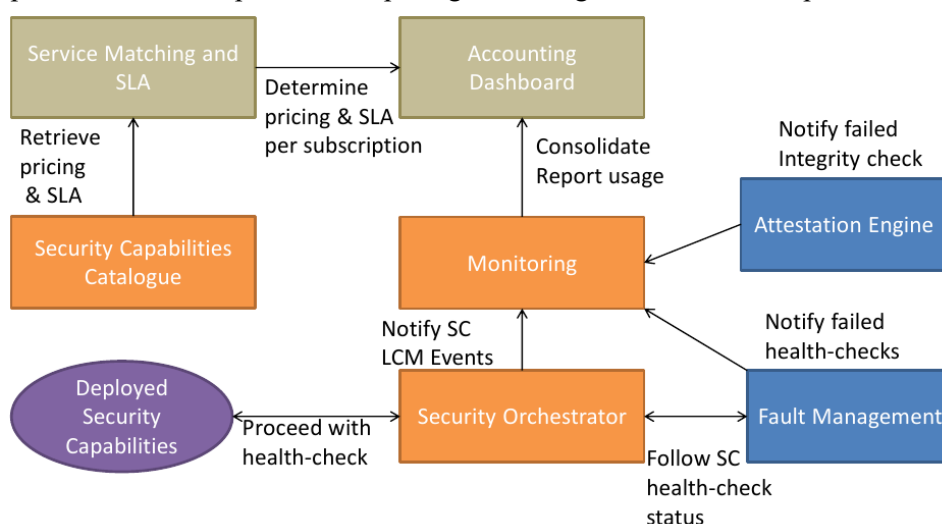


Figure 4: Billing framework architecture overview

2.4.3. SLA and billing extension for the Security Capability Catalogue (SCC)

Task T4.3 has contributed to the specification of the security capability storage model of the Security Capability Catalogue by imposing the presence of some SLA and billing-related properties in the definition of the storage model. These requirements were driven by the expected features of the billing framework components, including the service matching. Table 1 details the provided extension, seen below.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	28 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

Table 1: Billing and SLA properties involved in the security capabilities entrees of the catalogue

Name	Technical name	Purpose	Example	Component(s) using the property
Billing model	billing_model	Reference the supported billing models.	hourly	Service Matching and Accounting Dashboard
Subscription billing	subscription_billing	Price to access a SC at its subscription	20.0	Service Matching and Accounting Dashboard
Instance billing	instance_billing	Fees of an SC at its instantiation	1.0	Service Matching and Accounting Dashboard
Hourly cost billing	hourly_billing	Fees proportionate to the SC operation time	0.1	Service Matching and Accounting Dashboard
Tolerated SLA	sla	Amount of tolerated downtime per year	8h	Accounting Dashboard
SLA violation fees	sla_violation_fee	Discount on the non-tolerated downtime per hour	0.1	Accounting Dashboard
Supported deployment models	deployment_model	Supported PALANTIR deployment models	Cloud/MEC/vCPE	Service Matching
Detection Method	detection_method	Detection method of the protection method implemented by the SC	network_flow_monitoring	Service Matching
Mitigation Method	mitigation_method	Mitigation method of the protection method implemented by the SC	network_flow_filtering	Service Matching
Number of CPU needed for SC operation	nbCpu	Used to determine where to place a SC	2	Service Matching
Number of RAM (in MB) needed for SC operation	amountRam	Used to determine where to place a SC	256	Service Matching
Number of disk space (in MB) needed for SC operation	amountStg	Used to determine where to place a SC	1024	Service Matching

2.4.4. Involvement in PALANTIR usage scenarios

The billing framework is involved in four of the PALANTIR usage scenarios, and contributes to the following aspects:

- *SC registration & on-boarding*: The SCC collects the billing fees and SLA specification from the input provided by developer during the registration and store them in its database.
- *Initial deployment*: The service matching provides a quote and, for the mitigation of risk selected by the customer and registered the mitigated risks, the billing info, and the relevant security capabilities in its database. Once the SC is instantiated, the SO triggers health-check to assess its compliance with SLA. Eventually, the attestation engine notifies the monitoring framework the integrity status of the instantiating security capability, leading to the start of the billing and the registration of the billing event in the accounting dashboard.
- *Periodic attestation*: When the attestation engine identifies a security capability failing its attestation, it sends a notification to service matching to identify which subscription is impacted. The event on the subscription is then recorded in the accounting dashboard. When the service is restored, the security orchestrator notifies the service matching component to flag the impacted subscription as now fully recovered in the accounting dashboard.
- *Fault management*: When a failing health-check is detected by the fault management, it notifies the user through the security dashboard and notifies the service matching to flag the impacted subscription in the accounting dashboard. When the fault is mitigated and the impacted security is running back, the security orchestrator notifies the service matching to flag the impacted subscription is restored in the accounting dashboard.

The complete workflows formalizing the usage scenarios are provided in Annex A.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	30 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

3. Specifications

This section continues with mapping some of the PALANTIR requirements, provided in D2.1 and D2.2, as actual technical specifications that map to components related to the Dashboard.

3.1. Dashboard and Portal Specifications

The specifications of the Dashboard (Portal Frontend / UI) and the Portal Backend components have been derived by the relevant requirements in D2.1, as well as some aspects of the Use Cases 2 and 3, as seen in D2.2. Since PALANTIR will deliver a dashboard that is meant to integrate with various components, the relevant requirements are derived from multiple components, and, as such, the specifications also include the required integration with other components foreseen for each. The specifications are consolidated in the table below.

Table 2: Technical specifications for Dashboard and Portal

Req. ID	Requirement description	Spec. ID	Specification description
R1.1.1	The platform MUST provide registration and sign-in functionalities for the following roles: users, administrators.	DSB_S1	The portal backend must include a user-management component, which will allow for registration of users, and signing-in of administrators.
R1.1.2	The platform MUST provide a dashboard in order to present results of analysis.	DSB_S2	The dashboard must provide a Security Incidents view, and the security status and analysis results must be available in this and other views.
R1.3.18	The platform SHOULD provide streaming of resource utilization data for billing in the Dashboard.	DSB_S3	The dashboard should provide a visual overview of the resource utilization and the billing data, by integrating with the Billing and SLA component.
R1.3.28	The platform SHOULD provide an interactive workflow to review risks, statistics and security status of a SME.	DSB_S4	The dashboard should integrate with the RAF and TI in order to provide an interactive workflow to review risks, statistics and security status of a SME.
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) to mitigate the attack.	DSB_S5	The dashboard shall provide real-time alerts and notifications about attacks. The attack reports are provided by the TI.
		DSB_S6	The dashboard shall provide a “suggested actions” or “actions taken” view for each security incident. This is achieved by integrating with the TI.
R1.2.6	Security mechanisms used in a complex cybersecurity ecosystem SHALL be able to identify, distribute and allocate responsibilities between 5G ecosystem stakeholders.	DSB_S7	The dashboard shall provide a form for SC registration, which will contain a mandatory field about the mitigation capabilities of the SC. This is achieved by integrating dashboard with the SCC.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release			Page:	31 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0
				Status:	Final

Req. ID	Requirement description	Spec. ID	Specification description
R1.3.3	The platform SHALL provide a variety of SecaaS packages on the Catalogue.	DSB_S8	The dashboard shall provide Security Capabilities Marketplace and Inventory views, by integrating with the SCC.
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.	DSB_S9	The dashboard shall provide a SC registration view for SC Developers, which will accept only software images that are ready to be used, along with any extra metadata required by the SO and other components of the platform. This is achieved by integrating mainly with the SCC, having taken the whole platform into consideration for the data model. The registered SCs are validated, and the validation status is also visually shown in the dashboard.
R1.3.26	The platform SHOULD deliver additional security capabilities in function of specific use cases tasks.		
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.	DSB_S10	The dashboard shall provide a SC inventory view, which shows the status of the deployed SCs. This is achieved by integrating with the SO and the SecaaS through the message queues.
R1.3.14	The platform SHALL be able to retrieve the basic status for the security capabilities instantiated or available (in the Catalogue).		
R1.2.1	PALANTIR providers host SHALL provide telemetry and other auditing information relevant to the security mechanisms of the system.	DSB_S11	The overall threat protection status available in the dashboard shall also include the current status of the PALANTIR components and the security mechanisms in place. This is achieved by integrating with the SCHI and TAR.
R1.1.4	The platform SHOULD implement communication between PALANTIR components with a lightweight message queue	DSB_12	The portal backend should subscribe to the appropriate topics of the message queues system, in order to provide real-time data to the dashboard.
- (UC2 & UC3)	Both use-cases decree that a detected threat SHOULD be shared with any other potentially vulnerable stakeholder or node immediately, based on the set sharing policy.	DSB_13	The dashboard should enable managers and operators to select what kind of threat information should be shared in case of threat detection, by integrating with the Threat Sharing component.
		DSB_14	A notification about a detected threat in one tenant should also be sent to other tenants that either have common assets, or are in similar situations, if the appropriate threat sharing policy is set.

3.2. Threat Sharing Specifications

The specifications of the Threat Sharing component have been derived by the relevant requirements in D2.1, as well as some aspects of the Use Cases 2 and 3, as seen in D2.2. They are consolidated in the table below.

Table 3: Technical specifications for Threat Sharing

Req. ID	Requirement description	Spec. ID	Specification description
R1.4.6	The platform SHOULD provide an AI-based solution to deliver services, and be shared across the plain field; however, the data-sharing must ensure anonymity.	TS_S1	Data should be shared, in integration with the TI, in an anonymous way between different users.
R1.5.3	The platform SHALL be able to automatically classify the type of anomaly/threat and to share the intelligence information in a standard format.	TS_S2	The threat sharing component shall use a standard format to share anomalies and threats. This is achieved through integration with TI.
		TS_S3	The threats and anomalies, their classification, and other attributes shall be retrieved by the threat sharing component in order to be shared, by integrating with the TI.
- (UC2)	Use case 2 requires that threats SHOULD be shared between PALANTIR stakeholders, based on company policy.	TS_S4	Different tenants or users of the PALANTIR platform should have instant access to propagating threats that may affect them. The TI should mark a threat with the parameters that indicate a discovered threat that may affect other tenants, and the threat along with its important details is sent to potentially vulnerable tenants.
- (UC3)	Use case 3 decrees that threats SHOULD be shared between different PALANTIR ISP nodes, in order to inform potential targets of propagating network attacks.		

3.3. Service Matching Specifications

The specification of the service matching has been driven by requirements derived from the deliverable D2.1. They are consolidated in the table below.

Table 4: Technical specifications for Service Matching

Req. ID	Requirement description	Spec. ID	Specification description
R1.1.4	The platform SHOULD implement communication between PALANTIR components with a lightweight message queue	SM_S1	The service matching will connect to a Kafka broker instance, as suggested by WP6, to receive security requirements to solve and emit its deployment solutions.
R1.2.7	The PALANTIR eco-system SHALL be able to publish security	SM_S2	When a deployment solution is found by the service matching, the specification of the
Document name:		D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release	
Reference:		1.0	Dissemination: PU
Version:		1.0	Status: Final
Page:		33 of 58	

Req. ID	Requirement description	Spec. ID	Specification description
	KPI measuring the compliance of stakeholder with their Security Level Commitments.		involved security capabilities is stored, including the billing information and the service level agreement. This approach permits later to compute KPI service compliance with the subscription, including SLA aspects for billing purpose.
R1.3.5	The security capabilities SHALL provide the privacy specifications that are shown to infrastructure administrators that ultimately deploy such services.	SM_S 3	The service matching implements a “Quote” function, solving a security capability deployment without effectively enforcing it. This permits the dashboard and catalogue to prompt the user with the characteristics of security capabilities to be deployed, including their cost, the SLA and the privacy aspects priorly to their effective enactment.

3.4. Billing and SLA Specifications

The billing framework has been specified in line with the requirements expressed in D2.2. The following table details those requirements and describes how they are considered in the billing framework.

Table 5: Technical specifications for Billing and SLA

Req. ID	Requirement description	Spec. ID	Specification description
R1.2.1	PALANTIR providers’ host SHALL provide telemetry and other auditing information relevant to the security mechanisms of the system.	BSLA_S1	The billing dashboard exploits the monitoring interface of the Security Orchestrator to gather event impacting the billing. This telemetry encompasses events on the security capability lifecycle and the internal status of the instances (though health-checks).
R1.2.7	The PALANTIR eco-system SHALL be able to publish security KPI measuring the compliance of stakeholder with their Security Level Commitments.	BSLA_S2	The service matching stores the specification of deployed security capabilities locally, including the billing information and the service level agreement. Later, this information is exploited by the accounting dashboard to compute both (i) the charges of a customer's subscription and (ii) the retribution due to SLA violation. These KPIs are exposed to the SME Manager to ease its understanding of PALANTIR charges.
R1.3.9	The platform SHOULD be able to monitor the deployed security capabilities and expose such data	BSLA_S3	The billing framework actively exploits the health-check interfaces of security capabilities, exposed by the security

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	34 of 58	
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

Req. ID	Requirement description	Spec. ID	Specification description
	through programming interfaces for other internal components.		orchestrator. They are used to detect any occurring fault.
R1.3.18	The platform SHOULD provide streaming of resource utilization data for billing in the Dashboard	BSLA_4	The usage information of security capabilities is transmitted to the accounting dashboard through a continuous set of events relating to their lifecycle and the degradation of their operations.

4. Implementation

This section covers the implementation details of the different subcomponents introduced in previous sections. Specific technologies and techniques are covered, whether already applied or planned to be applied to the components and subcomponents. The logic and technologies are all explained in this, based on the progress made in the project so far. The PALANTIR portal and dashboard is explained, covering the logic and technologies of the Portal Backend, as well as the frontend / Dashboard web application. The Service Matching component's implementation is also explained, covering the dependencies, logic and development status. The current status and activities done and planned for the Billing and SLA and Threat Sharing components' implementation are covered, having taken into consideration that they are meant to be implemented during the period between December 2021 and April 2023. Finally, the overall experience of the Dashboard UI is laid out, along with some samples of it, at the time of writing.

4.1. PALANTIR Portal and Dashboard implementation

This section describes the current or expected implementation details of the Portal and its different submodules. At a glance, the Portal is a web application, with the Frontend implementing the ensemble of UIs / Dashboards, and making them available to the users as web pages, and the Backend implementing the logic that encompasses communication with most components, storage of data viewed by the users, as well as users and roles management. Figure 5 presents how different technologies and services compose the PALANTIR Portal.

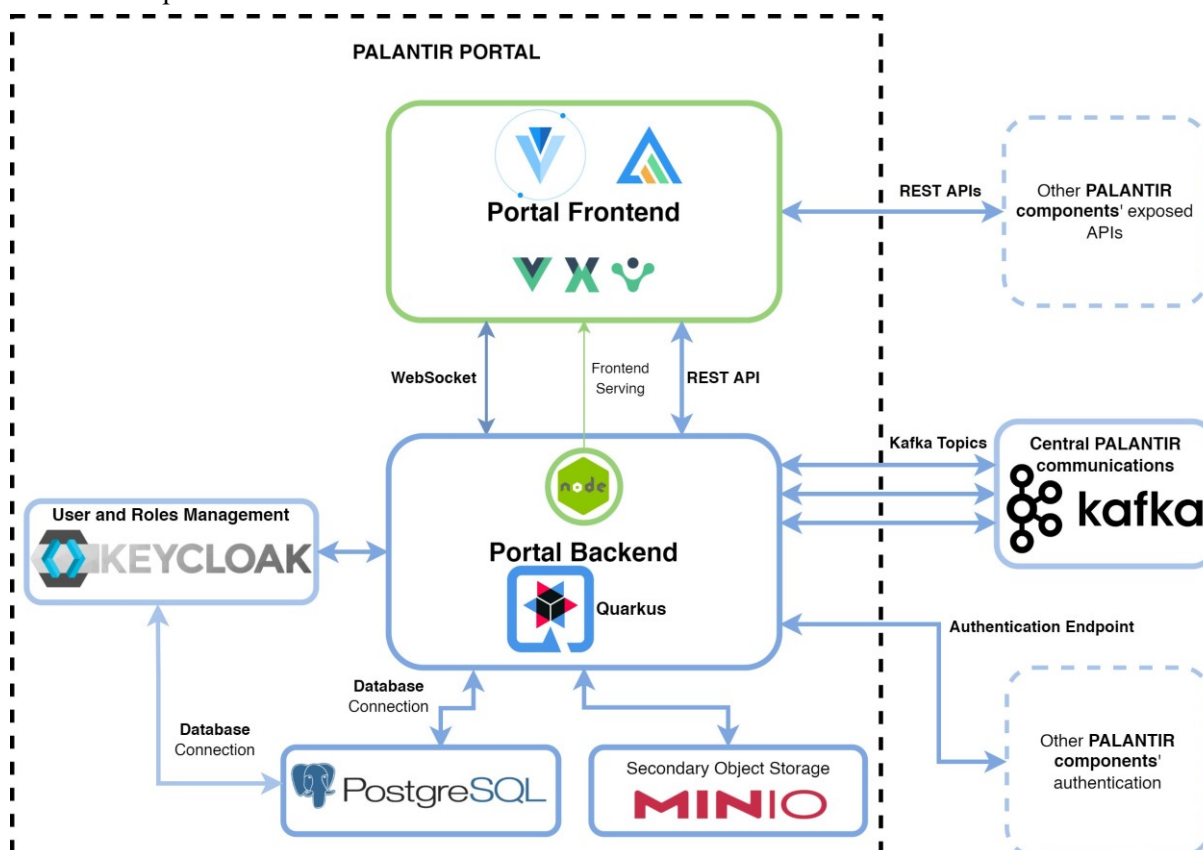


Figure 5: PALANTIR Portal and Dashboard technologies layout.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	36 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

The above view accounts for the implementation of the portal as an isolated project. In the following subsections, the implementation of the Dashboard web application and the Portal backend are laid out, in terms of code availability, dependencies and underlying logic.

4.1.1. Portal Backend Implementation

The portal backend is essentially a small collection of services, organised around the portal-backend service, which is implemented in Java 11 and based on the Quarkus 2.5 framework [6], as it is easily extensible, uses best-of-breed patterns, technologies and standards, and is lightweight. The main data storage for the portal-backend is done in a PostgreSQL 14.1 database [7], and the secondary object storage for binary data is done through the latest MinIO [8]. Connectivity with the datastores is achieved with the standard Quarkus extensions, using JDBC and Hibernate ORM [9] for PostgreSQL and a custom Java service implementation for input/output of binary data using the MinIO client. In order to facilitate connection with the PALANTIR data and event streams in Kafka [3], the backend uses the Kafka connector that is based on the Smallrye reactive messaging [10]. Keycloak [11] is used for user management as an external service, and it is connected to the portal backend with the Quarkus KeyCloak Admin Client [12], the usage of which is wrapped in custom service implementations. For the real-time alerts, the Quarkus WebSockets [13] extension is used, and an alert socket is created by each new client that connects.

In addition to the user authentication through the REST API, there is an endpoint that is usable only by other internal PALANTIR components that verifies a user is authenticated, what role the user has, so that other components can interoperate easily, even when they expose endpoints directed to the dashboard web application. The portal backend connects to the different Kafka topics that the platform is supposed to use, and has consumer handlers that use the methods of the message classifier service. The messages are processed in order to be stored, or further processed before storage, and are also sent to the connected users via the open WebSocket connections. The REST API is segregated based on the different concerns, mainly viewing data points or data sets, as well as some basic management options. In general, the overall implementation of the backend depends a lot on the modelling of relationships between types of users and data of interest, in message filtering from other components, the storage, and representation in the API.

4.1.2. Dashboard Web Application Implementation

The dashboard web application is based on Vue.js 2 [14], and served through a node.js 12 [15] server. The serving of this application, i.e., the node server, is coupled with the portal backend, which encloses the node.js server for the webapp in its environment. The webapp connects to the PALANTIR platform by authenticating through the user authentication endpoint of the backend REST API, which is exposed by the PALANTIR gateway.

The web application follows the material UI design [16], and as such, the Vuetify 2 [17] plugin is used, in order to create the style of the various reusable components of the UI. In order to make the entire application as a single page application with nested pages, the Vue router is used, along with the state management layer of Vuex [18] along with its shared mutations plugin [19], and transition animations between shared elements via v-shared-element [20]. Any charts in the application are created via the Vue ApexCharts extension [21]. State management, therefore, spans the entire web application, and visual components are prepared to be reused in different views. The communication with the platform is done via the Axios REST client [22] and the Socket.io WebSocket client [23].

In section 4.5 an overview of the current status of PALANTIR Dashboard UI is provided, as well as a description of the overall user that is currently planned by April 2023. Completing the functionalities of the dashboard will require integration with other PALANTIR components, which will be the focus during the period between December 2021 and April 2023.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	37 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

4.1.3. Code availability

The code of the implementation of both the portal backend and the dashboard web application can be found at [24]. The repository has everything required to run the dashboard in standalone mode, that is, without the rest of the PALANTIR components. There are detailed instructions on how to deploy and operate the portal and dashboard in the main README file. The project at a glance is laid out as a Java Enterprise project, however the dashboard web application is also included, in the path *src/main/dashboard-webapp*. In order to use both the main portal project and the frontend, the instructions for both should be followed in the order they appear in the README, first deploying the backend and then the frontend. All parameters necessary for any deployment are laid out in a *.env.example* file, which is used as a basis for a *.env* environment variables file.

4.2. Service Matching implementation

Up until December 2021, T4.3 has mainly consisted in:

- Conceiving the specification of the service matching based on the requirements expressed from the description of actions (DoA), in architecture deliverable, and from refinements identified during cross work packages collaboration.
- Elaborating the design of the service matching, specifically on aspects relating to FEAT1 (e.g., formulation of the constraint satisfaction problem aligning with PALANTIR operational issue in security capability selection).
- Developing and testing a first prototype featuring the selection of the security capability aligned with protection requirement (FEAT1) through the implementation of the related CSP and integration with the Kafka broker.

4.2.1. Code availability and dependencies

The code of the implementation of the service matching accompanying this deliverable can be found at [25] and can be built and operated by following the instructions from the main README file.

The prototype is implemented in Java language (version 11 of the OpenJDK) and exploits the SPRING framework version 2.5.6 because of its modularity and extensibility. This framework facilitated the integration with the Kafka broker through the Spring-Kafka plugin of version 2.7.8. The persistency for registering deployment solution with the associate security capability is permitted via the JDBC connector for relational database management system while the Jackson data bind layer enforces the data-access object paradigm. The prototype is conceived to run as a daemon.

As explained in subsection 2.3.2.2, FEAT1 of the service matching relies on constraint programming. Choco-solver framework 4.10.7 [4] served as the basis for implementing the constraint satisfaction problem. Its usage can be segregated in four different phases:

1. The creation of a model containing the different variables to be solved and their set of definition.
2. The configuration of the constraints limiting the solution space,
3. The solving phase, when solutions are being computed,
4. The access to the solutions and their interpretation.

Therefore, the logic of FEAT1 mainly consists of the following steps:

- The reception of a security requirements from the Kafka broker,
- The identification of the command and the requested processing,
- The preparation and solution of the constraint satisfaction problem,
- The interpretation of the result in the context of security capability deployment,
- The emission of the interpreted solution over the Kafka broker.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	38 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

4.2.2. Service matching execution

Figure 6 presents the execution of the SM:

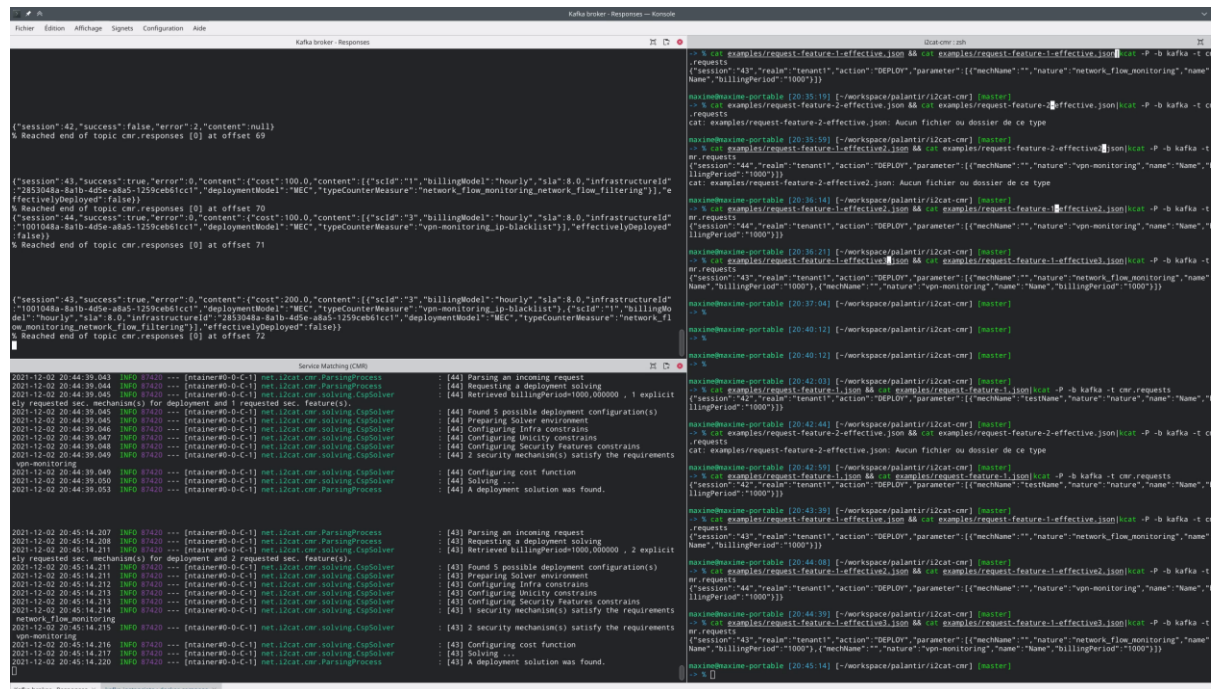


Figure 6: Screenshot of the Service Matching usage

In the screenshot above, the right column contains service matching commands we have manually injected in a local instance of a Kafka broker through the Kafkacat tool [26]. The messages are received and interpreted by the SM (bottom left console) and solved accordingly. The solutions are encoded and sent on a response topic, monitored in the top left console.

4.2.3. Used data formats for communication

The messages are structured as JSON strings on a `cmr.requests` broker topic. We detail the used grammar:

- *session*: A random numerical value to let the Service matching handle session with other components.
- *realm*: Identify the issuer of the request. Used for supporting multi-tenancy.
- *action*: Define which feature the Service Matching is called for. It can take the following values:
 - DEPLOY if the component is requested to determine a SC deployment plan from a risk to be mitigated with the effective deployment afterward (FEAT1)
 - SIMULDEPLOY if the component is requested to determine a SC deployment plan from a risk to be mitigated with no deployment afterward (i.e., a quotation) (FEAT1)
 - SECREQ if the component is invoked for identifying which SLA is supported by a specified SC (FEAT2).
- *parameter*: A list of objects containing:
 - a sole string field called `name` if FEAT2,
 - nature string, referring to the capabilities to be delivered by the security mechanism of FEAT1 (mitigation method or detection method),
 - `scId` string containing the id of a security capabilities if FEAT1,
 - `billing_period`, to define a reference period for computing the reference cost.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release			Page:	39 of 58	
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

Example:

```
{ "session": "43", "realm": "tenant1", "action": "DEPLOY", "parameter": [ { "mechName": "", "nature": "network_flow_monitoring", "scId": "", "billingPeriod": "1000" }, { "mechName": "", "nature": "vpn-monitoring", "scId": "", "billingPeriod": "1000" } ] }
```

The outgoing messages are emitted on the cmr.responses broker topic, and are structured as follows:

- *session*: The numerical value used previously
- *success*: a boolean indicating if solving has issued a solution or a security requirement result was found.
- *error*: a numerical value indicating that the Service Matching has encountered an error (0 means no error was encountered).
- *content*: a list of objects containing:
 - A value cost indicating the forecasted cost for SC deployment and operation (cost) if FEAT1
 - The list of the security capabilities to be deployed in FEAT1, including
 - Their id scId,
 - Their billing model,
 - The SLA,
 - The destination infrastructure id,
 - The supported deployment model,
 - The nature of the provided countermeasure,
 - The list of security requirements matching the security mechanism name if FEAT2
 - null if the success parameter is not true.

Example:

```
{ "session": 43, "success": true, "error": 0, "content": { "cost": 200.0, "content": [ { "scId": "1", "billingModel": "hourly", "sla": 8.0, "infrastructureId": "2853048a-8a1b-4d5e-a8a5-1259ceb61cc1", "deploymentModel": "MEC", "typeCounterMeasure": "network_flow_monitoring_network_flow_filtering" }, { "scId": "3", "billingModel": "hourly", "sla": 8.0, "infrastructureId": "1001048a-8a1b-4d5e-a8a5-1259ceb61cc1", "deploymentModel": "MEC", "typeCounterMeasure": "vpn-monitoring_ip-blacklist" } ], "effectivelyDeployed": false } }
```

4.2.4. Development Status

The final version of the Service Matching will implement FEAT1 and FEAT2 completely. Nonetheless, T4.3 has focused its effort on FEAT1 during the first 12 months of task T4.3 activity, as deemed as the most critical feature for leveraged PALANTIR test scenarios.

- Currently, FEAT2 is yet to be implemented although the interfaces and specification have been defined.
- FEAT1 is mostly complete but lacks (i) the registration of solutions into a database, (ii) the interaction with the Security Capabilities Catalogue to effectively retrieve the list and specification of effectively deployable security capabilities and (iii) the connection with the security orchestrator to retrieve the configuration of deployment environments and initiate the deployment of security capabilities.

The missing features will be developed and described in the second iteration of this deliverable.

4.3. Billing and SLA implementation

The activity of T4.3 has mainly consisted in eliciting precise specifications and architecture for the billing framework, and the core implementation activity is postponed to the second iteration of the deliverable. The main reason for this choice resides in its distributed nature across PALANTIR architecture and its dependences on the results to be provided by other tasks, namely:

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release			Page:	40 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0
				Status:	Final

- *T3.1 (Security and Data Privacy Services)*: This task will deliver security capability implementation for evaluation purpose and specify health-check interfaces and internal logic to proceed with health check, used for assessing the respect of the service level agreement. The technical specification and the implementation of the internal health-check is still an on-going effort.
- *T3.2 (Services Orchestration and Response)*: This task designs and implements the monitoring interfaces used to proceed with the follow-up of the instances of the security capabilities and emit notifications on evolutions in their lifecycle affecting the billing and the service-level agreement. The technical specification of these interfaces is still active.
- *T4.1 (Security Dashboard, Reporting and Threat Sharing)*: The user interface for inspecting the PALANTIR billing is to be designed and provided by this task. Now, the technical specification of this interface is still on-going.
- *T4.2 (Data Breach Notification & Fault Management)*: The fault-based management component oversees identifying the component failing their health-check and ceasing to respect their service-level agreement. The current status of this component is still under technical specification.
- *T4.4 (HW, SW and Configuration Attestation)*: The attestation engine assesses the integrity of operated security capability instances and notifies when they get tampered, impacting the PALANTIR service level agreement. The component is under implementation phases and the technical interface to be employed has been identified.

Task T4.3 has contributed to this effort by conducting design and implementation work of the service matching component, to be used for identifying impacted subscriptions by a degraded or recovered security capability (FEAT2). In the next iteration, task T4.3 will design and implement the back end of the accountability dashboard, in charge of the computation of the subscription fees to be charges on PALANTIR customers.

4.4. Threat Sharing implementation

Up until December 2021, the elicitation of the precise specifications and architecture were the only topics of focus given regarding the threat sharing. Since the activity of T4.1 has been focused on the Portal and the Dashboard, and due to the nature of the component, the threat sharing implementation is to be done in coordination with WP5 after December 2021, mostly as part of the integration activities. The complete threat sharing implementation will be available in deliverable D4.3. There are, however, some considered courses of action regarding the implementation at the time of writing this deliverable.

As seen in its architecture and specifications, the Threat Sharing component is drafted as a service that bridges the portal and the threat intelligence. Also, it certainly has to bridge different third-parties, like separate PALANTIR deployments as in use case 3 or CIRTs or CSIRTs, meaning that an open threat sharing standard has to be followed. Consequently, data regarding vulnerabilities, threats and incidents has to follow that standard, from the moment it is generated by the Threat Intelligence. That is required because the correlation mechanism that discovers potentially affected parties may need to operate on data that is a combination of data originating from outside the platform, as well as data from the TI components. This will also streamline the management of threat sharing policies as well as the storage of threat, vulnerability, mitigation and IoC data. An added bonus for such a standard would be to enable the sharing of best practices, strategic analysis results, and situational awareness, in order to better support decision making across different organisations, something that is encouraged [27]. Finally, the threat sharing standard should extend beyond just the data model, and provide a widely adopted or easily compatible, and, most of all, secure communication scheme. A threat sharing standard that is based on JSON, provides a clear and concise (in opposition of XML schemes) data model and schema, and provides the specifications of a RESTful API to be implemented is certainly preferable. It would also be even more preferable if communication models divergent from simple client-server can be enabled, such as hub-and-spoke and peer-to-peer through the threat sharing communications standard [28].

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	41 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

Several CTI-sharing standards exist, however there is no single de facto standard, or set of standards, widely adopted by organisations and by CSIRTs operating at multiple organisational, sectoral, national, and international levels. However, some standards have emerged that are more prominent and better supported than others. The ETSI technical committee for cybersecurity [29] states that the way forward in interoperability, and perhaps integration, will be through the OASIS CTI work [30]. There are also some studies that have evaluated various threat sharing solutions [31]–[35]. Although there are numerous available standards and protocols, the ones that stand out are (a) Structured Threat Information eXpression (STIX), (b) Trusted Automated Exchange of Intelligence Information (TAXII), and (c) Cyber Observable eXpression (CybOX). CybOX is a semantics schema that is used for formulating observations, and has definitions for classes like certificates, emails and even raw data such as headers has been integrated into STIX 2.0. STIX [36] is one of the two standards maintained by OASIS regarding CTI sharing. It is a standard for the serialization and interchange of CTI data, and is completely open-source. It includes a vast and inclusive set of definitions regarding security, such as vulnerabilities, indicators, attack patterns, intrusion sets, malware and malware analysis, attack tools, and courses of action for prevention or remediation. STIX has diverged from its rigid XML beginnings, and is now featuring an improved JSON data format, that allows for easy human and machine-readable exchange. STIX is also the only standard that appears to be at least loosely adopted, compared to others, due to its structure and usability. After comparisons conducted with other CTI-sharing schemes, it appears that STIX is currently by far the richest standard, it offers good reasoning capabilities in terms of semantics, and has a community that maintains a variety of open-source tools that help with modelling, interchange and validation. Even corporations have created CTI-sharing platforms that are based on STIX, such as Microsoft Interflow [37]. STIX is also complemented by other standards in terms of data structures, such as the Common Vulnerability Scoring System, the Common Platform Enumeration and Software Identification. It should be noted, however that STIX does require privacy measures in order to avoid critical information leakage, as is common with all CTI-sharing standards. The transport protocol for STIX is TAXII [38], another standard maintained by OASIS. TAXII enables the exchange of CTI over HTTPS, and is convenient because it defines a RESTful API that aligns with common sharing models. TAXII enables multiple communication models for secure communication, such as Hub and Spoke, Publish-Subscribe and Peer-to-Peer. In TAXII, servers are plumbing for CTI in STIX format, between clients, with each server having some channels that clients can publish or subscribe to.

Consequently, the starting point for efforts of the PALANTIR Threat Sharing implementation appear to be data modelling based on STIX 2.1 and communications based on TAXII 2.1, as they are the most widely adopted and reliable standards. The levels of adoption or deviations from these, if any, will be shown in the complete Threat Sharing implementation of the next iteration. The decision to use these standards as a starting point is also derived from the fact that there exist open-source projects that utilize them, such as the Medallion TAXII server [39] and TAXII client [40] implementations. There are also official tools for modelling, validation, and pattern matching for CTI based on STIX and TAXII [41], and even a visual modeller [42]. It should be noted, however, that the data to be shared is by no means private or sensitive. In order to achieve this, the consortium will agree on the specific subset of the standard that is to be used. Moreover, the users of the platform shall have the ability to dictate which threats are shared, and what data is to be included in the sharing.

It is foreseen that with integration with WP5 and the Dashboard web application, usage of standards such as the above will yield the necessary threat sharing standard by April 2023. Steps will be taken to follow the defined PALANTIR architecture and ensure that the CTI-sharing aspect of the platform is properly functioning.

4.5. Dashboard - Actors' views

This section proceeds to explain and showcase the views available to the users of PALANTIR through the dashboard. It should be noted that this is only a tentative set of views, and they are subject to changes until the next iteration. The final version of the Dashboard views will be available in deliverable D4.3.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	42 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

The Dashboard is laid out as a web application with a clean UI that houses various utilities. In this section the UI is viewed as a walkthrough of the experience from the perspective of the actors/users. All actors have to initially enter a valid PALANTIR Portal URL, and are then prompted for authentication through the login screen, as seen in Figure 7 below.

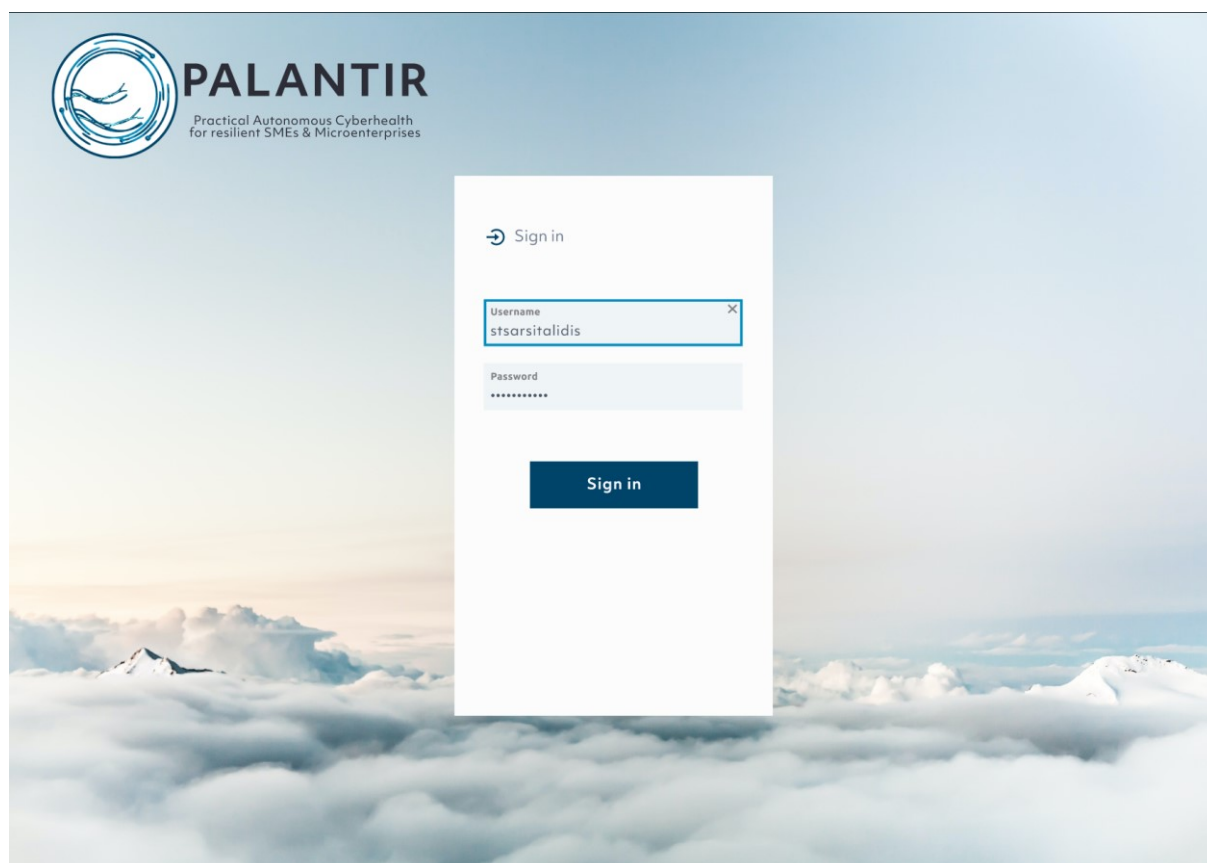


Figure 7: PALANTIR Dashboard login Screenshot

The user that is authenticated has to be already registered in the PALANTIR platform. The user is supposed to have a specific role within the platform, and thus, the authenticated is redirected to the landing page that pertains to their role. The user roles that are admissible through the login page are (a) the administrator, (b) the SME Manager, (c) the Network Operator, (d) the Security Capabilities Developer. The redirection of each user role is viewed below:

- The administrator is redirected to the UI of the User management component, which allows for registration of other users and the issuing of roles to the registered users.
- The SME Manager is redirected to the Threats Dashboard page, which is the main landing page of the dashboard and provides a brief overview of the state of the network and the platform. The menu sidebar provides access to most of the main functionalities of the dashboard, with an inclination towards more generic data visualization and business-oriented management.
- The Network Operator is also redirected to the Threats Dashboard page. The menu sidebar provides access to the majority of the technical capabilities of the platform, while also having access to visualization aspects.
- The Security Capabilities Developer is redirected to the Dev Dashboard, which is entirely different from the Threats Dashboard, as it shows data related to the developer's own security capabilities, while also providing a window into some public threats.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	43 of 58	
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

All the authenticated users that operate within the main dashboard (everyone apart from the administrator) get live notifications from the platform as they happen. The alerts and notifications are popups on the top of the screen, in any view they may currently be in, and pertain to assets or SCs they own. These pop-up notifications are colour-coded based on severity, with low severity or simple notifications being green, medium severity alerts being yellow, and critical or operation failure notifications being red. Unknown severity incidents, or generic notifications just have a grey background. If the notification pertains to an incident, it means it is kept in an incidents page, and thus the pop-up contains a link to it on that view/page. In general, any notification that pertains to an element that is viewable somewhere on the dashboard in more detail, or is managed or controlled through a view, then the notification contains appropriate links.

The overall generic layout of the dashboard includes a retractable sidebar on the left, which serves as a menu, and a few generic buttons on the top of each screen. The retractable sidebar on the right shows the user role on the bottom, and the list of views that are available, given the role. Clicking anywhere on the sidebar, other than the buttons, expands the sidebar in order to show the title of each view, near the button that serves as the link to the view. The menu sidebar for each role can be viewed in Figure 8, both in the normal and the expanded form. The top right corner of the screen shows the name of the user currently logged in, and their avatar, which also acts as a button in order to show the user-specific options and the button to log out. These elements are always visible in the main dashboard.

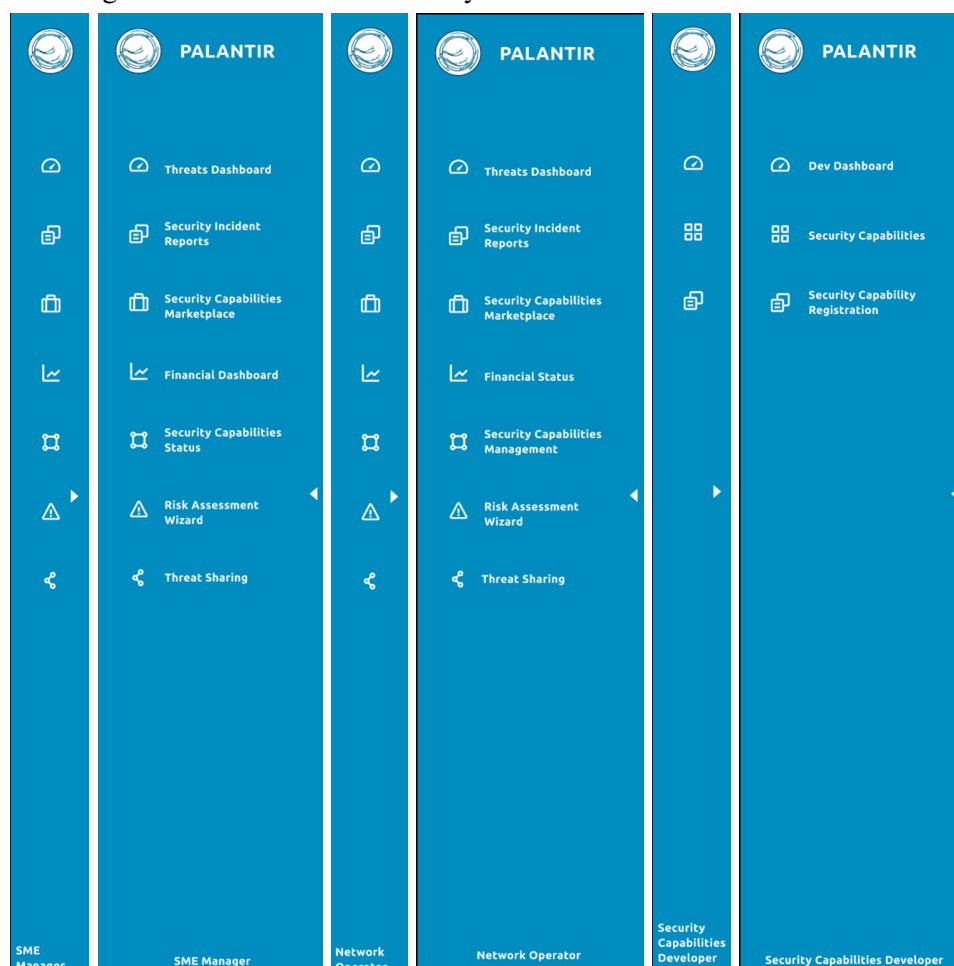


Figure 8: Dashboard menu sidebar per user type, retracted (left) and expanded (right).

In the following subsections, the experience of the different user types is followed in more detail, laying out the different available options and capabilities available to them.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	44 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

4.5.1. User management

The administrator is led directly to the user management component, which is the KeyCloak [11] UI, and not the main PALANTIR Dashboard. The platform is set in a way that the main platform roles are already defined, and the only steps the administrator needs to take are the creation of users and the appointment of roles, as seen through the role mappings. The administrator may not be allowed through the main dashboard, however is the only role that is trusted with introducing users through PALANTIR and even changing other security and authentication aspects. Since, as mentioned in the Portal backend implementation, section 4.1.1, KeyCloak is a service of its own, it can also span other aspects that extend beyond the PALANTIR components, depending on the deployment model and the network it resides in. The brief walkthrough in this subsection applies to the default parameters set in deployments of PALANTIR which are not further modified.

For the PALANTIR platform, there is one KeyCloak realm in which all user data resides called “PALANTIR” which is pre-set if KeyCloak is deployed using the default parameters. The administrator can appoint other administrators through the “Master” realm. In the case that the administrator only has to do with settings about PALANTIR, then the only pages of the PALANTIR realm they need to concern themselves with are:

- The Users Management console, which allows for creation, search, update and deletion of users. In the user profile tab, they can add attributes about the users manually if needed. The user role can be also set here, on a per-user basis. The point, however is to have only ONE user role per each user in PALANTIR platform, as the roles are clearly segregated in the dashboard.
- The Groups Management console, which allows for creation and management of groups. In the case of PALANTIR, the group should correspond to an organization, in the case of a centralized multi-tenant deployment, such as the cloud and edge one. Otherwise, the group is either disregarded, or should be set in a custom way, depending on the needs of the organization.

KeyCloak can potentially extend to authentication schemes and capabilities that extend beyond the scope of this project, but this also means that the organization or provider, depending on the deployment, can integrate the PALANTIR components into a much larger ecosystem of existent services. It should be noted, however, that if the deployment diverges from the defaults by a lot, the administrator may not be admissible through the PALANTIR Portal at all, and will have to access KeyCloak directly, and that will be impacted by the UCs and the work done in WP6 later.

4.5.2. SME Manager views

The SME Manager, once logged in, is redirected to the Threats Dashboard, and is given access to the following views list:

- Security Incident Reports
- Security Capabilities Marketplace
- Financial Dashboard
- Security Capabilities Status
- Risk Assessment Wizard
- Threat Sharing

Firstly, the Threats Dashboard provides a complete overview and status of the platform and the network. It includes visual elements that provide an overall assessment of the status, a latest threat findings classes, and top incident types. These elements are essentially small statistics views that give an idea of what is going on. Hovering over them highlights them and shows a popup with a few more details, and some might even give some links to specific incidents, or other related matters of interest. The Overall Status element, and the incident findings are not only threats or vulnerabilities recognised by the underlying platform, mostly the TI, but also the status of the Platform, as reported by the TAR. On the bottom of the Threats Dashboard is a graph of incidents over time, which aggregates incidents as the time passed. A sample of the Threats Dashboard is shown below in Figure 9.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	45 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

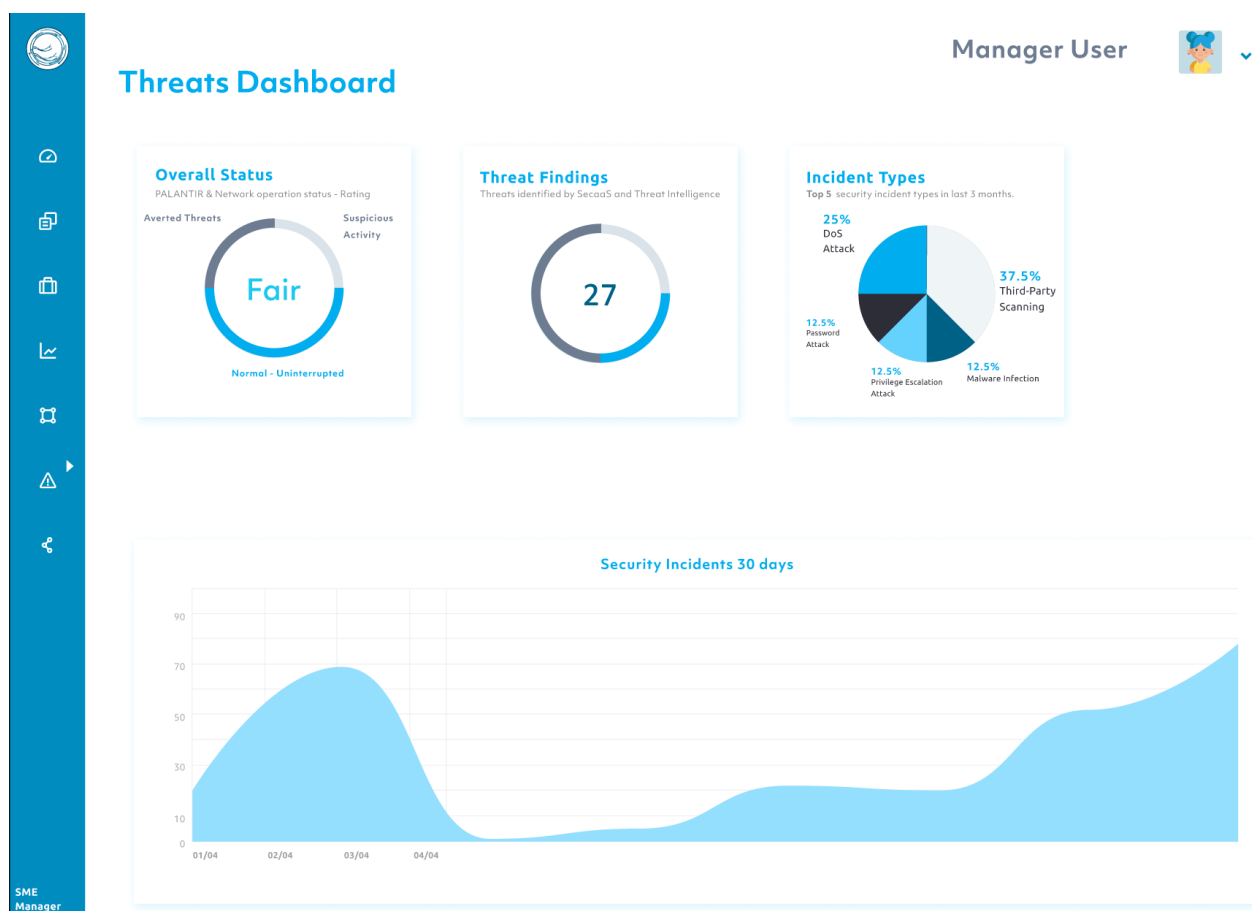


Figure 9: PALANTIR Threats Dashboard UI Sample

Secondly, the SME Manager has access to the past Security Incident reports, which is a list of the past incidents, with each entry exposing the incident type, health state, recommendation application state, and colour-coded severity. The colour-coding in the incidents view is (a) red for severe, (b) yellow for medium severity, (c) green for low to no severity, (d) grey for unknown or unclassifiable severity. The main tab of this view is called “results”, as there is another tab where special filters can be added, based on some parameters, called “applied filters”. The most basic filters, which are name, health status and date span, however, are always available on the top. The Security incident reports view is in many ways a companion view of the Threats Dashboard, and both are very dependent on the data stored in the portal backend. Incident reports also include any attestation incidents, which are most of the time coded as severe incidents. The user can view the latest incidents immediately, select one to view its details, and filter the ones they want using the basic filters on the top, or switching tab to a more fine-grained search using more filter options.

Another view available to the SME Manager role is the Security Capabilities Marketplace. This view is essentially the point in which search and purchase of available Security Capabilities can be done. The quick search is always available on the top of this view, where the user can enter some keywords in a text input, based on which a generic search is done on SCs and results are fetched upon clicking the filter button, or the refresh button next to it. By default, the results are visible as the main part of this view, as a tab called “results”. There is, however, another tab, called “applied filters” on which a more fine-grained search can be done based on more specific parameters. For instance, the user can select to add a filter, and then select the filter type, such as billing, privacy or security, the specific parameter, and a value. For example, a SC search filter may be of “security” type, be done on “threat protection” parameter, and a value like “ddos”, “injection” or “remote execution”. On the applied filters tab, a list of such filters can be applied, by adding different filters on available search options, and those filters are

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release	Page:	46 of 58
Reference:	1.0	Dissemination:	PU
Version:	1.0	Status:	Final

applied to every search done, when the apply filters button on the bottom is pressed. Once applied, the button on the bottom is greyed out, and the tab is switched to the results. The results tab shows a list of SCs that fit the given criteria. The list of SCs is sorted by relevance by default, which is determined by the Catalogue, the Service Matching and the Risk Assessment, and exposes the name, keywords as dictated by the Developer, price, details button and purchase button for each SC. The details show a visualization of some parameters about the SC, as fetched from the SCC. Basically, for the SME Manager, the software and integrity details are excluded and everything else about the SC is shown as a beautified visualization of the parameters based on the fetched JSON. In general, the SME Manager has a more business-oriented version of this view, and financial and security parameters are prioritized. In addition, the SCs that are suggested for purchase by the Network Operator appear first in the results when entering the view. When an SC is selected, near the purchase button there exists a marker about whether the subscription is within the current set limits by the Manager, or if it would exceed them, and by how much, accompanied by a bar that shows the difference. The Security Capabilities Marketplace is meant to be the place where the SME Manager can have the final say on which SCs the organization will be able to use in the network.

Another view the SME Manager has an important role in is the Financial Dashboard. The Financial Dashboard is the main point in which the user can view the overall economic status, as well as more fine-grained details regarding the currently subscribed offers, billing and SLA infringements. On the top of the view are three metrics which are always visible. These metrics are the current charges, the number of SLA incidents, and the number of subscribed offerings. The rest of the view is composed of the following tabs:

- The “status & limits” tab allows the user to set two kinds of limits on spendings; (a) monthly limit, which is the preferred maximum amount of money that the organization plans to spend for a month, and (b) the emergency limit, which is the foreseen amount of additional pending in case of a security emergency situation. Near the monthly limit is a bar which shows the current spending compared to the monthly spendings limit. When this tab is active, on the bottom there exists a graph of the spendings per time, with black-coloured points showing the aggregated monthly spendings, and the spending events as blue-coloured points, for the past 90 days. Hovering over a point shows the exact amount and motive.
- The “subscribed offers” tab shows a list of the offerings the organization has subscribed to, the motive, which shows the origins of the costs, and the costs they currently have aggregated for the current month.
- The “billing events” tab is also a list similar to subscribed offers, but it shows the date of each event, its nature, the specifics of the event, and the subscribed offering it is related to. The motive is an action that has been done through the platform, either automatic or manual. The subscription, deployment of SCs, fault detection and restoration, remediation of a threat, are all examples of the events shown in this tab.
- The “past invoices” tab shows the aggregated bills from previous months. For each period the charged items are shown and the overall cost.

The financial dashboard allows for financial monitoring, as well as setting quotas that help in the decision making for both the SME Manager and the Network Operator.

The SME Manager role has access to the Security Capabilities Status view in order to inspect the overall status and performance of the Security Capabilities the enterprise has subscribed to. On the top of the view are three main indicators: (a) The number of SCs currently in use, (b) the number of active SC instances within the network of the organization, and (c) the status of these SCs. The default status of the SCs is “OK”, when there has been an SLA violation recently it appears as “SLA Violation”, the “Breach” status appears when a breach has been found recently, and finally “SC Fault” when a SC is found to be faulty by the TAR. Below the three main indicators, there is a basic text search box that can be used to filter the list of SCs visible in the majority of the view, and a switch for the cost of SCs to be viewed for the “past month”, “past year” or “all time”. For each SC in the list, there is a row that shows its name, its id, number of instances currently running, and two buttons: “details” and “actions”. The

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	47 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

details that appear upon pressing it are their usage, in terms of latest activity, how many incidents they helped detect and/or avert through monitoring or remediation or impedance, and some information on the costs. Below these also exist some information about the SC that is the same as in the marketplace. The only action available to the SME Manager is the ability to either recommend a SC “to be removed or replaced”, or to mark it “to be retained”, in order to aid the decision making of the Network Operator. This particular view is meant only as a monitoring view for the SME Manager, with the only one action, and the details are once again business-oriented.

The Threat Sharing view for the SME Manager, as an early draft, is the point in which CTI from multiple environments can be accessed visually. Many specifics will depend on the selected CTI-sharing standard; however, the more generic stuff is laid out. The SME Manager will have access to three tabs for threat sharing, each with its own functionality:

- **Cyber-Threat Intelligence:** A view to browse shared threats, prioritized either by novelty or by relevance.
- **Internal Threats:** In this view, the SME Manager can view threats found in their organization’s network, and mark which ones are meant for sharing, and which ones are not.
- **CTI Sharing Policies:** Here the SME Manager can put policies in place, so that specific types of CTI data can be shared automatically, while others are not. There are also options for adding time limits for publishing and manual publishing of threats.

The way data are visualized in this view, and the kind of filtering on the sharable data, are all dependent on the CTI-sharing data model to be used.

The risk assessment wizard view is simply a view from the RAF imported in the dashboard web application. The SME Manager mostly answers to questions that pertain to the state of the organization as a whole through a questionnaire.

4.5.3. Network Operator views

The Network Operator, once logged in, is redirected to the Threats Dashboard, and is given access to the following views list:

- Security Incident Reports
- Security Capabilities Marketplace
- Financial Status
- Security Capabilities Management
- Risk Assessment Wizard
- Threat Sharing

In the Threats Dashboard, the Network Operator has the same view as the SME Manager, however the incidents can redirect to a Security Incident Reports view that is more detailed. The Network Operator may also have a more technical description for some types of incidents when hovering over the displayed data.

The Security Incident Reports view is almost identical to the one the SME Manager sees; however, they are able to view more details as well as remediation actions when selecting an incident. The details for the Network operator include the selection of recommended action, if human intervention is deemed necessary for an incident, as well as a few specifics on the incident and the recommendation in the format the TI emits. A sample of the main Security Incident Reports UI for the Network Operator is shown below in Figure 10.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	48 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

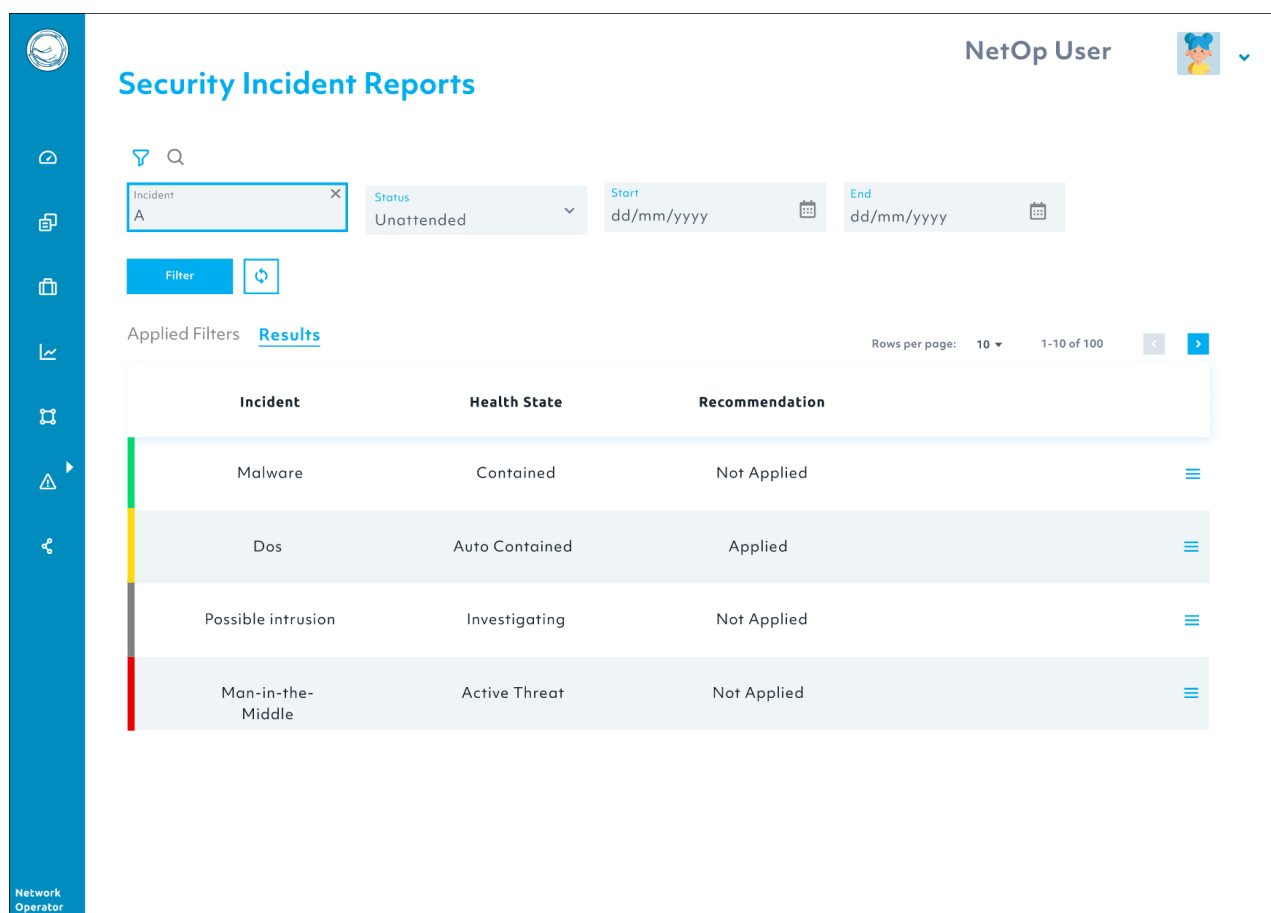


Figure 10: Dashboard Security Incident Reports view (Network Operator)

The Network operator can just select one of the incidents with no applied recommendation, and apply them as they see fit. Moreover, the “applied filters” tab allows for more parameters to be used for search or filtering, that are more technical and specific. The usability of this view is the same as with the SME Manager role, although the options and the shown incidents in each case may differ.

The SC Marketplace view for the Network Operator is largely the same as the one of the SME Manager, however with a few key different options. Firstly, all the SCs are accessible to the Network Operator, regardless of applicability to or compatibility with the network of the organization they represent or work for. The non-applicable SCs can be visible, and in the details the Network Operator can inspect the reasons why it is inapplicable, such as “unsupported deployment model”. For the applicable ones, the Network Operator can recommend them for purchase, and set a priority for the purchase, rather than purchase them directly. The other differences are about the visible and searchable parameters. On the applied filters tab, the Network Operator has access to more parameters, which include the “software” and “integrity” descriptors, which are prioritized along with the “security” descriptors. When selecting and SC to view its details, for the Network Operator, the software and integrity details are included, as opposed to the SME Manager’s version, and everything about the SC is shown either as JSON with the ability to hide or show the properties of objects for better navigation, or view it as raw JSON that can be copied. When an SC is selected, near the purchase recommendation button there exists a marker about whether the subscription is within the current set limits by the Manager, or if it would exceed them, and by how much.

The financial status view is almost identical to the SME Manager’s financial dashboard, but with a key difference; it does not allow for setting any values and only shows them. Other than that, it shows the same data about the organization, so that any SC purchase and subscription recommendations by the

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	49 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

Network Operator are further informed, and for the Network Operator to be able to justify the need for an increased budget devotion to the platform, if such a need exists.

An important view for the Network Operator is the Security Capabilities Management view. This view, in terms of layout, is the same as the Security Capabilities Status view of the SME Manager, but with more data visualized on the “details” of the SCs, and more available “actions” on the SCs. The details of a SC include more software and deployment information, in addition to everything else the SME Manager can view, with all data also available in raw JSON form that can be copied. The actions on a SC by the Network Operator are:

- Decommission a SC / unsubscribe: The ability to do so is enabled only if the SME Manager has not marked the SC as “to be retained”. Otherwise, the SC can be either decommissioned temporarily and can be easily restored, or can be permanently unsubscribed. For the subscription to be stopped, and the SC to be removed from the organization’s network, either the SME Manager has to have the SC marked for deletion already, or the SME Manager receives a notification for the SC subscription suspension and deletion. That notification lets the SME Manager have the final decision, unless the SC had been already marked.
- Set a priority number for each threat protection or inhibition: This is used for the automated responses to threats, in order to avoid potential overlaps of different SCs. The SC exposes the list of threats it helps avert, and the Network Operator can set a priority number, with higher meaning preferred, to each of them. For a given threat, the SC with the highest priority will be preferred by the underlying mechanisms of the PALANTIR platform.
- Access the CLI or GUI of the SC, if available: If the SC has its own interface, then access to it is given, either a link or as the necessary information needed to make that connection.
- Link SC with threats from threat sharing that can be averted: The SC is by default linked with threats, as a means to monitor or avert them. However, there may be new threats that are introduced in the threat sharing component, which have no known SC that can help mitigate a threat. If the Operator knows a SC that can be used for a new threat, then they can instruct the platform to use the selected SC for such a threat, and make this information available for threat sharing.

The Threat Sharing view for the Network Operator, as an early draft, is similar to the version of SME Manager. The differences are that (a) there is no access to the CTI Sharing Policies tab and (b) in the Internal Threats tab, the Operator has the ability to add and edit some CTI data that originate from the organization’s network. Once again, the Network Operator has a more technical point of view of Threat Sharing.

The risk assessment wizard for the Network Operator is largely the same as the SME Manager, with just a few differentiations on the questions and the included overall risk view. The Network Operator defines the assets on a more technical level through it. The dashboard only integrates the risk assessment wizard by including its own UI into it.

4.5.4. Security Capabilities Developer views

The Security Capabilities Developer, once logged in, is redirected to the Developer Dashboard, and is also given access to:

- Security Capabilities view, which shows details about their own SCs.
- Security Capabilities Registration, which allows for registration of new SCs into the PALANTIR platform.

The landing page for the SC Developer is the Dev Dashboard, an overview of the developer’s SCs’ performance by means of the SCO and Billing components, as well as opportunities for developing new SCs by means of Threat Sharing. As such there are two tabs in this view:

- SC Status: This is the default tab, a dashboard with a simple visualization of the performance of the SCs the developer has registered, both in operational and financial terms. On the top there are three main indicators about the developer’s SCs. On the right is the SC status indicator,

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	50 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

which expresses the current state as one of the following: “OK”, “SLA Violation”, “SC Fault”, “Breach”. If SC status indicator is not in the “OK” state, it provides details about the issues that have arisen, upon hovering. Next to it is an earnings indicator, with a switch below it, which shows the total earnings of the past month, and can be changed into past year and all time. The indicator on the top left shows the number of active subscriptions on the developer’s SCs. Below these, there is a graph which shows two lines, one being monthly earnings as time passes, and another showing the subscription number as time passes.

- CTI: This is almost the same as the Cyber-Threat Intelligence tab in the Threat Sharing views for the SME Manager and Network Operator. The only difference is that threats that have no SC matched to them are highlighted and prioritized in the list.

The goal of this dashboard view is to be a quick introduction into the state of affairs, for the SC Developer.

The PALANTIR platform is extensible due to the SC Developers, who create and then register new Security Capabilities. The SC Registration view facilitates exactly this, in close integration with the SCC. This view is a small wizard-like environment in which the Developer goes through stages, each being in its own tab. The stages / tabs for the SC registration are:

- Software / VNF description: Prompts for the connection with a software image repository, the address of the image, and the missing descriptors.
- Security description: A form containing SC Security descriptors.
- Billing description: A form containing billing and SLA parameters.
- Privacy description: A form containing privacy descriptors.

The SC Developer goes through these steps in order to register a SC in the Catalogue and onboard it in the SO. Once the registration is completed on the side of the developer’s UI, they are redirected to the Developer’s Security Capabilities view.

The Developer’s Security Capabilities view shows a list of all the SCs the developer has. At a glance the UI is, once again, similar to the SC status and management views of the previous roles. The SCs that appear are the ones the SC has registered, and the indicators pertain to those SCs. Another big difference is in the last three fields in each SC entry in the list. Instead of cost per SC, the developer sees earnings per SC. The “details” button shows all the parameters the SC Developer has registered, plus a few more regarding the deployment, in a popup. The “actions” button shows a popup with the following options:

- SC Update: Update the parameters or the software of the SC, through the SC Registration view.
- SC Interface Access: If such an option is available for the developer in the given SC, the user can access SC instances as a maintainer, in order to perform some on-line configuration changes when needed.

The Developer’s SCs view also shows the status of the listed SCs visually. When an SC is highlighted red, it means that there is an error or issue with it. If the left side of the SC entry is blinking, it means that it is still going into the onboarding process.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	51 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

5. Conclusions & future work

This deliverable provided the initial detailed layout of the architecture, low-level design and specifications of the components that are closely related to the dashboard, from WP4.

The information provided here is the current, up-to-date architecture and design for each component or subcomponent. This design covers everything from the internal separation of logic concerns, to the interactions between the modules, and provided the layout of the PALANTIR platform's top-level components. All of these specifications and designs pave the way for the development efforts within each of them. Future discussions on integrations with other components will be impacted by the work of this deliverable. This deliverable has also provided a view on the issues to be tackled during the integration, which span the entire project, as other components are combined with all the dashboard-related components in order to provide the complete PALANTIR user experience.

Besides the design, this deliverable gathers the low-level, development-related details in the form of technical specifications (refined from the general requirements introduced in D2.1 and the use cases of D2.2), some technical decisions, benefits and justifications about each subcomponent, as well as pointers to implementation. The implementation, both already done and planned, has been laid out in detail, demonstrating the technologies used and the target functionality, both in terms of logic and user experience.

As the project goes into its second phase, following December 2021, the next steps mainly have to do with fulfilling any requirements that are not yet fulfilled in the implementation, and refining the implementations of the components in order to integrate them. Some important next steps include the implementation of a complete billing dashboard, along with its backend, further integration with WP3, mainly the SO, as well as integration with the TI of WP5, the standardization and implementation of the threat-sharing component and dashboard, implementation of FEAT2 of the SM (SLA imputation), and fine-tuning the designs of the UIs and the data kept in the portal backend. Code existing in the repositories will continue to be developed and improved upon, while new repositories will be provided for the Billing and SLA, and the Threat Sharing components.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	52 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status: Final

References

- [1] 'Figma: the collaborative interface design tool.', *Figma*. <https://www.figma.com/> (accessed Dec. 23, 2021).
- [2] K. N. Brown and I. Miguel, 'Chapter 21 - Uncertainty and Change', in *Foundations of Artificial Intelligence*, vol. 2, F. Rossi, P. van Beek, and T. Walsh, Eds. Elsevier, 2006, pp. 731–760. doi: 10.1016/S1574-6526(06)80025-8.
- [3] 'Apache Kafka', *Apache Kafka*. <https://kafka.apache.org/> (accessed Dec. 23, 2021).
- [4] C. Prud'homme, J.-G. Fages, and X. Lorca, 'Choco Solver Documentation', p. 41.
- [5] 'Choco-solver', *Choco-solver*. <https://choco-solver.org/> (accessed Dec. 23, 2021).
- [6] 'Quarkus - Supersonic Subatomic Java'. <https://quarkus.io/> (accessed Dec. 23, 2021).
- [7] P. G. D. Group, 'PostgreSQL', *PostgreSQL*, Dec. 22, 2021. <https://www.postgresql.org/> (accessed Dec. 23, 2021).
- [8] M. Inc, 'MinIO | High Performance, Kubernetes Native Object Storage', *MinIO*. <https://min.io> (accessed Dec. 23, 2021).
- [9] 'Using Hibernate ORM and JPA'. <https://quarkus.io/guides/hibernate-orm> (accessed Dec. 23, 2021).
- [10] 'Apache Kafka Reference Guide'. <https://quarkus.io/guides/kafka> (accessed Dec. 23, 2021).
- [11] 'Keycloak'. <https://www.keycloak.org/> (accessed Dec. 23, 2021).
- [12] 'Maven Repository: io.quarkus » quarkus-keycloak-admin-client'. <https://mvnrepository.com/artifact/io.quarkus/quarkus-keycloak-admin-client> (accessed Dec. 23, 2021).
- [13] 'Using WebSockets'. <https://quarkus.io/guides/websockets> (accessed Dec. 23, 2021).
- [14] 'Vue.js'. <https://vuejs.org/> (accessed Dec. 23, 2021).
- [15] Node.js, 'Node.js', *Node.js*. <https://nodejs.org/en/> (accessed Dec. 23, 2021).
- [16] 'Material Design', *Material Design*. <https://material.io/design> (accessed Dec. 23, 2021).
- [17] 'Vuetify — A Material Design Framework for Vue.js', *Vuetify*. <https://vuetifyjs.com/en/> (accessed Dec. 23, 2021).
- [18] 'What is Vuex? | Vuex'. <https://vuex.vuejs.org/> (accessed Dec. 23, 2021).
- [19] I. Klymov, *vuex-shared-mutations*. 2021. Accessed: Dec. 23, 2021. [Online]. Available: <https://github.com/xanf/vuex-shared-mutations>
- [20] 'justintaddei/v-shared-element: Declarative shared-element transitions for Vue.js'. <https://github.com/justintaddei/v-shared-element> (accessed Dec. 23, 2021).
- [21] 'Vue-ApexChart - A Vue Chart wrapper for ApexCharts.js', *ApexCharts.js*. <https://apexcharts.com/docs/vue-charts/> (accessed Dec. 23, 2021).
- [22] *axios*. axios, 2021. Accessed: Dec. 23, 2021. [Online]. Available: <https://github.com/axios/axios>
- [23] 'Socket.IO'. <https://socket.io/> (accessed Dec. 23, 2021).
- [24] 'Dashboard / Portal', *Dashboard / Service Matching*. <https://github.com/palantir-h2020/dashboard-portal> (accessed Dec. 23, 2021).
- [25] 'Dashboard / Service Matching', *Dashboard / Service Matching*. <https://github.com/palantir-h2020/dashboard-servicematching> (accessed Dec. 23, 2021).
- [26] M. Edenhill, 'kcat'. Dec. 2021. Accessed: Dec. 13, 2021. [Online]. Available: <https://github.com/edenhill/kcat>
- [27] D. K. Tosh, S. Shetty, S. Sengupta, J. P. Kesan, and C. A. Kamhoua, 'Risk Management using Cyber-Threat Information Sharing and Cyber-Insurance', p. 11.
- [28] J. Hautamäki and T. Hämäläinen, 'A model of Cyber Threat Information Sharing with the Novel Network Topology', p. 10, 2021.
- [29] S. Dahmen-Lhuissier, 'ETSI - CYBER', *ETSI*. <https://www.etsi.org/committee/cyber?jjj=1630054642776> (accessed Dec. 23, 2021).
- [30] 'OASIS Cyber Threat Intelligence (CTI) TC | OASIS'. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cti (accessed Dec. 23, 2021).

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	53 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

- [31] N. N. P. Mkuzangwe and Z. C. Khan, 'Cyber-Threat Information-Sharing Standards: A Review of Evaluation Literature', *Afr. J. Inf. Commun.*, no. 25, pp. 1–12, Jun. 2020, doi: 10.23962/10539/29191.
- [32] E. W. Burger, M. D. Goodman, P. Kampanakis, and K. A. Zhu, 'Taxonomy Model for Cyber Threat Intelligence Information Exchange Technologies', in *Proceedings of the 2014 ACM Workshop on Information Sharing & Collaborative Security - WISCS '14*, Scottsdale, Arizona, USA, 2014, pp. 51–60. doi: 10.1145/2663876.2663883.
- [33] V. Mavroeidis and S. Bromander, 'Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence', in *2017 European Intelligence and Security Informatics Conference (EISIC)*, Athens, Sep. 2017, pp. 91–98. doi: 10.1109/EISIC.2017.20.
- [34] C. S. Johnson, M. L. Badger, D. A. Waltermire, J. Snyder, and C. Skorupka, 'Guide to Cyber Threat Information Sharing', National Institute of Standards and Technology, NIST SP 800-150, Oct. 2016. doi: 10.6028/NIST.SP.800-150.
- [35] P. Kampanakis, 'Security Automation and Threat Information-Sharing Options', *IEEE Secur. Priv.*, vol. 12, no. 5, pp. 42–51, Sep. 2014, doi: 10.1109/MSP.2014.99.
- [36] 'Introduction to STIX'. <https://oasis-open.github.io/cti-documentation/stix/intro> (accessed Dec. 23, 2021).
- [37] 'Microsoft Interflow: a new Security and Threat Information Exchange Platform', *Microsoft Security Blog*, Jun. 23, 2014. <https://www.microsoft.com/security/blog/2014/06/23/microsoft-interflow-a-new-security-and-threat-information-exchange-platform/> (accessed Dec. 23, 2021).
- [38] 'Introduction to TAXII'. <https://oasis-open.github.io/cti-documentation/taxii/intro> (accessed Dec. 23, 2021).
- [39] *oasis-open/cti-taxii-server*. OASIS TC Open Repositories, 2021. Accessed: Dec. 23, 2021. [Online]. Available: <https://github.com/oasis-open/cti-taxii-server>
- [40] *oasis-open/cti-taxii-client*. OASIS TC Open Repositories, 2021. Accessed: Dec. 23, 2021. [Online]. Available: <https://github.com/oasis-open/cti-taxii-client>
- [41] 'OASIS CTI Resources', *CTI Resources*. <https://oasis-open.github.io/cti-documentation/resources#taxii-21-specification> (accessed Dec. 23, 2021).
- [42] *STIX 2.1 Drag and Drop Modeler*. STIX-Modeler, 2021. Accessed: Dec. 23, 2021. [Online]. Available: <https://github.com/STIX-Modeler/UI>

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	54 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

Annex A: PALANTIR functional workflows

In this annex, the deliverable lists the several PALANTIR functional workflows. They describe the main operation scenarios of the PALANTIR platform and frame how T4.1 and T4.3 components are expected to contribute.

SC registration & onboarding

This workflow details how the PALANTIR platform integrate new security capabilities designed by developers.

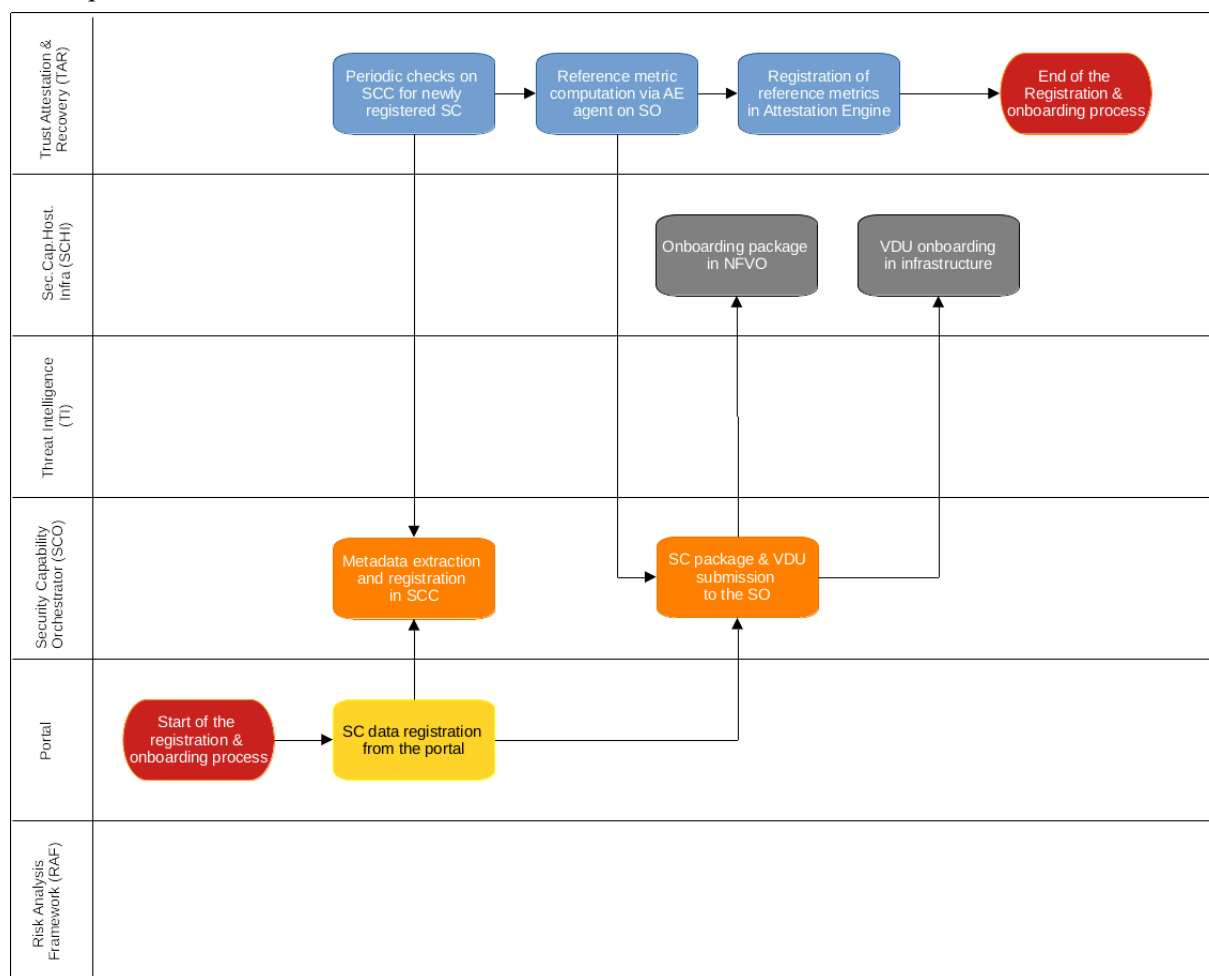


Figure 11: SC Registration and Onboarding Workflow

Initial deployment

This workflow details how a Network Operator from the PALANTIR service provider configures the protection of a new PALANTIR customer.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release					Page:	55 of 58
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

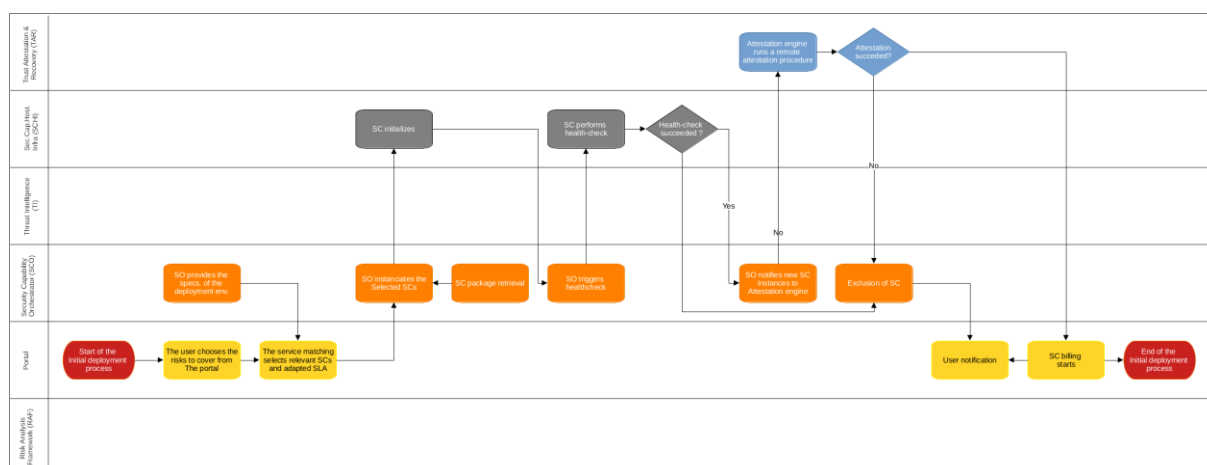


Figure 12: Initial Deployment Workflow

Event handling

The following figure describes how the PALANTIR platform reacts to a new threat occurrence and how it is guided by the Network Operator.

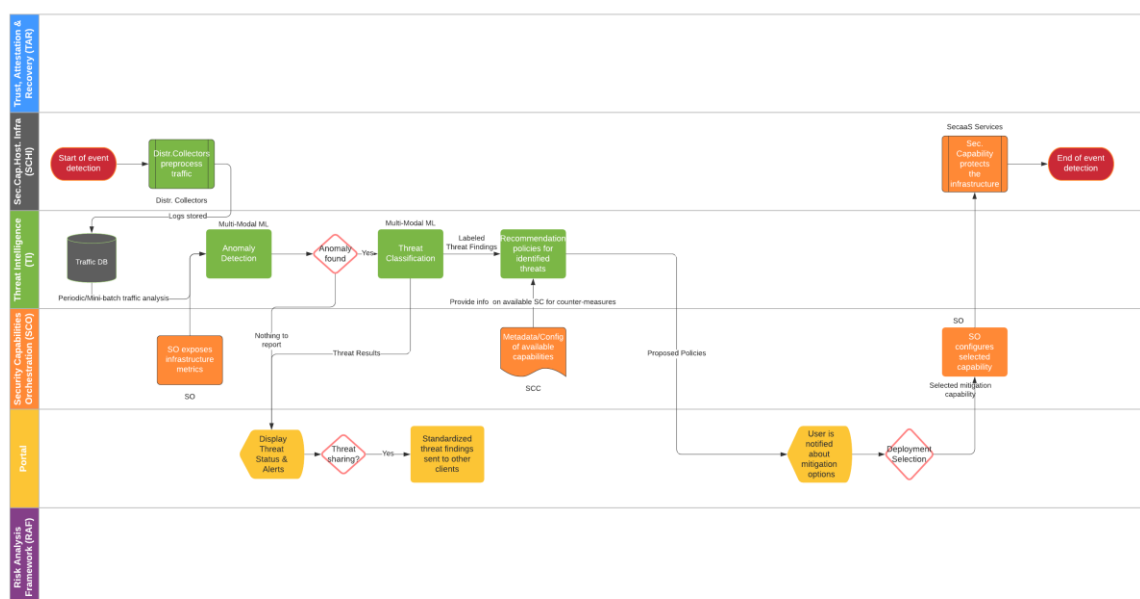


Figure 13: Event Handling Workflow

Periodic attestation

The periodic attestation workflow encompasses how the PALANTIR platform reacts to (i) the occurrence of a new data breach and to (ii) a PALANTIR node failing its attestation.

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	56 of 58	
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

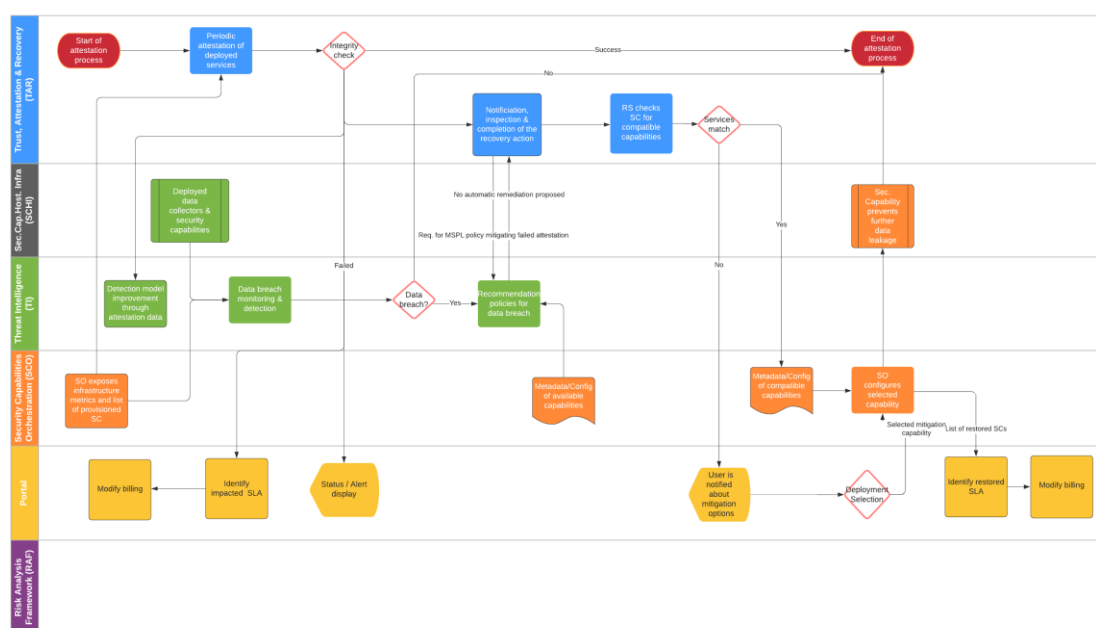


Figure 14: Periodic Attestation Workflow

Fault Management

The fault management workflow specifies how the PALANTIR platform reacts to a security capability fails its health check.

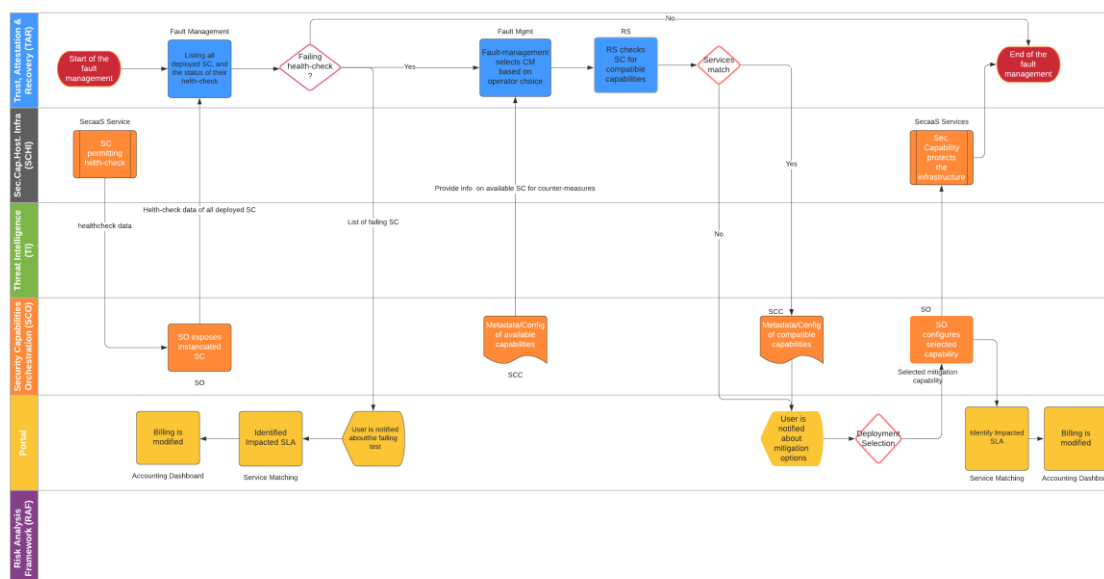


Figure 15: Fault Management Workflow

Document name:	D4.1. PALANTIR Dashboard, Reporting and Threat Sharing Platform - First release				Page:	57 of 58	
Reference:	1.0	Dissemination:	PU	Version:	1.0	Status:	Final

Annex B: Use Case to PALANTIR user role mappings

Based on the use cases, which can be viewed in detail in D2.2, similarities between roles of different scenarios have been found. The main PALANTIR dashboard user roles boil down to two: (a) the Network Operator and (b) the SME Manager. These particular user roles are expected to be the ones mostly active within the use cases, mainly by leveraging the access given to them through the PALANTIR Dashboard. A mapping between the use case roles and the PALANTIR platform roles is shown below.

Table 6: UC roles to PALANTIR role mapping

Use Case No.	Use Case Role	PALANTIR platform Role & brief description
1	Doctor/Healthcare Practitioner	SME Manager: <ul style="list-style-type: none"> Access to threat protection status Access to real-time alerts and notifications.
1	Admin	Network Operator: <ul style="list-style-type: none"> Remediation action upon alert Potential use of SC selection for deployment.
2	Operator	Network Operator: <ul style="list-style-type: none"> Remediation action upon alert Prevention action upon alert Access to real-time alerts and notifications.
2	Manager/Admin	SME Manager: <ul style="list-style-type: none"> Access to real-time alerts and notifications. Threat sharing policy setting
3	Provider	Network Operator: <ul style="list-style-type: none"> Remediation action upon alert Prevention action upon alert Access to real-time alerts and notifications.
3	Infrastructure provider	SME Manager: <ul style="list-style-type: none"> Access to real-time alerts and notifications. Threat sharing policy setting