



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)

D6.1 Integration & Validation Report: Use case results and playbook

Document Identification			
Status	Draft	Due Date	30-06-2022
Version	1.0	Submission Date	30-06-2022

Related WP	WP6	Dissemination Level (*)	Public
Related Deliverable(s)	D2.2, D2.3, D2.4, D3.1, D4.1, D4.2, D5.1		
Lead Participant	SFERA	Lead Author	Izidor Mlakar (SFERA)
Contributors	PALANTIR consortium	Reviewers	Akis Kourtis (ORION)
			Ludovic Jacquin (HPELB)

Keywords:
PALANTIR Pilots, Minimum Viable Product, KPIs, first prototypes, integration, playbook

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:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	2 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

Document Information

List of Contributors	
Name	Partner
Dimitris Papadopoulos, Antonis Litke, Dimitris Giagkos, Orestis Kompougias	INFILI
George Athanasiou	DBC
Carolina Fernández, Maxime Compastié	i2CAT
Antonio López Martínez, Gregorio Martínez Pérez, Manuel Gil Pérez	UMU
Georgios Gardikis	SPH
Ludovic Jacquin, Supreshna Gurung	HPELB
Davide Sanvito, Roberto Bifulco	NEC
Diego López, Antonio Pastor Perales, Jerónimo Núñez Mendoza	TID
Vangelis Logothetis, Ioannis Neokosmidis	INCITES
Anastasios Kourtis, Andreas Oikonomakis, Dimitris Santorinaios	NCSR
Akis Kourtis, George Xilouris	ORION
Stylianios Tsarsitalidis	UBITECH
Izidor Mlakar, Primož Jeran, Valentino Šafran	SFERA

Document History			
Version	Date	Change editors	Changes
V0.1	25/5/2022	SFERA, ORION	Initial draft
V0.2	20/5/2022	WP6 partners	First round of contributions
V0.3	17/6/2022	WP3, 4, 5, 6 partners	Second round of contributions after cross WP alignment
V0.4	20/6/2022	SFERA	Final editing and proofreading, version ready for internal review
V0.5	27/6/2022	SFERA	Final version after peer review
V1.0	29/6/2022	INF	QA check, additional review by i2CAT

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	SFERA	27/6/2022
Quality manager	INFILI	29/6/2022
Project Coordinator	DBC	30/6/2022

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	3 of 77
Reference:	D6.1	Dissemination:	Public	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	10
1. Introduction	11
1.1. Objectives and goals of the deliverable	11
1.2. Relation with D6.1 and other WPs.....	12
2. Minimum Viable Product	13
2.1 The PALANTIR platform, first version.....	13
2.1.1 Risk-based Analysis	14
2.1.2 SecaaS and Security Capabilities Orchestrator	14
2.1.3 Dashboard, Reporting and Threat Sharing	14
2.1.4 Service Matching (SM)	15
2.1.5 Trust Attestation and Verification Framework	15
2.1.6 Personalized remediation and policies	15
2.1.7 Threat Intelligence	16
2.2. PALANTIR workflows.....	17
2.2.1 Risk Assessment Functionality	17
2.2.2 Services Attestation and Verification Functionality	18
2.2.3 Policy design and Personalized Remediation.....	19
2.2.4 SC Registration and Onboarding	22
2.2.5 Initial Deployment	23
2.2.6 Threats detection & reaction	25
2.2.7 Fault and Breach Management.....	26
2.3 PALANTIR Evaluation Methodology, first version.....	27
2.3.1 PALANTIR Evaluation Methodology	27
2.3.2 PALANTIR KPIs	28
3. PALANTIR PILOTS.....	40
3.1. Pilot 1: Securing private medical practices with lightweight SecaaS	40
3.1.1 Motivation and Outline	40
3.1.2 Deployment Model	40
3.1.3 The Testbed Infrastructure	40
3.1.4 Scope and Threats to be demonstrated in Pilot 1	42
3.2. Pilot 2: Uninterrupted Electronic Commerce with Cloud SecaaS	43
3.2.1 Motivation and outline	43
3.2.2 Deployment model.....	44
3.2.3 The Testbed Infrastructure	44
3.2.4 Scope and Threats to be demonstrated in Pilot 2	45
3.3. Pilot 3: Live Threat Intelligence Sharing in a large-scale Edge scenario	46

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	4 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

3.3.1 Motivation and outline	46
3.3.2 Deployment model	47
3.3.3 The Testbed Infrastructure	47
3.3.4 Scope and Threats to be demonstrated in Pilot 3	48
4. PALANTIR Testbed and Innovation Labs	50
4.1. Installation guide	50
4.1.1 Dashboard – Portal	50
4.1.2 Dashboard - Service Matching	52
4.1.3 SecaaS – Security Capabilities	54
4.1.4 Security Capabilities Orchestration - Security Capabilities Catalogue	54
4.1.5 Security Capabilities Orchestration - Security Orchestrator	56
4.1.6 Trust Attestation & Recovery - Attestation Engine	56
4.1.7 Trust Attestation & Recovery - Fault and Breach Management	57
4.1.8 Threat Intelligence - Anonymization Service	57
4.1.9 Threat Intelligence - Anonymization Service	57
4.1.10 Threat Intelligence - Data Pre-processing	58
4.1.11 Threat Intelligence - Multi-Modal Machine Learning - Anomaly Detection	58
4.1.12 Threat Intelligence - Remediation Engine	58
4.1.13 Deployment models	58
4.2. PALANTIR MVP 1.0, demonstration	60
4.2.1 Storyline 1: Botnet Attack	60
4.2.1.1 Risk Assessment - Risk Profile Selection	61
4.2.1.2 Risk Assessment - Asset Identification	62
4.2.1.3. Service Matching	63
4.2.1.4. Threat Intelligence	64
4.2.2 Storyline 2: Data Breach Detection and Recovery via AI-based Log Analysis	66
4.2.3 Storyline 3: Attestation and Attestation Failure	71
4.2.4 HPE-Attestation Engine Standalone Demo	72
5. Conclusions	76
6. References	77

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	5 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

List of Tables

<i>Table 1: PALANTIR KPIs</i>	39
<i>Table 2: A Draft of PILOT 1 specific KPIs</i>	43
<i>Table 3: A Draft of PILOT 2 specific KPIs</i>	46
<i>Table 4: A Draft of PILOT 3 specific KPIs</i>	49
<i>Table 5: PALANTIR Deployment models</i>	60

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	6 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

List of Figures

Figure 1: Conceptual view of the PALANTIR solution.....	12
Figure 2: The Conceptual Design of the PALANTIR Platform.....	13
Figure 3: The Risk Assessment Block: The four phases for Risk Management.....	17
Figure 4: The Conceptual Design of the Risk Assessment and Recommendation of PALANTIR services functionality.....	18
Figure 5: Initial Attestation workflow.....	18
Figure 6: Periodic attestation and remediation for failed attestation workflows.....	19
Figure 7: an example of a personalized policy, (a) the FSM representation and (b) The UML snippet.....	20
Figure 8: The initial design Policy design and Personalized Remediation workflows.....	21
Figure 9: The initial design of the SC Registration and Onboarding Workflow.....	23
Figure 10: The Workflow for the initial deployment process, current implementation.....	24
Figure 11: The Workflow for the deployment process to be implemented.....	24
Figure 12: The Workflow for Threats detection & reaction in PALANTIR Platform, part 1.....	25
Figure 13: The Workflow for Threats detection & reaction in PALANTIR Platform, part 2.....	26
Figure 14: The Initial Workflow designed for the integration of Fault Management.....	27
Figure 15: Lightweight SecaaS deployment model in Pilot 1.....	41
Figure 16: The detailed representation of the vCPE.....	41
Figure 17: The Actor diagram for Pilot 1 as Reported in D2.4.....	42
Figure 18: The Pilot 2 Testbed.....	44
Figure 19: The Actor diagram for Pilot 2 as Reported in D2.4.....	45
Figure 20: The testbed infrastructure for Pilot 3.....	47
Figure 21: The 5G edge location for Pilot 3.....	48
Figure 22: The Actor diagram for Pilot 3 as Reported in D2.4.....	48
Figure 23: Storyline 1 high-level workflow.....	61
Figure 24: Risk Profile Selection questionnaire sample.....	62
Figure 25: Asset Identification questionnaire sample.....	63
Figure 26: Service Matching Screenshot in Storyline 1.....	63
Figure 27: Logging output of all Threat Intelligence components.....	64
Figure 28: Dashboard notification after a new threat is detection.....	65
Figure 29: Dashboard notifications after the remediation has been correctly applied.....	66
Figure 30: Infrastructure topology for Storyline 2.....	66
Figure 31: Brute-force attempt on the protected infrastructure.....	67
Figure 32: The logs generated by the brute-force attack are successfully identified as outliers by MAD.....	67
Figure 33: TCAM labels the outliers as “hydra-ssh” attack.....	68
Figure 34: The network operator is notified about the data breach incident via the Portal.....	68
Figure 35: A double mitigation policy is proposed by RR.....	69
Figure 36: The firewall SC interrupts the brute-force attack.....	69
Figure 37: IR initiates the backup process after the RR trigger.....	70
Figure 38: IR initiates the backup process after the RR trigger.....	70
Figure 39: Workflow for Storyline 3.....	71
Figure 40: Identification and notification of a compromised container in Storyline 3.....	72
Figure 41: HPE-OS Verification: Failed attestation alert due to IMA policy violation.....	73
Figure 42: HPE-FirmAtt: Failed Firmware Attestation alert.....	74
Figure 43: HPE-RMmangt: Updating new Reference Measurement.....	74
Figure 44: HPE-DIME: New kernel module added alert in iLO.....	75

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	7 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0
		Status:	Final		

List of Acronyms

Abbreviation / acronym	Description
BYOD	Bring Your Own Device
CAPEC	Common Attack Pattern Enumeration and Classification
CAPEX	Capital Expenditure
CRM	Customer Relationship Management
CSP	Cloud Solution Provider
CVE	Common Vulnerabilities and Exposures
DL	Deep Learning
DNS	Domain Name System
ENISA	European Union Agency for Cybersecurity
EPC	Evolved Packet Core
FBM	Fault and Breach Management
FTP	File Transfer Protocol
FSM	Finite State Machine
GUI	Graphical User Interface
IoC	Indicators of Compromise
iSCSI	Internet Small Computer Systems Interface
ISP	Internet Service Provider
KVM	Kernel Virtual Machine
LAN	Local Area Network
LTE	Long Term Evolution
ME	Microenterprise
MEC	Multi-access Edge Computing
MISP	Malware Information Sharing Platform
ML	Machine Learning
MTTD	Mean Time to Detect or Discover
MTTR	Mean Time to Recovery
NAT	Network Address Translation
NFC	Near Field Communication
NFV	Network Functions Virtualization
NFVO	Network Function Virtualisation Orchestration
NIST	National Institute of Standards and Technology
NSA	Non-Standalone
OFDM	Orthogonal Frequency Division Multiplexing
OSM	Open-Source MANO
PAN	Personal Area Network
RAN	Radio Access Network
PoP	Point of Presence
SA	Stand Alone
SAS	Serial Attached SCSI

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	8 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

Abbreviation / acronym	Description
SDN	Software-Defined Networking
SFTP	Secure File Transfer Protocol
SLA	Service-Level Agreement
SM	Service Matching
SQL	Structured Query Language
UC	Use Case
VIM	Virtualized Infrastructure Manager
VNF	Virtualized Network Function
WAN	Wide Area Network
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WPA	Wi-Fi Protected Access
5G PPP	5G Infrastructure Public Private Partnership

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	9 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

Executive Summary

The present document incorporates details on the first version of the PALANTIR integrated solution and the three test beds, i.e., the PALANTIR Pilots, that demonstrate the solution and the platform.

PALANTIR provides a multi-layered, infrastructure-wide approach for threat monitoring, cyber-resiliency, and knowledge sharing in heterogeneous platforms. Infrastructure is built upon the features of Network Functions Virtualization (NFV), scalable Machine Learning (ML) towards hybrid Threat Intelligence, attestation techniques for secure infrastructure and trusted services, as well as standardization and threat-sharing methods to risk analysis, network operation, monitoring and management.

Section 1 serves as an introductory section to the deliverable and a compact presentation of the effort carried out in the WP, as well as the inter-relations of WP6 to other WPs.

In Section 2 of this document, the first version of the PALANTIR Minimum Viable Product (MVP) is described, i.e., the first version of the PALANTIR platform. To this end, Section 2 outlines activities related to the integration and adaptation of all different modules, components, APIs, and algorithms defined, designed, and implemented into a complete functional platform to be deployed in a suitable qualification environment, resulting from the integration of all these elements and providing a first prototype suitable for tests and validation. Section 2 also outlines the evaluation methodology and KPIs to be used to assess the outcomes and PALANTIR systems as a whole.

In order to demonstrate and evaluate the first prototype, three pilots were envisaged as a suitable qualification environment. The pilots are outlined in detail in Section 3 of this document. The pilots are deployed in three distinct locations, situated in Athens (EL), Madrid (ES), and Maribor (SI). More specifically:

- Pilot 1 is deployed in Athens (EL) and implements a Lightweight SecaaS deployment model for the protection of small businesses from data breaches and ransomware attacks. The testbed is designed to leverage the PALANTIR solution in the scope of medical data protection, illustrating relevant cases of incident detection and mitigation activities to safeguard patient data and prevent medical identity theft.
- Pilot 2 is deployed in Maribor (SI) and implements a hybrid deployment model with Cloud SecaaS deployment for most of the PALANTIR components and Lightweight SecaaS deployments to enable full access to the distributed nature of a retail and service-oriented Microenterprise business environment, with on-premises and cloud-based solution. The testbed is designed to leverage the PALANTIR solution and demonstrate multiple threat scenarios (including targeted attacks such as spyware/ransomware and DDoS) on the corporate infrastructure and the e-commerce platform.
- Pilot 3 is jointly deployed in Athens (EL) and Madrid (ES). Its goal is to experimentally demonstrate the PALANTIR solution's operational capacity in the 5GENESIS and 5TONIC testbeds. These 5G-enabled testbeds can emulate traffic from multiple SecaaS clients on their edge network as well as complex parallel attacks in large-scale MEC scenarios.

Section 4 is dedicated to an outline of the PALANTIR Testbed and Innovation Labs. This section aims at providing data required for the PALANTIR Workshops' organization and enable external stakeholders to experience and experiment with PALANTIR testbed and get accustomed to its usage. To this end PALANTIR Developers designed the first draft version of an installation guide. The PALANTIR demonstrators have outlined the first demonstration of PALANTIR, the MVP version 1.0, in three storylines and one standalone scenario. This section will be further evolved and refined in Deliverable D6.2.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	10 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

1. Introduction

1.1. Objectives and goals of the deliverable

The current document is the deliverable “D6.1 Use Cases and Risk Reduction Measures” which comprises the major outcomes of “**Task 6.1 - T6.1. Component testing, Integration, Qualification and Testing**”, “**T6.2. UC 1: Securing private medical practices with lightweight SecaaS**”, “**T6.3. UC 2: Uninterrupted Electronic Commerce with Cloud SecaaS**” and “**T6.4. UC 3: Live Threat Intelligence Sharing in a large-scale Edge scenario**”, building upon the findings of the “D2.3 Requirements & High Level design - Final” and “D2.4 Use Cases and Risk Reduction measures”, and development progress reported in “D3.1 PALANTIR Secure Services Platform-First Release”, “D4.1 Dashboard, Reporting and Threat Sharing Platform-First Release”, “D4.2 Trust, Attestation and Verification Framework-Specifications of first release” and “D5.1 Hybrid Threat Intelligence Framework- First release”.

Task 6.1 focuses on activities related to the integration and adaptation of all the different modules, components, APIs, and algorithms defined, designed, and implemented in other work packages. The goal is to achieve a functional platform (i.e., PALANTIR Minimal Viable Product), deployed in a suitable qualification environment, resulting from the integration of all these elements and providing a prototype suitable for tests and validation. Integration is carried out on a step-by-step basis under a planned procedure, running local interoperability tests after each step. The steps that follows are: i) technical identification of the results produced already (functional tests), ii) implementation of the integrated prototype platform (modular tests, inter-modular tests), iii) technical assessment of the outcome (scalability tests, security tests), and iv) fine-tuning and overall optimization.

Task 6.2 is dedicated to implement Use Case 1 pilots, focused on Lightweight SecaaS for the protection of small businesses from data breaches and ransomware attacks. To this end, the PALANTIR solution will be leveraged in the scope of medical data protection, illustrating relevant cases of incident detection and mitigation activities to safeguard patient data and prevent medical identity theft. T6.2 will be deployed on one of PALANTIR’s experimental facilities (Athens testbed), together with the required functions and software. T6.2 will showcase the deployment by means of remote access, on specific on-demand demos (for exploitation goals) and those events the project attends (for dissemination goals).

Task 6.3 implements Use Case 2 pilots, built around secure electronic commerce, with the example of a typical retail and service-oriented Microenterprise that also maintains an e-shop. The subcontracted ME comprises of 3 offices located in 3 different cities in Slovenia and daily processes real customer and corporate data. First, the appropriate infrastructure will be deployed and integrated, based on the Slovenia testbed (this deliverable). T6.3 will then study multiple threat scenarios (incl. targeted attacks such as spyware/ransomware and DDoS) on the corporate infrastructure and the e-commerce platform and will assess the effectiveness of the PALANTIR framework in order to combat the attacks, attest the integrity of the infrastructure and exploit the threat sharing capabilities of the platform (upcoming deliverable D6.2).

Task 6.4 will demonstrate the PALANTIR SecaaS-protected network on a realistic, large-scale scenario, in which it will be tasked with jointly analysing data from multiple vendors (i.e. SMEs & MEs) and with leveraging cyber threat intelligence information to and from national and international knowledge sharing infrastructures (e.g. via interfacing with MISP) to deploy tailored cybersecurity measures. T6.4 will be demonstrated on the Madrid 5TONIC [1] and Athens 5GENESIS [2] testbeds which will facilitate the traffic emulation on the clients’ edge network and the simulation of complex network attacks, by exploiting the Mouseworld lab from partner organization TID. PALANTIR will be expected not only to successfully produce a mitigation plan for the targeted clients but also to take full advantage of its live threat sharing module by deploying recommendations and proactive cybersecurity measures to the other clients of the protected network. The primary audience of this document consists of the core PALANTIR stakeholders who will participate in the deployment and use of PALANTIR Solutions, i.e.,

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	11 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

Service Providers, PALANTIR Operators and Platform Integrators, SME owners, Service Developers and GDPR subjects. Although the primary audience mainly consists of consortium members who will design and implement the components and modules of the PALANTIR system, this document is of wider interest to extended communities of cyber security stakeholders in order to drive and foster the adoption of PALANTIR solution and standardization for the SME/ME sector.

1.2. Relation with D6.1 and other WPs

The presented pilots were designed in conjunction with the elicitation of the interim version of the PALANTIR requirements documented in D2.1 and overall system architecture documented in D2.3, D2.4, and a conceptual view of the PALANTIR architecture is shown in Figure 1 to facilitate readability and tracking of the Use Case workflows.

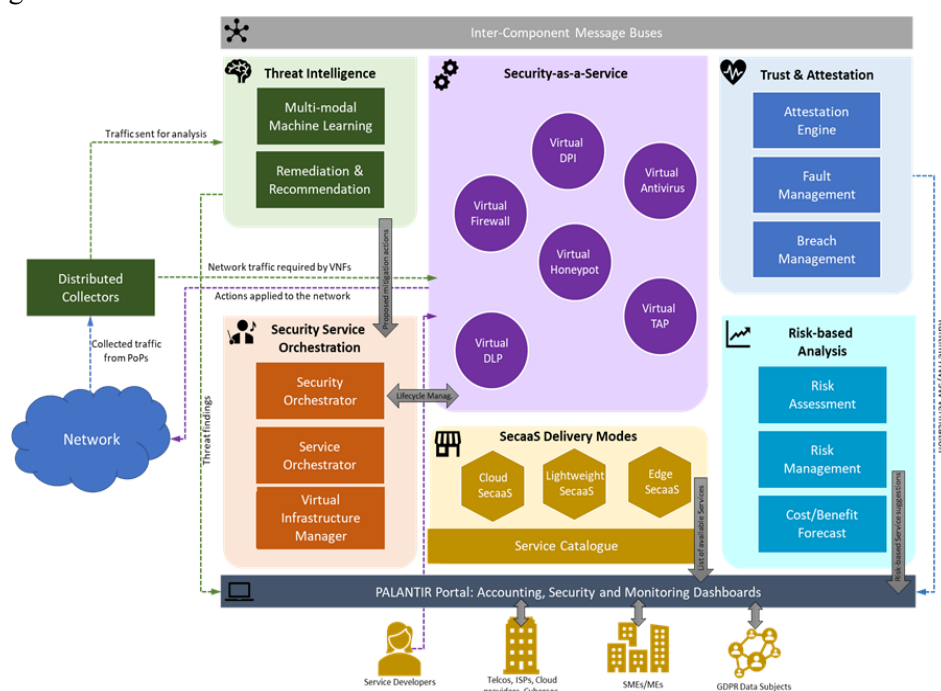


Figure 1: Conceptual view of the PALANTIR solution

A brief description of the workflow between the PALANTIR components is also provided below:

- The *Risk-based analysis* component allows the quantification of security/privacy threats based on security/privacy impact assessment and its correlation with attack surface analysis.
- *Threat Intelligence* traces traffic from the network and VNFs through *Distributed Collectors*, analyses it for signs of malicious activity and outputs the detected anomalies to the *Remediation Engine*.
- The *Remediation Engine* proposes reactive measures against cyberattacks (security rules, new topologies etc) to the *Security Service Orchestrator*.
- The *Security Service Orchestrator* pushes back selected actions and lifecycle management messages to the running *SecaaS*.
- The *Trust, Attestation & Recovery* component periodically attests the infrastructure's physical and virtual components for signs of compromise and do the necessary recovery steps

Furthermore, the present deliverable is linked to the technical WPs (**WP3**, **WP4**, **WP5**) to ensure that technological developments are generally aligned with the pilot demonstration.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	12 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

2. Minimum Viable Product

This section reports on the integration activities related to Task 6.1 with its main objective to create and maintain the PALANTIR testbed infrastructure and set up the mechanisms to monitor and validate the PALANTIR platform. Within this section, we outline the PALANTIR Platform as the integrated PALANTIR solution (Section 2.1), we highlight the operational workflows between components (Section 2.2) and outline the first version of the PALANTIR Evaluation methodology to be used to evaluate the efficiency and applicability of PALANTIR Solution.

2.1 The PALANTIR platform, first version

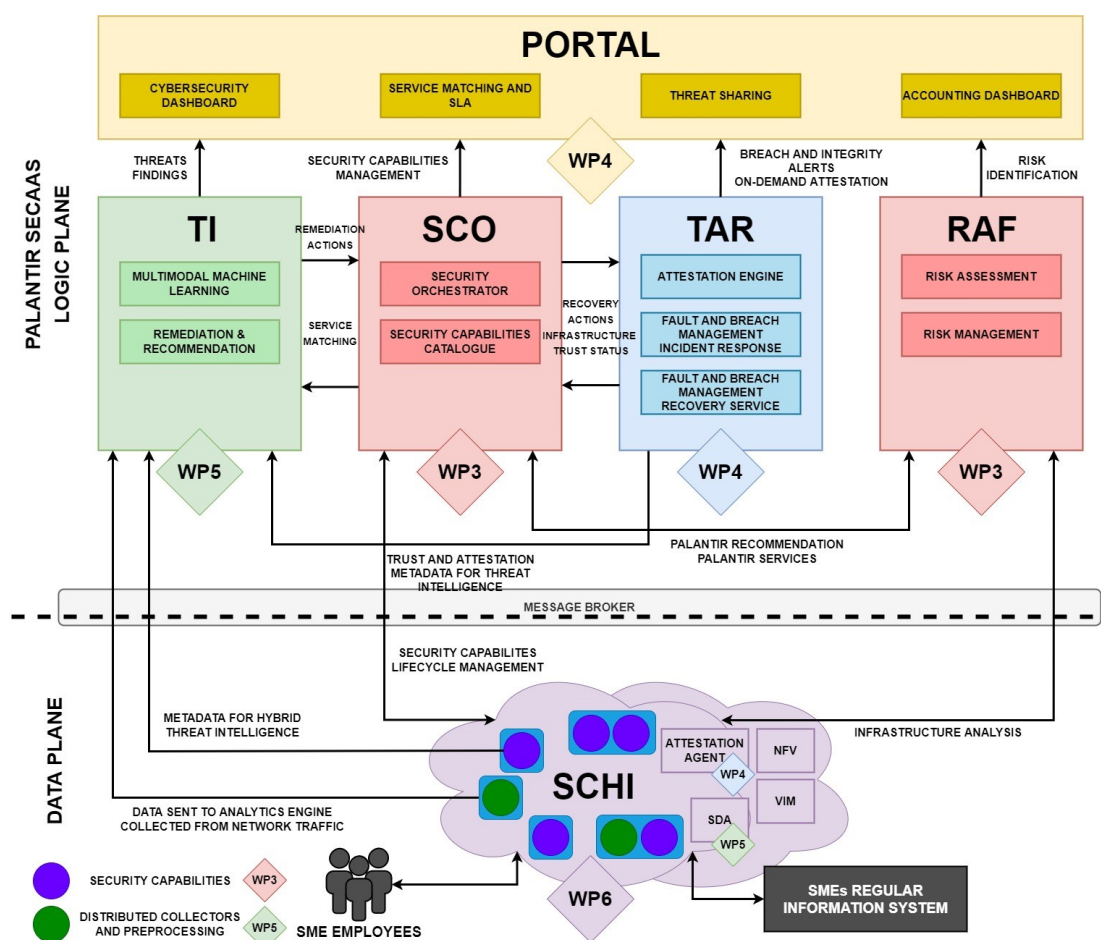


Figure 2: The Conceptual Design of the PALANTIR Platform

Figure 2 outlines how PALANTIR components and stakeholders are interconnected within the PALANTIR platform. Risk-based Analysis Framework (RAF) is used to analyse the targeted business, identify risks, and identify appropriate PALANTIR Services using the SecaaS and Security Capabilities Orchestrator (SCO). Inside the SCO, the Security Orchestrator (SO) operates at the level of the SME infrastructure to deliver monitoring and mitigation features adapted to the security posture. The Trust, Attestation and Recovery Framework (TAR) is responsible for continuously monitoring the PALANTIR's Security Capability Hosting Infrastructure (SCHI) to detect signs of attacks or erroneous behaviour. The Threat Intelligence (TI) oversees the protection provided by the security capabilities with advanced analytics mechanisms based on ML and Deep Learning (DL) to detect threats and to generate remediations able to address the detected threats automatically. And finally, the Portal, i.e., the PALANTIR Dashboard, represents the entry point of the platform, with different functionalities exposed

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	13 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

to different user roles. The functional role of each component is described in detail in the following subsections.

2.1.1 Risk-based Analysis

Risk assessments can be held quantitatively or qualitatively. Quantitative risk assessments require monetary or numerical values for risk factors whereas qualitative methods employ non-numeric priority or criticality values. We employ a quantitative approach in our model due to three reasons. First, the underlying metrics of the CVSS has numerical values assigned to them since the CVSS is a quantitative approach. Second, in quantitative approach, the evaluation and the results are based on objective criteria and thus more suitable for an IT system risk assessment. Lastly, quantitative approach is more suited for measuring the security level of an IT system in terms of the three common security pillars (confidentiality, integrity, and availability).

2.1.2 SecaaS and Security Capabilities Orchestrator

The Security Capabilities (SCs) are the minimal action unit in PALANTIR. Each of these instances intervenes at the level of the SME infrastructure to deliver a specific monitoring and mitigation feature adapted to the security posture. They directly depend on the SO for their configuration.

On the other hand, the SCO is the main PALANTIR pillar devoted to the management of the images and running instances of the SCs. It encompasses two subcomponents:

- **Security Capability Catalogue (SCC):** permits developers to share and monetise their Security Capability implementation by making it available to the platform for exploitation. It also provides a searchable registry that eases the interaction with both the SO and the Service Matching (SM), when selecting appropriate SCs.
- **Security Orchestrator (SO):** oversees the lifecycle management of the SCs by conducting operations to instantiate, de-instantiate and reconfigure the SC instances through actions that are aligned with the security operator directives. It also provides means to monitor custom metrics on the infrastructure, which is used during the attestation procedure; and is also able to monitor generic and/or custom metrics on the running SC instances, which can be used for further threat analysis, security posture assessment and billing computation.

2.1.3 Dashboard, Reporting and Threat Sharing

The PALANTIR Dashboard is the entry point of the platform, with different functionality exposed to different user roles. Dashboard encompasses both the Cybersecurity Dashboard and the SM. There are three subcomponents, which, coupled with the rest of the platform, provide a view of the network's security:

- **Portal backend:** It connects to the message broker of the platform to collect threats, incidents and actions done on the network, as events created by other components. It stores threats, actions, and incidents, and provides access to them via REST API. It also enables user management, user authentication, user role assignment, and associating users with assets of the organization.
- **Threat sharing:** It collects threats and Indicators of Compromise (IoC) from the message broker and portal backend. It allows for setting up the sharing policy, in order to control what kind of data is to be shared, and how. Finally, it provides access to threat intelligence collected via the platform and provides access to other IoC and threat knowledge bases.
- **Dashboard Web Application:** The frontend of the dashboard, the user-facing GUI application that enables users to access the platform. It provides visual access functionality exposed by the dashboard subcomponents, as well as the SM, Security Capabilities Catalogue, Billing Framework, Risk Assessment Framework, Security Capabilities Orchestrator, and all Threat Intelligence subcomponents. Finally, it visualizes notifications regarding actions or threat events, in real-time.

The main types of users of the dashboard, and the features related to them, are:

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	14 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

- **Network Operator:** They have full access to the technical aspects of the platform and can specify actions to be enacted. They are also notified about the completion of an action on the platform, or the occurrence of an attack or threat, in real-time.
- **SME Manager:** They have access to the managerial and financial aspects of the platform and can set policies for threat intelligence sharing. They too are notified about various events in real-time.
- **Security Capabilities Developer:** They can register and onboard new security capabilities and update them. They can also view how well their SCs are performing.

2.1.4 Service Matching (SM)

The SM intervenes in the selection of the adequate security capability compatible with the technicity of the deployment environment it should be instantiated in. When a protective feature should be enacted to respond to a risk to mitigate or a detected threat, the component will autonomously pinpoint pertinent implementation from the catalogue able to cope with the situation, analyse the capacity and properties of the infrastructure, and propose a solution accompanied with a quote to the security operator through the portal interface. The SM can also trigger the orchestrator to proceed with an effective deployment. Reciprocally, when a security capability instance ceases to behave as expected (due to failed integrity checks as reported by the attestation engine or malfunction as alerted by the recovery service), the SM identifies which subscription is being impacted, and signal the billing framework to adapt the accounting accordingly and notify the SME manager.

2.1.5 Trust Attestation and Verification Framework

The TAR is responsible for continuously monitoring the PALANTIR's Security Capability Hosting Infrastructure (SCHI) to detect signs of attacks or erroneous behaviour. The TAR is also leveraged by the Security Capabilities Orchestration to ensure no untrusted node or capability is used to enforce the PALANTIR SecaaS solution. The TAR comprises of three components, the Attestation Engine (AE), the Recovery Service (RS) and the Fault and Breach Management (FBM) component.

The AE utilises the Trusted Computing paradigm to continuously monitor SCHI to detect subversion of the hardware, firmware, software, or configurations on SCs and hosting environment and assessing trust. The Trusted Computing paradigm builds on Roots of Trust (RoT) to protect critical data (e.g., cryptographic keys, secrets), particularly a Trusted Platform Module (TPM), and software methods, such as measured boot, to enable trust verification of the platforms. The remote attestation protocol verifies the integrity of platform by comparing the list of measurements recorded in the TPM since boot with the list of measurements representing the expected software and configuration of the platform. An incorrect, missing, or additional measurement evidences an unexpected security posture of the platform. These Trust Computing methods are complemented with new methods and solutions to perform hardware attestation, containerised workload attestation and runtime monitoring such as memory inspection capability of the platform to detect any unexpected change of code or data already loaded in memory.

PALANTIR-protected infrastructure hosts one RS instance that is responsible for deploying recovery policies when attestation faults are detected. In addition to automation and mitigation services offered by the SO, the FSMs also include alerts/messages and interventions that require human action. In such cases, notifications or requests can be sent to the Dashboard based on the FSM result. Notification or request can then be handled by a PALANTIR operator. The goal of the RS component is to periodically check the status of the PALANTIR platform and to respond to possible faults or malfunctions. Similarly, to the IR, a RS policy is defined with SME/ME specific recovery goals.

2.1.6 Personalized remediation and policies

The PALANTIR infrastructure hosts an instance of the Recommendation and Remediation tool that is in charge of reacting to the threats reported by the TCAM. Reactions may comprise the reconfiguration of existing security capabilities in the managed networks, the deployment of new security capabilities,

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	15 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

or the exploitation of other services in the PALANTIR infrastructure, for example the FBM service provided by the IR instance. The reactions are based on recipes, i.e., generic sequences of operations that are needed to react to a specific threat (e.g., botnet attacks). The tool is able to adapt the generic operations defined in the recipes to the status of the network, such as the presence of a specific security capability needed to deploy a recipe, for example the iptables packet filter.

PALANTIR-protected infrastructure hosts one IR instance that is responsible to trigger mitigation policies when a threat/attack related to data breach is detected by TI. The objective of the IR is to handle remediation policies for threat mitigation, especially those that cannot be handled by, for instance, the cross-system or stakeholder notifications which may be specific for an individual entity (ME/SME). IR implements the predefined FSMs to tackle with the specific incident. Eclipse Papyrus is the environment used for modelling and creating FSMs. Eclipse Papyrus [3] provides a straightforward "drag and drop" procedure for exporting the finished FSM model to a UML document containing the FSM settings. The Spring State Machine is then built inside the Spring Boot Java application using the UML document.

2.1.7 Threat Intelligence

The Threat Intelligence is in charge of complementing the protection provided by the security capabilities with advanced analytics mechanisms based on ML and DL and to automatically generate remediations able to address the detected threats. It comprises four main components:

- **Distributed Collection and Data Processing (DCP):** collects data from heterogeneous sources, pre-process them (e.g., data is anonymised) using scalable pipelines able to cope with big volume of data. It also includes an additional dashboard that enables the monitoring of the health and resources usage of all deployed components
- **Multi-Modal Anomaly Detection (MAD):** starting from the data collected by the DCP, it runs a set of anomaly detection modules able to detect abnormal behaviours from the data. Two pipelines operating on two data modalities (network traffic and system logs collected from end-hosts) are run in parallel and include approaches based on DL, ML, and dynamic graphs
- **Threat Classification & Alarm Management (TCAM):** starting from the outliers detected by the MAD, it is responsible for classifying them either as false positives or as specific types of threats, providing a corresponding confidence score for each predicted label. The two data modalities are supported through a pair of ML models to classify complementary attack scenarios. Both MAD and TCAM modules have been developed using scalable ML frameworks to efficiently accommodate big data processing.
- **Recommendation & Remediation (RR):** given the output of TCAM, it generates the correct strategy to react to network attacks and malware host infections. It analyses the most suitable recipe (set of operations to handle a specific attack type), customises it for the specific network topology, generates the required reconfiguration commands and sends them to the corresponding security capabilities, while at the same time informing the network operator through the dashboard.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	16 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

2.2. PALANTIR workflows

During PALANTIR's design process, the project has elicited from a set of functional and technical requirements a set of components to be implemented to meet the project objectives. Each of those have defined interfaces, procedures to exchange information and coordinate their action into a common objective. In this section, we list a set of use-case-agnostic procedures, detail the components' interactions needed to fulfil them, and document them as documented sequence diagram.

2.2.1 Risk Assessment Functionality

RAF provides a simplified and comprehensive view of risk management or risk assessment for use within SMEs. The philosophy of this framework is to be suitable for collecting information even from non-experts and avoid obfuscating workflows for the information entry. The phases anticipated for RAF remain as described in D2.2, D2.4 and are depicted in Figure 3.

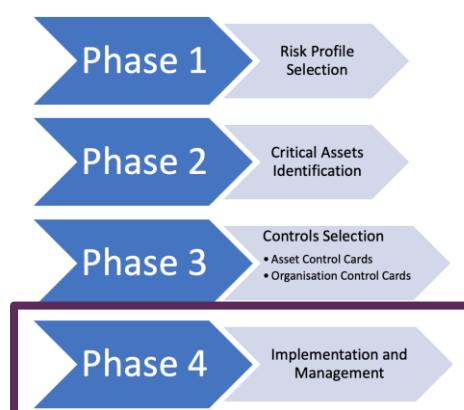
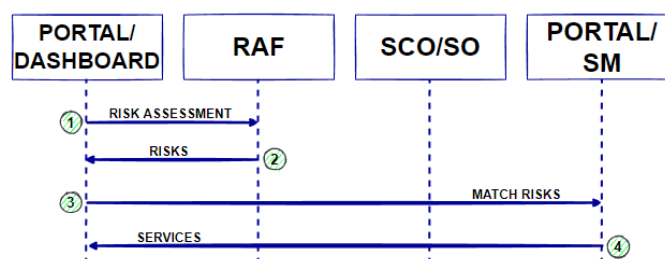


Figure 3: The Risk Assessment Block: The four phases for Risk Management

In this deliverable, we focus on Phase 4 of the Risk Assessment, i.e., the Implementation and Management. During this phase the analysis team identifies actions and recommends an action list, setting forth the direction for security improvement. Many of the identified risks should be resolvable directly through the PALANTIR platform. Namely, PALANTIR, being a holistic framework, takes into consideration asset control cards and provides means for implementation of mitigation for limiting the risk. Figure 4 outlines the role of RAF component in such cases, and how RAF framework can be used to optimise, personalise, and automate the “protective functionality” of PALANTIR. Based on the selection of the risk assessment on the Portal (1), the RAF provides suggested risk remediation functionalities to be leveraged as a result (2). The selected functionalities are matched with available SCs using the SM component (3). The SM autonomously pinpoints the services from the catalogue able to cope with the situation, analyse the capacity and properties of the infrastructure, and propose a solution accompanied with a quote to the security operator through the portal interface (4).



Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	17 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

Figure 4: The Conceptual Design of the Risk Assessment and Recommendation of PALANTIR services functionality

2.2.2 Services Attestation and Verification Functionality

The AE is responsible for establishing the trustworthiness of the SCHI and SCs running on it. To this end, it applies trusted computing attestation to the physical nodes and the SCs, starting at the initial deployment and throughout their entire lifecycle.

The SCO orchestrates and manages the SCHI and SCs and notifies the AE of any event concerning a change in the infrastructure topology (deployment or removal of physical nodes and SCs). The SO also hosts a Reference Measurement plugin that generates the known-good measurements for the new nodes and SCs deployed in the SCHI; where these baseline sets of measurements are required by AE to execute the remote attestation procedure. Each physical node in the SCHI hosts an Attestation Agent, in charge of forwarding the integrity measurements of the platform and the SCs running on it to the AE, that uses them during the remote attestation procedure. The Recovery Service (RS) intervenes every time an integrity failure occurs, sending the SO the appropriate remediation actions to be undertaken to re-establish the security posture of the SCHI.

Figure 5 shows the workflow which occurs when a new platform is introduced in the SCHI, or a new SC instance is deployed on it. The SO notifies the AE upon registration of a new node managed or overseen by the former, and (when applicable) also provides some infrastructure-related metrics for such node. The AE performs a remote attestation cycle on the new platform / SC in order to evaluate its trustworthiness. In the case of a physical node, the AE verifies that no unauthorized changes have been made to the hardware devices on the platform and that all its software, from the UEFI up to the application layer, has not been compromised; in the case of a SC, the AE verifies that its runtime environment is trusted. If the verification fails, the AE sends a failure response to the RS, which suggests the SO not to use that instance of SC or to exclude the physical node from the infrastructure. Otherwise, the new node / SC remains in the infrastructure and will be periodically attested.

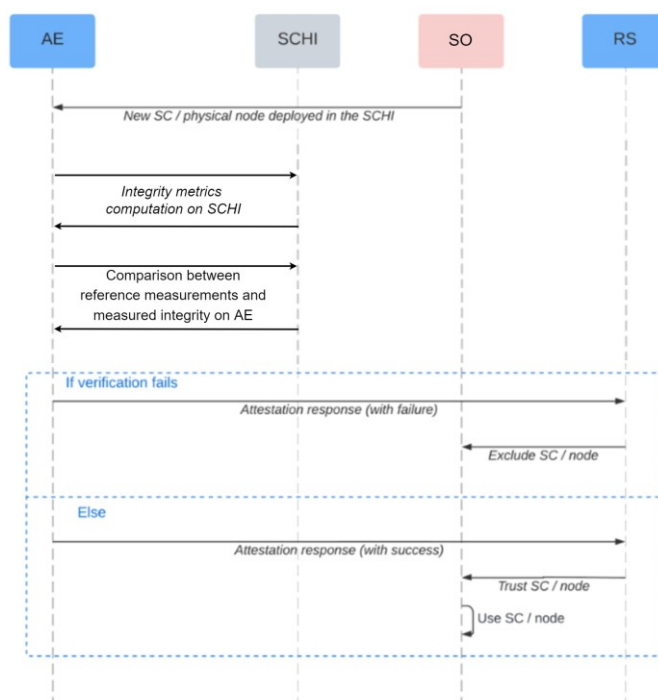


Figure 5: Initial Attestation workflow

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	18 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

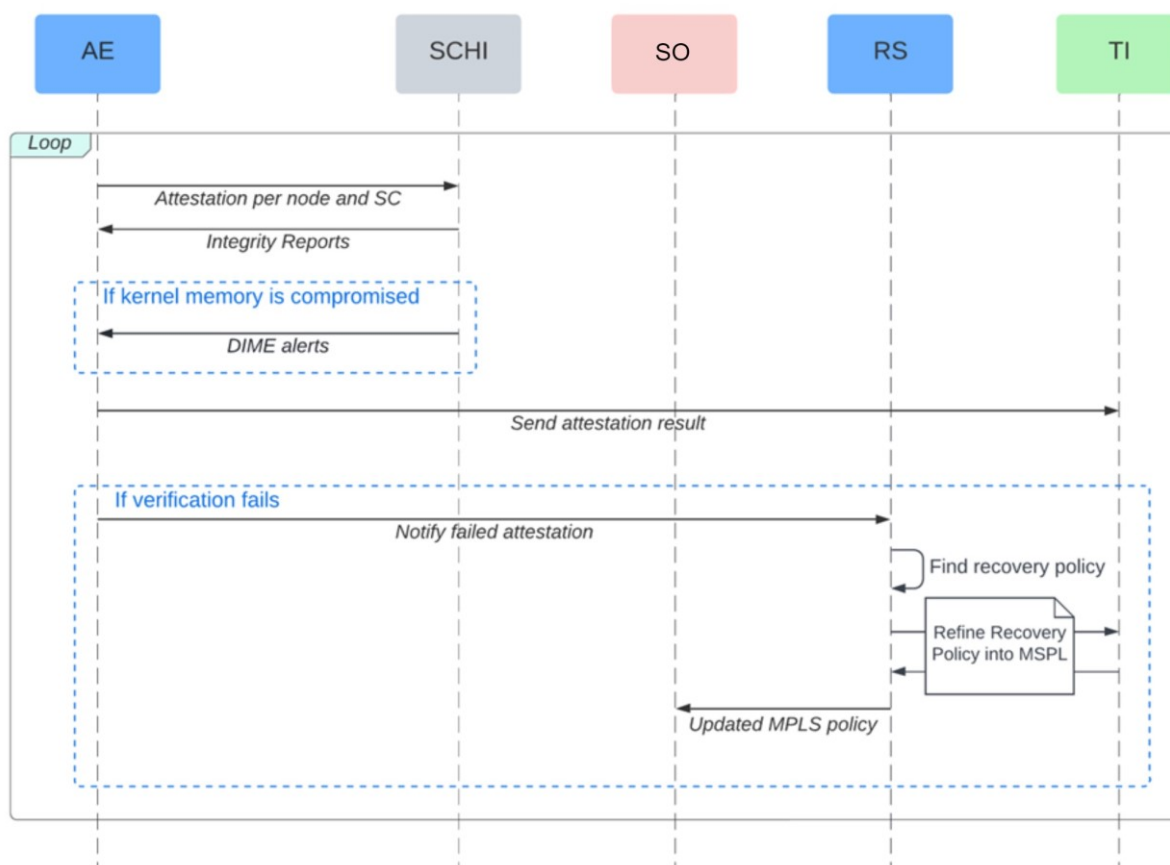


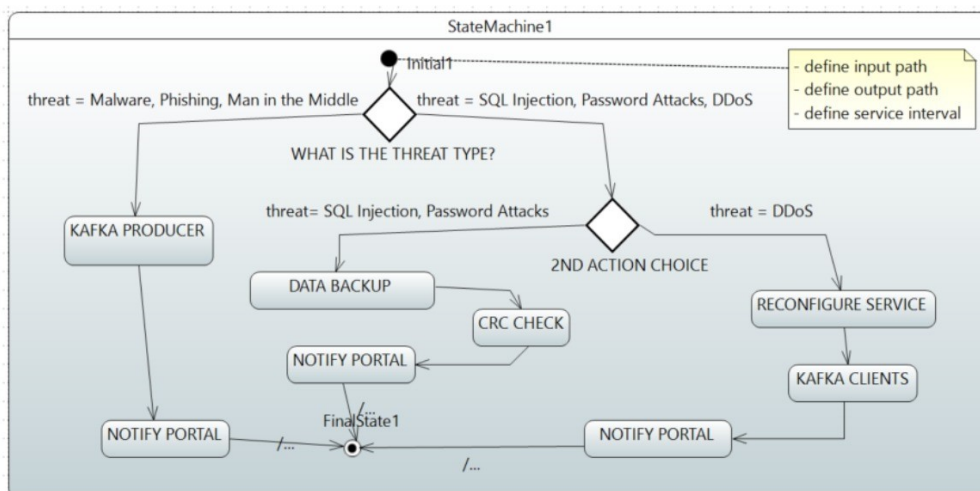
Figure 6: Periodic attestation and remediation for failed attestation workflows

After the initial attestation, the AE performs periodic runtime attestations on the infrastructure and the services in order to monitor their trustworthiness over time, as represented in Figure 6. The infrastructure topology is periodically iterated, and, for each node, the AE performs an attestation in order to verify that all the applications and kernel modules loaded on the host system are authorized and have not been compromised, and that the software executing in each SC on the node is the expected one. Moreover, the DIME kernel module, present in the SCHI, constantly monitors kernel memory in order to detect any tampering with the kernel code or its critical data structures in RAM, such as the system call table, and, as soon as it detects signs of compromise, it sends a runtime alert to the AE. All attestation results are sent to the Threat Intelligence (TI) so that they can be used for further analysis by ML algorithms. If the attestation result contains an integrity failure, the AE also sends a notification to the RS, which is responsible to find a suitable recovery policy based on the type of attestation failure. The RS then interacts with the TI, in particular with its Remediation and Recommendation (RR) component, in order to refine the recovery policy into a MSPL. Then, the RS forwards the MSPL to the SO, which enforces the remediation process accordingly.

2.2.3 Policy design and Personalized Remediation

Remediation policies in WP4 regarding Fault and Breach Management (FBM) are designed using a graphical tool Eclipse Papyrus, as described in deliverable D4.2 and stored as UML files. Using this approach, a policy is automatically exported and integrated into the FBM's Spring State Machine [8], where it awaits to be activated by an outside trigger, in case of PALANTIR a message to appropriate Kafka topic. Personalized Policies are defined as UMLs for both the Incident Response and the Recovery Service subcomponents, and each services implement sits own set of Kafka Topics. Figure 7 outlines an example of a remediation policy for data backup.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	19 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final



a) FSM policy designed in Papyrus tool

```
<?xml version="1.0" encoding="UTF-8"?>
<uml:Model xmi:version="20131001" xmlns:uml="http://www.eclipse.org/uml2/5.0.0/UML" xmlns:xmi="http://www.omg.org/spec/XMI/20131001" xmi:id="_2L_zYKN8EeeHApNcf-4Pw" name="two_separate_state_machines">
  <packagedElement xmi:type="uml:StateMachine" xmi:id="_6ZqGoKN8EeeHApNcf-4Pw" name="StateMachine1" visibility="public">
    <region xmi:type="uml:Region" xmi:id="_710XwKN8EeeHApNcf-4Pw" name="Region1">
      <ownedComment xmi:type="uml:Comment" xmi:id="_cdPBsPCREeyVlrl9IFvbng" annotatedElement="_PE3xcKN9EeeHApNcf-4Pw">
        <body>- define input path&#xD;
          - define output path&#xD;
          - define service interval</body>
      </ownedComment>
      <transition xmi:type="uml:Transition" xmi:id="_S9PKIKN9EeeHApNcf-4Pw" source="_PE3xcKN9EeeHApNcf-4Pw" target="_vFvT4FQPEeyfC6ZISXoRFQ"/>
      <transition xmi:type="uml:Transition" xmi:id="_6VjE4KOLEeeHApNcf-4Pw" kind="local" source="_CITSYKN9EeeHApNcf-4Pw" target="_HlcGYKN9EeeHApNcf-4Pw">
        <effect xmi:type="uml:FunctionBehavior" xmi:id="_5rD2EKOOEeeHApNcf-4Pw" name="sysName">
          <language>bean</language>
          <body>sysName</body>
        </effect>
      <transition>
        <transition xmi:type="uml:Transition" xmi:id="_eT1RYFQZEeyfC6ZISXoRFQ" source="_r6z-EFQREeyfC6ZISXoRFQ" target="_9zTpoFQREeyfC6ZISXoRFQ"/>
        <transition xmi:type="uml:Transition" xmi:id="_dSu7wPCQEEyVlrl9IFvbng" source="_F8i3EFQREeyfC6ZISXoRFQ" target="_JeiYgPCQEEyVlrl9IFvbng"/>
        <transition xmi:type="uml:Transition" xmi:id="_ki3hgPCQEEyVlrl9IFvbng" source="_JeiYgPCQEEyVlrl9IFvbng" target="_fy-q8PCQEEyVlrl9IFvbng">
          <effect xmi:type="uml:FunctionBehavior" xmi:id="_SSkvEPCSEyVlrl9IFvbng" name="sysName2">
            <language>bean</language>
            <body>sysName2</body>
          </effect>
        </transition>
        <transition xmi:type="uml:Transition" xmi:id="_mOq8APCQEEyVlrl9IFvbng" source="_fy-q8PCQEEyVlrl9IFvbng" target="_HlcGYKN9EeeHApNcf-4Pw"/>
        <transition xmi:type="uml:Transition" xmi:id="_3KBC0PCQEEyVlrl9IFvbng" source="__QX5gKN8EeeHApNcf-4Pw" target="_CITSYKN9EeeHApNcf-4Pw"/>
        <transition xmi:type="uml:Transition" xmi:id="_BHv3QPCREeyVlrl9IFvbng" name="threat = SQL Injection, Password Attacks, DDoS" source="_vFvT4FQPEeyfC6ZISXoRFQ" target="_MB9WIFQREeyfC6ZISXoRFQ"/>
        <subvertex xmi:type="uml:State" xmi:id="__QX5gKN8EeeHApNcf-4Pw" name="KAFKA PRODUCER"/>
        <subvertex xmi:type="uml:State" xmi:id="_CITSYKN9EeeHApNcf-4Pw" name="NOTIFY PORTAL"/>
        <subvertex xmi:type="uml:FinalState" xmi:id="_HlcGYKN9EeeHApNcf-4Pw" name="FinalState1"/>
        <subvertex xmi:type="uml:Pseudostate" xmi:id="_PE3xcKN9EeeHApNcf-4Pw" name="Initial1"/>
        <subvertex xmi:type="uml:Pseudostate" xmi:id="_vFvT4FQPEeyfC6ZISXoRFQ" name="WHAT IS THE THREAT TYPE?" kind="choice"/>
        <subvertex xmi:type="uml:State" xmi:id="_F8i3EFQREeyfC6ZISXoRFQ" name="DATA BACKUP"/>
        <subvertex xmi:type="uml:Pseudostate" xmi:id="_MB9WIFQREeyfC6ZISXoRFQ" name="2ND ACTION CHOICE" kind="choice"/>
        <subvertex xmi:type="uml:State" xmi:id="_WVSAEFQREeyfC6ZISXoRFQ" name="RECONFIGURE SERVICE"/>
        <subvertex xmi:type="uml:State" xmi:id="_r6z-EFQREeyfC6ZISXoRFQ" name="KAFKA CLIENTS"/>
        <subvertex xmi:type="uml:State" xmi:id="_9zTpoFQREeyfC6ZISXoRFQ" name="NOTIFY PORTAL"/>
        <subvertex xmi:type="uml:State" xmi:id="_JeiYgPCQEEyVlrl9IFvbng" name="CRC CHECK"/>
        <subvertex xmi:type="uml:State" xmi:id="_fy-q8PCQEEyVlrl9IFvbng" name="NOTIFY PORTAL"/>
      </region>
    </packagedElement>
  </uml:Model>
```

b) an UML representation

Figure 7: an example of a personalized policy, (a) the FSM representation and (b) The UML snippet

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	20 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

The definition of the FSM in the Eclipse Papyrus [3] tool with available Papyrus nodes is presented on the Figure 7a. This is an example of the policy creation for the backup service. Over the Portal interface, the user defines the input path for the data directory, which need to be stored on a secure backup server. User should define the output path, which is the directory on the backup server where data will be stored. And the last user option would be to define the interval of the backup. This will be the periodic time of the service execution. Figure 7b outlines the UML representation of the FSM diagram. UML is generated in the Eclipse Papyrus tool after the FSM model is saved. The generated UML model can be directly used to initialize Spring State Machine and executed on the specific trigger from by rest of the PALANTIR platform, using the Kafka Messaging protocol.

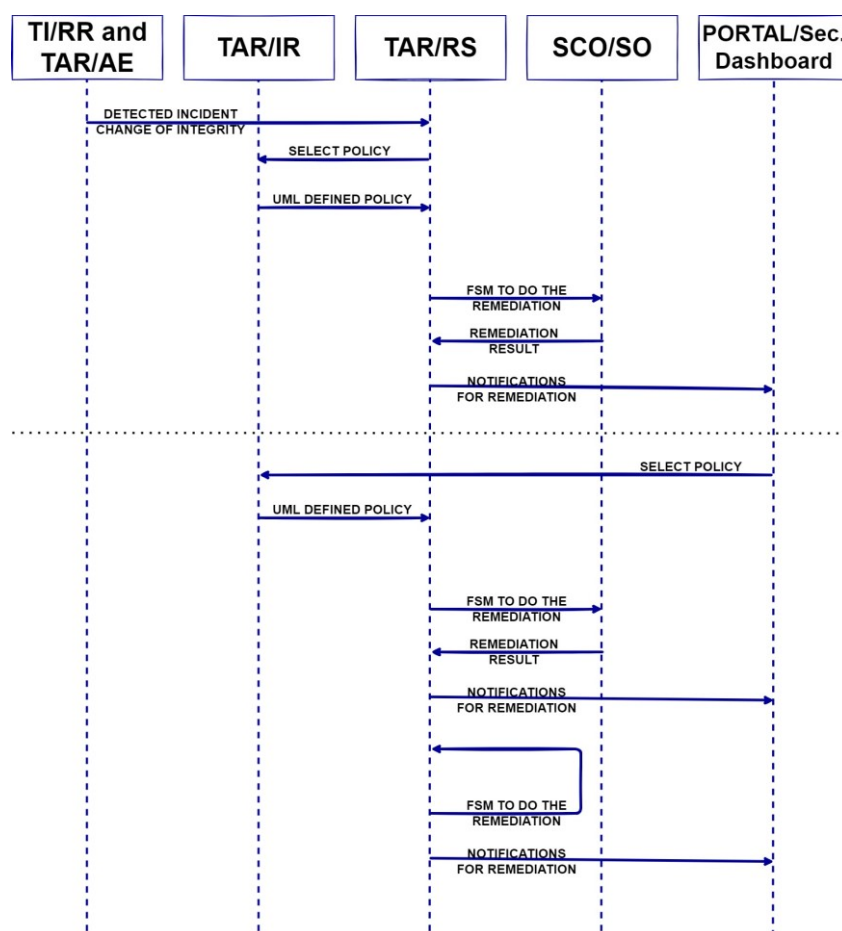


Figure 8: The initial design Policy design and Personalized Remediation workflows

Figure 8 outlines the two workflows used to integrate IR and RS subcomponents into the PALANTIR platform. In the upper part of the workflow, we can see the TI/TAR triggering the selection of policy from the IR/RS subcomponent of the FBM. Based on the type of payload that arrives to the IR/RS subcomponent, the adequate remediation policy is selected from the FBM. The selected policy, in the form of the UML, is then fed to the IR/RS subcomponent, which executes the FSM that is built from the provided UML. Remediation steps provided in the FSM are executed one after another. In one of the steps, IR or RS can ask for the enactment of remediation action from the SO and retrieve the results. In the end, the notification is sent to the Portal from IR or RS. In the second workflow, a user from the Portal is selecting the policy from the FBM. After the policy is selected, the IR or RS runs the needed FSM that can call for the action from the SO and notify about the done remediation or do the remediation withing it's domain and send the notification to the user in the Portal Dashboard.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	21 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

The RR tool in the TI component performs the following workflow from the definition of the reaction policy (aka mitigation recipes) to the response to threats reported by TCAM. It should be noted that RR recipes focus on enforcing mitigation measures on the managed network while FSM policies are applied on the client infrastructure protected by the PALANTIR platform. The recipes are defined a-priori by the security experts responsible for defining the incident responses. Such recipes must contain:

- a description of the triggering conditions (e.g., the mitigated threat, the information that must be comprised in the threat alert);
- a set of enabling conditions that check if such recipes are actually enforceable in the specific managed network (e.g., the security capabilities needed to deploy the recipes);
- the list of operations that must be performed.

The result is a DB containing all the recipes that can be used in a specific domain.

After a threat is reported by TCAM to the RR, the following steps are performed:

1. given the nature of the threat, the tool searches in the recipe DB the ones suitable to mitigate the threat, checking if all the enabling conditions of the selected recipes are fulfilled;
2. the RR interprets the chosen recipe and adapts the contained operations to the specific landscape of the managed network;
3. the tool translates the operations into reconfiguration commands for the involved security capabilities;
4. the tool sends the reconfiguration commands to the security capabilities and notifies the network administrator through the PALANTIR dashboard.

2.2.4 SC Registration and Onboarding

The SC registration and onboarding procedure serves the SC developer to make available its SC to the PALANTIR subscribers. During this process, the developer consigns the information of the security capability to be registered in the Dashboard interface and uploads the SC package containing its implementation. The SCC collects and stored information specific to PALANTIR specification such as the billing modalities, delivered security features. The SO retrieves the packages and supervises its onboarding in components located in the SCHI, such as the NFVO and the VIMs. In the meantime, the AE periodically evaluates from the SCC any newly registered SC. When some are found, the AE exploits a specific plugin in the SO to collect integrity reference measurement from the on-boarded package to store them and, later, use during the remote attention process.

The SCC enables the registration of new security capabilities. This functionality is exposed via the Dashboard web application, through which the Security Capability developer defines the SC. The input data is verified, and the corresponding software is onboarded through the SO. Figure 9 outlines the initial workflow.

During the first step of the procedure, the developer fills the PALANTIR specific information into the portal. The later interfaces with the security catalogue to proceed with the registration of the new SC entree. The feedback of this operation is provided to the Portal and showed to the user.

The developer can then proceed with the upload of the package containing the SC implementation from the portal. The collected package is sent to the SO, which will then dispatch in the components of the SCHI, i.e., the NFVO and the VIM.

Once the package is effectively on-boarded, the SCHI components communicate feedback to the SO, which sill is communicated to Dashboard to inform the developer of the process.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	22 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

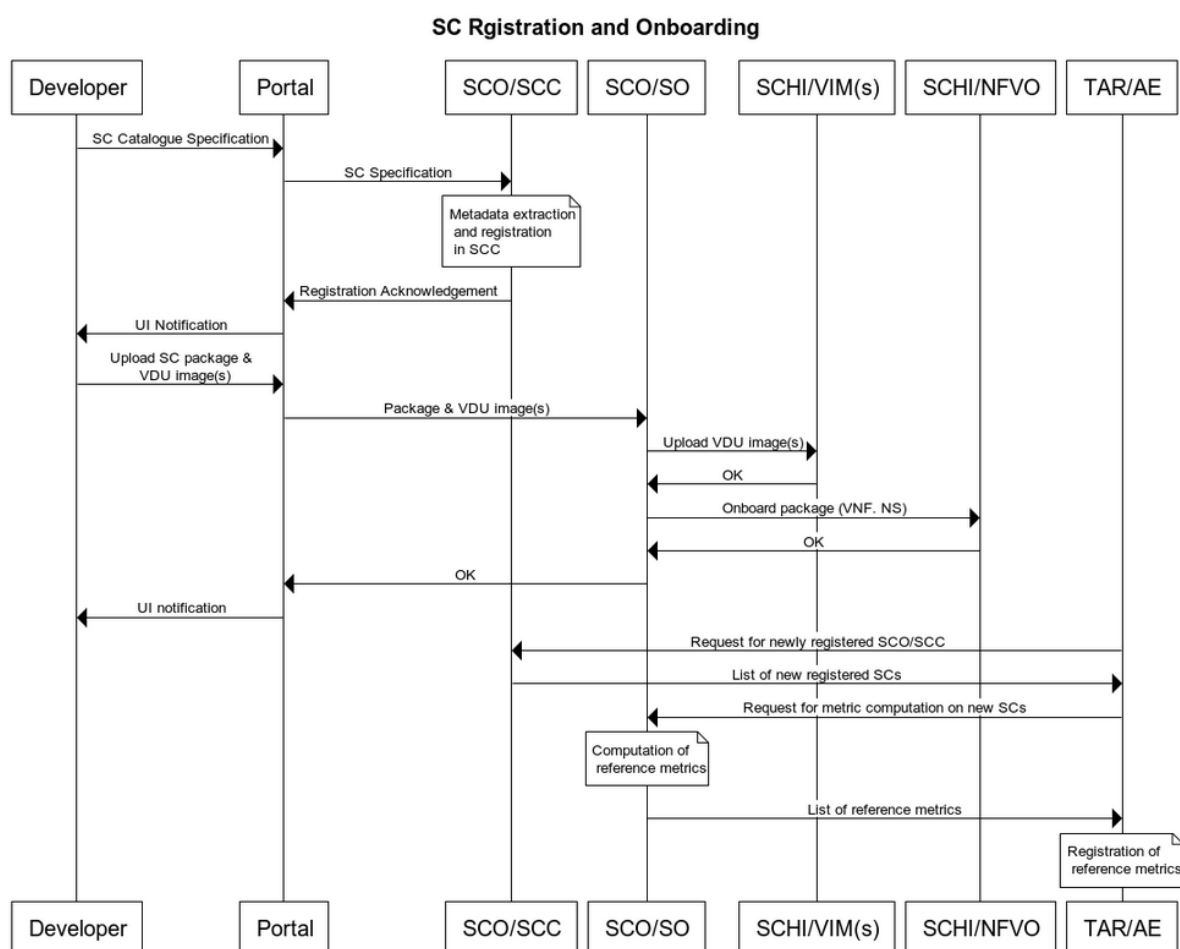


Figure 9: The initial design of the SC Registration and Onboarding Workflow

Concurrently to these steps, the AE continuously pools the SCC for newly registered SCs. When at least one is returned by the SCC, the AE contacts the SO to trigger the reference measurement plugin. This one will access the packages stored in the NFVI and compute the reference metrics for integrity and returns them to the AE. The later them will store them into an internal registry and will later use them as a source of truth for evaluating the integrity of deployed SC instances.

2.2.5 Initial Deployment

The initial deployment procedure enables a PALANTIR operator in deploying adequate SCs to mitigate a security risk identified in the perimeter of a subscriber SME. Specifically, after having used the RAF component, the user can select a set of mitigation to deploy from the portal while being informed of the induced cost. In reality, a mitigation consists in this context in a set of high-level security features to be leveraged by PALANTIR. The SM, the SO and the SCC work conjunctly to identify which implementation suits the customer's context and deploy the SC instance.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	23 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

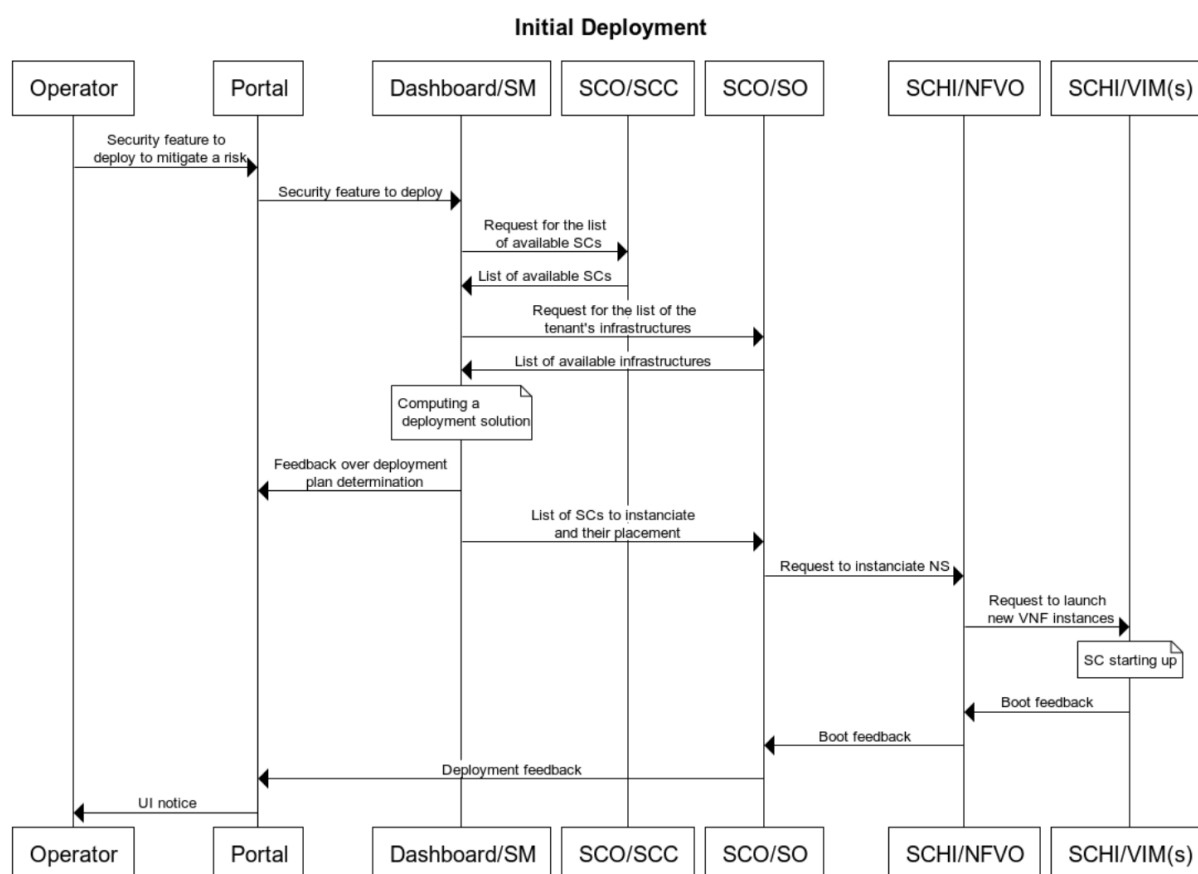


Figure 10: The Workflow for the initial deployment process, current implementation

The precise initial deployment process is represented in the sequence diagram in Figure 10. At the initial stage of the process, the operator indicates through the portal the risk remediation to set up. In practice, these remediations consist of a set of security features to be enforced by PALANTIR platform. This set is transferred to a request to the SM, that will determine adequate security capabilities to instantiate. To that extent, the SM solicits from the SCC the list of the SCs available for deployment and from the SO the list of infrastructures accessible to the PALANTIR customer. Once this process is over, the user is informed of the beginning of the SCs deployment, while the SO is requested to proceed. The latter exploit the NFVO and the VIM(s) to instantiate the SCs in the form VNFs instances. Once the boot process is over, a notification is sent to the operator.

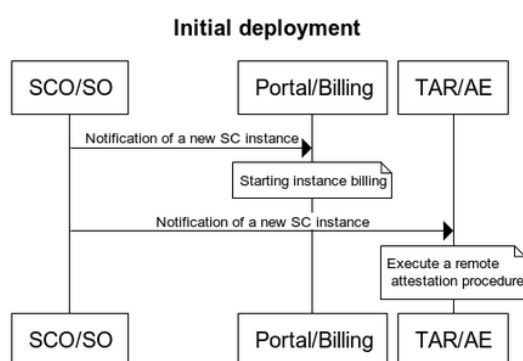


Figure 11: The Workflow for the deployment process to be implemented

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	24 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

As outlined in Figure 11, in future development of the project, the SO will contact the billing framework once the SC has completed its instantiation to initiate the billing. The AE will also be informed of the situation to conduct an initial remote attestation procedure and quarantine the SC if found compromised.

2.2.6 Threats detection & reaction

The threats detection & reaction procedure includes the set of operations performed by PALANTIR components to detect, classify, and mitigate a security threat. This procedure takes place after the *Risk Assessment* and *Initial Deployment* procedures, respectively, have been completed.

The TI component retrieves the input data (e.g., network traffic and system logs from protected assets), analyses it for security threats and finally interacts with the SO to mitigate the identified threat.

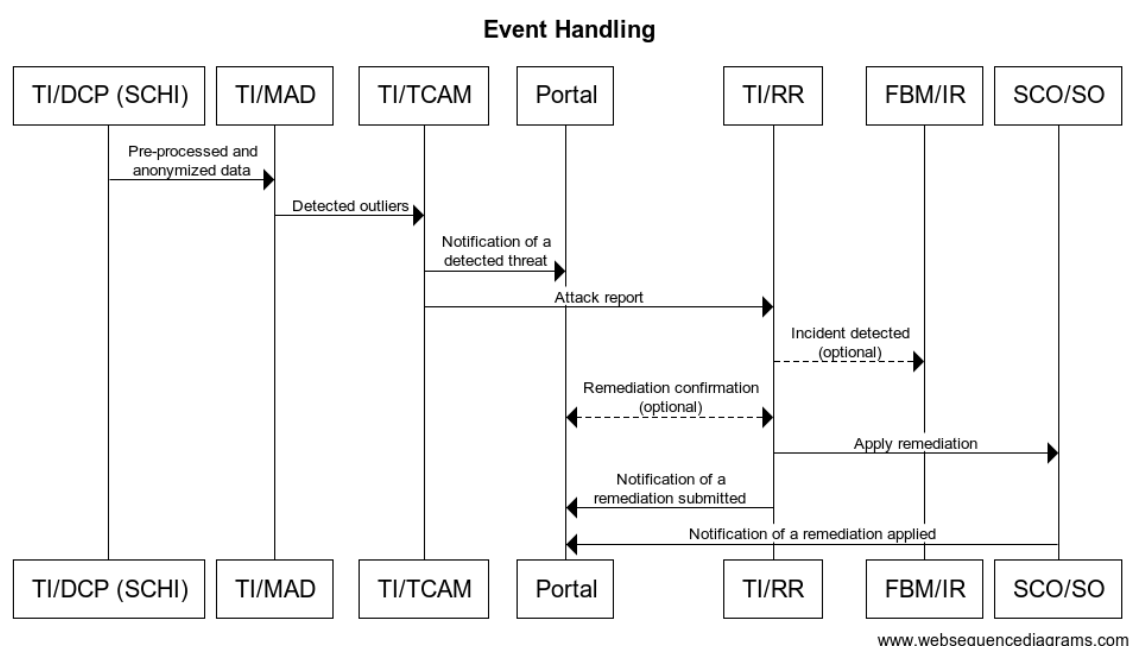


Figure 12: The Workflow for Threats detection & reaction in PALANTIR Platform, part 1

Figure 12 and Figure 13 detail the interactions related to threats detection & reaction across all the components in the PALANTIR Platform. The Distributed Collection and Data Processing (DCP) module, running within the Security Capabilities Hosting Infrastructure (SCHI), collects the input data for the analytics components and pre-processes it (e.g., personal data is anonymized). The pre-processed and anonymized data is then sent to the Multi-Modal Anomaly Detection (MAD) module, which analyses the data to detect abnormal behaviour, i.e., outliers, using DL and ML algorithms.

The detected outliers are aggregated and sent to the Threat Classification & Alarm Management (TCAM) module, which either classifies them as false positives or associates them with a specific threat label. Each predicted label is also accompanied with a confidence score. The user is immediately notified about the detected threats through the Portal and an attack report is provided to the Recommendation & Remediation (RR) module. This module generates the correct remediation strategy to address the specific type of threat just identified. For some types of threats, the remediation requires operations which go beyond the (re)configuration of a Security Capability, and thus, the RR might involve the Incident Response (IR) component part of the Fault and Breach Management (FBM). More details about this last interaction are provided in a dedicated workflow, *Fault Management*, described in Section 2.2.7.

Depending on the user preferences, the RR can optionally involve the user in the loop to confirm the proposed remediations through interaction on the Portal.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	25 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

The RR then requires the SO to apply the remediation to the SCHI. At this point, the RR notifies the user about the submission of the remediation and, finally, the SO acknowledges the correct application of the suggested remediation.

Aiming at a *hybrid* Threat Intelligence combining signature-based methods with analytics ones, the initial steps of the Threats detection & reaction workflow are concurrently executed with the following additional interactions. Once a specific threat label is obtained, the rest of the Event Workflow operations continues as described above.

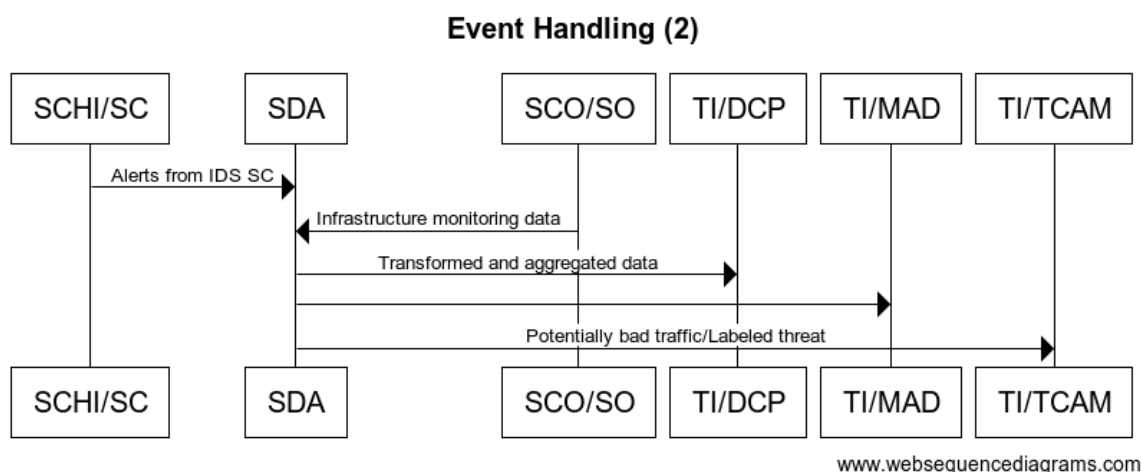


Figure 13: The Workflow for Threats detection & reaction in PALANTIR Platform, part 2

The Semantic Data Aggregator (SDA) is a monitoring framework (currently under development) which aggregates and combines heterogeneous data sources and complements the DCP operations.

Depending on the input data, the SDA interacts with the rest of TI components at different points.

For example, alerts generated by signature-based IDS (e.g., Suricata) with a specific threat label would be directly forwarded to the TCAM's alarm management. On the other hand, alerts reporting a suspiciously bad traffic without a label could be used as a signal of anomaly, but a threat classification step would still be required.

Another potential data source comes from infrastructure monitoring interface of the SO. Anomalies detected on such data on one hand could provide additional context for the detection of a threat, on the other hand could be used to monitor the behaviour of the SCs for attestation purposes (in this case the rest of the operations would fall under the Fault Management workflow described in the Fault and Breach Management section).

2.2.7 Fault and Breach Management

FBM, which consists of the IR and RS, executes predefined policies that are defined in the form of Finite State Machine. In Figure 14 we can see the flow diagram of IR and RS performing remediation steps for detected incidents and the service integrity checks. Fault and Breach Management is executing remediation steps in real-time immediately upon the arrival of the message about the detected changes from the RR or AE component.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	26 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

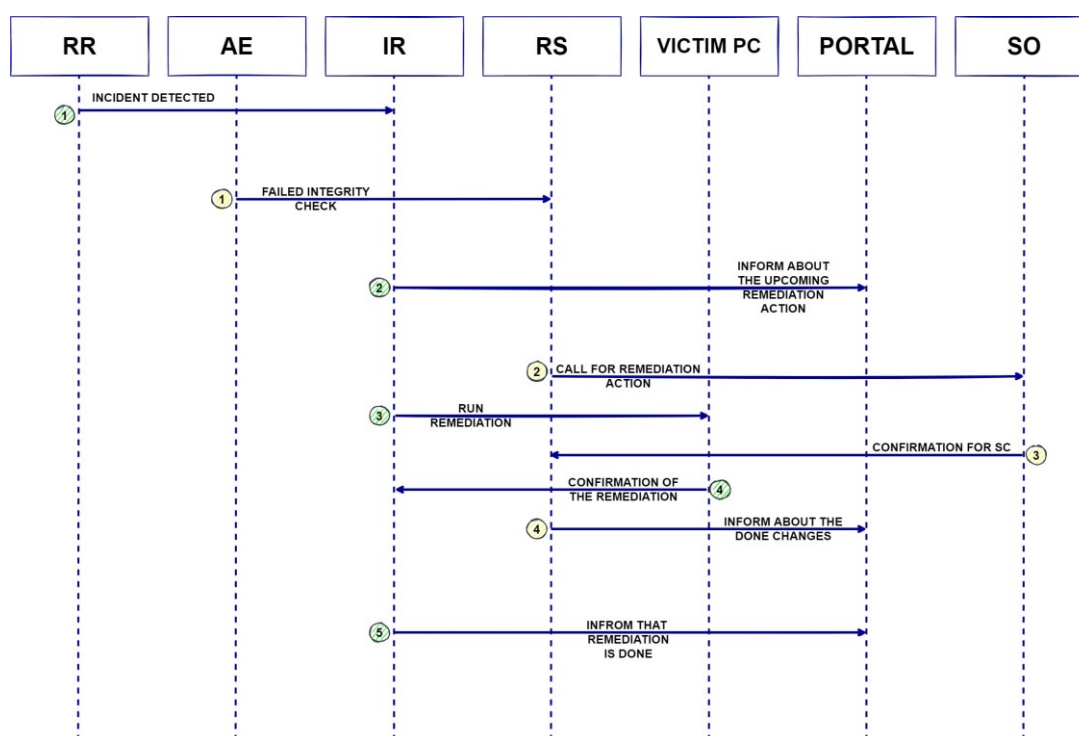


Figure 14: The Initial Workflow designed for the integration of Fault Management

In the diagram in Figure 14, two workflows are executed.

The first workflow with the IR is doing the remediation action once the RR sends a message about the detected incident. It informs the user (e.g. PALANTIR operator) on the portal in the form of a notification about the upcoming remediation, then the remediation is executed. In the end, the user is again notified in the Portal about the remediation success or failure in case of some difficulty.

The second workflow that includes RS is dealing with the result of the integrity check of the services that arrive from the AE. If the integrity check is shown as failed, RS executes the specifically defined FSM, which calls for the SO to do the needed remediation steps. In the end, similarly to the first workflow, the user is informed about the remediation steps done over the Portal.

2.3 PALANTIR Evaluation Methodology, first version

2.3.1 PALANTIR Evaluation Methodology

In this section, we present the KPIs for the key dimensions and the main fields of measurement introduced in the context of the PALANTIR framework. The following list of KPIs is designed to track a diverse set of objectives, related to performance (e.g., threat detection cycle-time improvement), coverage (e.g., variety of attack classes considered) and customer satisfaction (e.g., availability of different billing modules) compared to existing commercial solutions and/or best practices.

Following the ISO 9001:2015 [9] quality management standard, each KPI is based on the S.M.A.R.T. methodology [4], an effective tool to assess the suitability of objectives set, and to measure progress towards the strategic objectives of the project. To this end, the following KPIs are designed to be:

- Specific: clearly defined so that it has a single interpretation
- Measurable: quantifiable and interpreted in terms of size or degree.
- Attainable: representing a realistic goal.
- Relevant: related to the project's context and aligned with its strategic directions.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	27 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

- Time-bound: trackable within a certain timeframe.

2.3.2 PALANTIR KPIs

The full list of KPIs that are elicited for the evaluation of the PALANTIR platform are presented in the below table. Each KPI has a unique ID following the KPI-Txy-n format, where [xy] represents its relevance with a specific task and n its serial number (e.g. K-T31-1 is the first KPI related to T3.1). If a KPI is relevant for a whole work package, the K-WPx-n notation is followed instead.

For each KPI, a description is provided along with expected value(s) denoting acceptable, good and excellent performance accordingly. Each indicator is structured in a way to address at least one specific requirement (as documented in D2.3), indicated by the “Origin” column. Furthermore, a “Justification” column is used to specify the primary source of the respective indicator, while the “Means of validation” column is used to denote the relevant deliverable for the validation of each KPI. Finally, the last column, “Preliminary evaluation”, provides an initial evaluation of each KPI based on the first integrated prototype, which was showcased during the 1st Project Review through a number of live demonstrations (PALANTIR Storylines).

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
K-T31-1	Number of offered SC dedicated to the detection of security events	< 1 1 - 3 > 3	R1.3.20, R1.3.21, R1.3.23, R1.3.24, R1.3.26	DoA / Evaluate the actual amount of SCs of detection type implemented	D3.1, D3.2	3 (Snort IDS SC, Suricata IDS SC, Iptnetflow SC)
K-T31-2	Number of offered SC dedicated to the mitigation of threats and reactive controls deployment	< 1 1 - 3 > 3	R1.3.19, R1.3.22, R1.3.24, R1.3.26	DoA / Evaluate the actual amount of SCs of mitigation type implemented	D3.1, D3.2	Iptnetflow SC
K-T31-3	Actions implemented in order to reconfigure and perform tasks into SC	< 3 3 - 6 > 6	R1.3.4, R1.3.6	DoA, D2.3 / Evaluate the actions implemented and their effectiveness	D3.2	3 (Snort IDS SC - run, add_rule, del_rule)
K-T31-4	SC packages into the SCC	< 5 5 - 8 > 8	R1.3.3, R1.3.7, R2.5.2	DoA / Evaluate the variety of SC packages offered	D3.2	5 (Snort IDS SC, Suricata IDS SC, Iptnetflow SC, Iptables SC, Netflow Collector SC)
K-T31-5	Size of SC image into the SCC related the VNF instances implemented (compressed)	> (1+0.1*n) GB (1+0.1*n) GB - (500+100*n) MB <(500+100*n) MB (n = number of VNF instances)	R1.3.3, R1.3.7	DoA, D2.3 / Evaluate the size of actual SC images into the public Docker Hub repository	D3.2	200-300MB (Snort SC, Suricata SC, Iptnetflow SC)
K-T32-1	Added overhead (in time) to deploying an SC compared to	>= 20% / 25% / 35%	R1.3.1, R1.3.12, R1.3.13	DoA, D2.3 / Quantitative evaluation of the introduced overhead during mitigation	D3.2	3% / N/A / N/A

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	28 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
	bare NF deployment (light / cloud / edge)	11%-19% / 16%-24% / 25%-34% <= 10% / 15% / 25%		response (via new instances)		
K-T32-2	Added overhead (in time) to configuring an SC compared to bare NF configuration (light / cloud / edge)	> 40% / 45% / 60% 30%-30% / 35%-44% / 50%-59% < 30% / 35% / 50%	R1.3.2	D2.3 / Quantitative evaluation of the introduced overhead during mitigation response (via reconfiguration)	D3.2	20% / N/A / N/A
K-T32-3	Success rate in lifecycle management actions	< 90% 90% - 95% > 95%	R1.3.2	D2.3 / Quantitative assessment on the reliability of orchestrated actions	D3.2	100%
K-T32-4	Degree of authentication in the implemented interfaces	Basic Auth Basic Auth or OAuth OAuth mTLS or mTLS	R1.2.9, R1.2.2 R2.3.4	D2.3 / Qualitative evaluation of the security of the authentication procedure	D3.2	N/A N/A N/A
K-T32-5	Degree of conformance to retrieval and exposure of monitoring data from the SCs	Retrieval of health status from the SC (as day action) and exposure via common interfaces (REST and/or Kafka) to the PALANTIR components (e.g., billing system, operational dashboard) (Above) + Retrieval of generic OS metrics from within the SCs and exposure via common interfaces (REST and/or Kafka) to the PALANTIR components (e.g., billing system, operational dashboard) (Above) + Retrieval of custom OS metrics from within the SCs and exposure via	R1.3.14 R1.3.9 R1.3.17	D2.3 / Qualitative evaluation of the amount of monitored data from the SCs	D3.2	N/A Retrieval of monitoring data from SCs and exposure via common interfaces Provision of custom persistence for definition of monitoring data

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	29 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
		common interfaces (REST and Kafka) to the PALANTIR components (e.g., billing system, operational dashboard)				
K-T32-6	Level of integration with attestation procedures	Notification of life-cycle management to the TAR operations on SC instances (i.e., instantiation and deletion) when executed on the NFV orchestrator (Above) + Notification to the TAR of onboarding and deletion of SC packages into the SCC + Notification with expected container runtime data of the SCs in the infrastructure (Above) + Integration of the Attestation Engine plugin into the SO	R1.4.1	D2.3 / Qualitative assessment of the covered data to support the attestation procedures	D3.2	N/A Retrieval and exposure of runtime data from the VIM Retrieval and exposure of runtime data from the SCs in the VIM
K-T33-1	Threats and vulnerabilities identification associated with SME/ME assets	(Similar to the threats and vulnerabilities mitigated by PALANTIR SCs) < 1 1 - 3 > 3	R1.2.11	D2.1 / Evaluate the number of threats and vulnerabilities identified by the RAF	D3.2	N/A The number of threats and vulnerabilities associated to assets in the RAF database.
K-T33-2	Minimise risk assessment and analysis calculation automation level	< 60% 60% - 80% > 80%	R1.2.10, R1.3.28, R1.3.31	D2.1 / Quantitative assessment of the RAF's automation of the risk-related calculations	D3.2	N/A Calculation based on number of discrete steps requiring human intervention

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
						versus total steps
K-T33-3	Number of discrete risk management proposals available from RAF recommendation engine	< 2 2 - 5 > 5	R1.3.31	UCs / Evaluate the capabilities of the RAF engine in proposing recommendations	D3.2	N/A The number of all possible recommendations available in RAF database
K-T33-4	Minimum number of possible cost/benefit analysis proposals presented to the PALANTIR customer	< 1 1 - 3 > 3 (altered in order to be more specific)	R1.3.31	DoA / Evaluation of different offers to customers in terms of cost and benefit	D3.2	N/A The minimum number of possible SC deployment based on cost/benefit analysis
K-T33-5	Time to calculate risk management recommendations	> 5 mins 1 - 5 mins < 1 min	R1.3.31	D3.1 / Constraint from the architecture	D3.2	N/A Calculate the minimum time period between the successful request of recommendations and the provision of the requested list. Time from submit to retrieve the list through the RAF dashboard
K-T34-1	Time to access SC metadata OR provide relevant search/query results	> 30 secs 2 secs - 30 secs < 2 secs	R1.3.3, R1.3.2	DoA, D3.1 / Quantitative evaluation for search procedures in the SCC	D3.2	1.5 sec Time to get response when fetching (meta)data regarding a SC
K-T34-2	Time to register and onboard SC Package	> 1 min for accepted and for state change (success,	R1.3.7	DoA, D3.1 / Quantitative evaluation of onboarding a new SC package into the SCC	D3.2	0.5 sec for accepted

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
		onboarded, failed, etc.) as total time 1 sec - 1 min for accepted and for state change (success, onboarded, failed, etc.) as total time < 1 sec for accepted and < 1 min for state change (success, onboarded, failed, etc.)				7 sec for state change Time to get response when doing a SC registration on the SCC
K-WP4-1	Time to discover critical info & alerts in the security dashboard	> 1min, 60-30sec, <30sec	DOA	DOA / Evaluation of the delay to inform the operator about a security incident	D6.2	<1min Storyline 1
K-WP4-2	% Reduction in Mean Time to Detection of a breach	<20%, 20-60%, >60%	DOA	DOA / Evaluation of performance gain on other breach detection methods	D5.2, D6.2	N/A
K-WP4-3	Number of billing models available	<1,1-3,>3	DOA	DOA / Qualitatively evaluate the flexibility of billing system	D4.1	4
K-T41-1	Indicators showed in the results of analysis	1-3, 4-6, 6-10 (e.g., evaluate the richness of UI for threat description)	R1.1.2	D2.1 / Evaluate the awareness given to the operator	D4.3	N/A
K-T413-1	Number of supported instances streaming the billing data	1-5 (e.g., limited scalability and evaluation), 6-15 (e.g., typical SC usage for a ME & SME, medium evaluation), 16-30 (e.g., targeting SME with a diversified information system, e.g., multi-site, or operating IaaS cloud/MEC resources),	R1.3.18	D2.1 / Evaluate the scalability of the billing framework	D4.3	N/A (No value since the billing framework is still under implementation)
K-T41-2	Number of actions proposed to remediate a reported attack	1-2,3-4,>5 (evaluate the complexity of proposed option for mitigation)	R1.5.4	D2.1 / Evaluate the maximum complexity showed to the security operator as UX	D4.3	N/A (No complete implementation of the UI available)

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
K-T41-3	Number of status item supported to reflect the state of instantiated SC and in Catalogue	1, 2-5, 6-10 (e.g., indicate only if a SC is instantiated, the context of operation, or more detail information).	R1.3.7	D2.1 / Evaluate if PALANTIR UI exposed instance context to the operator	D4.3	N/A (No complete implementation of the UI available)
K-T41-4	Monitor data available through telemetry	1-3 (e.g., only basic infra-related resource), 3-5 (e.g., additional contextual information); >5	R1.3.9	D2.1 / Evaluate how much telemetry data is reported in the dashboard to give awareness to the operator	D4.3	N/A (No complete implementation of the UI)
K-T413-2	Number of supported SC instances by the broker	1-5 (e.g., limited test for broker scalability on SCs) 6-15 (e.g., limited test for broker scalability on multiple infra) 16-30 (e.g., Broker evaluation for scalability on multiple infra & tenant/customers)	R1.1.14	D2.1 / Evaluate how the Dashboard is able to scale with an increasing number of instantiated SC	D4.3	2
K-T41-5	Mean time for sharing a detected threat w/ other potential vulnerable stakeholder	<10" (almost instantaneous) 10"-1' (quick) >1' (slow)	UC2 & UC3	D2.2 / Evaluate the efficiency of the threat sharing framework	D4.3	N/A (no implementation on available yet)
K-T41-6	Number of detected threats for classification	1-3 (limited support for threat scenarios) 4-10 (acceptable support for threat scenarios) >10 (strong support for threat scenarios)	R1.5.3	D2.1 / UX evaluation of the threat description	D4.3	N/A (no implementation on available yet)
K-T41-7	Number metrics reported by the by the real-time information feature	<2, 3-5,>6	DOA	DOA / Report how extensively the real-time information feature covers the SC/SCHI	D4.1, D.4.3	N/A (no implementation on available yet)
K-T41-8	Number of supported in correlation relationships by the dashboard	1,2-3,>3	DOA	DOA / Report how extensively the threat sharing can correlate exchanged TI with the security situation of a stakeholders	D4.1, D.4.3	N/A (no implementation on available yet)

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
K-T41-9	Number of properties supported by the IoC database	1,2-3,>3	DOA	DOA / Evaluate the richness of the IoC data model used in threat sharing	D4.1, D4.3	N/A (no implementation available yet)
K-T42-1	Number of supported remediation scenarios for compromised customer's system	1-2 (limited support for threat scenarios) 3-6 (acceptable support for threat scenarios) >6 (strong support for threat scenarios)	R1.4.3 & 1.3.30	D2.1 / Evaluate how elaborated are PALANTIR mitigations for SMEs resource under attack	D4.4	1 (Storyline 2)
K-T42-1	Mean time to detection of failing health status of a SC	<10" (almost instantaneous) 10"-1' (quick) >1' (slow)	R1.4.4	D2.1 / Evaluate the RS efficiency in detecting failed node	D4.4	N/A (RS feature not implemented yet)
K-T42-3	Number of supported remediation scenarios for compromised SC	1-2 (limited support for threat scenarios) 3-6 (acceptable support for threat scenarios) >6 (strong support for threat scenarios)	R1.4.2	D2.1 / Evaluate the RS in its ability to recover tainted nodes (SC and host)	D4.4	1 (Storyline 3)
K-T43-1	Number of instance properties used for SLA analysis	0, 1-2, >2	DOA	DOA / Evaluate how precise is the evaluation of the SLA	D4.3	N/A (No implementation at the moment)
K-T43-2	Number of properties to use in service matching	1,1-3,>4	DOA	DOA / Evaluate the preciseness of selection criteria of service matching	D4.3	6
K-T43-3	Number of events used in billing computation	<3,4-6,>7	DOA	DOA / Evaluate the extensiveness of the billing model	D4.3	N/A (Billing framework not implemented yet)
K-T44-1	Number of attested SC	1-5 (e.g., limited test for AE scalability on SCs) 6-15 (e.g., limited test for AE scalability on multiple infra) 16-30 (e.g., AE evaluation for	R1.2.8	D2.1 / Evaluate the scalability of AE	D4.4	1 (Storyline 3)

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	34 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
		scalability on multiple infra & tenant/customers)				
K-T44-2	MTTD of compromised node	<10sec (almost instantaneous) 10sec-1min (quick) >1min (slow)	R1.4.1	D2.1 / Evaluate AE performance	D4.4	3sec (Storyline 3)
K-T44-3	MTTR of a compromised node (automatic remediation)	<30sec (almost instantaneous recuperation) 30sec-2min (quick recuperation) , >2min (slow recuperation)	R1.4.2	D2.1 / Evaluate the performance of AE+RS+SO in node recovery	D4.4	1min (Storyline 3)
K-T44-4	Time to escalate a node suspicious state	>1min,60-30sec, <30sec	DOA	DOA / Evaluate detection performance of the AE	D4.4	3sec (Storyline 3)
K-T44-5	Number of tested node properties Hardware Attestation Firmware Attestation OS Attestation SC (Container) Attestation Kernel Runtime Attestation	1,2-3,>4	DOA	DOA / Evaluate the attestation coverage provided by the AE	D4.4	5
K-T51-1	Mean time to ingest, pre-process, anonymize and store data	<10sec (almost instantaneous) 10sec-1min (quick) >1min (slow)	DoA, R1.1.3	DoA, D2.1 / Evaluation of the scalability of the DCP module	D5.1/D5.2	61 sec (w/o pre-processing or anonymisation) Anonymization alone: 1.56 ms/flow
K-T52-1	Mean time to detect an anomaly	<10sec (almost instantaneous) 10sec-1min (quick) >1min (slow)	DoA, R1.1.3	DoA, D2.1 / Evaluation of the scalability of the MAD module	D5.1/D5.2	Less than 1 second/flow for all MAD components
K-T53-1	Mean time to classify a threat	<10sec (almost instantaneous) 10sec-1min (quick) >1min (slow)	DoA, R1.1.3	DoA, D2.1/ Evaluation of the scalability of the TCAM module	D5.1/D5.2	Less than 10 seconds
K-T54-1	Average time to generate a remediation	<10sec (almost instantaneous) 10sec-1min (quick)	DoA, R1.1.3	DoA, D2.1 / Evaluation of the scalability of the RR module	D5.1/D5.2	Less than 1 second

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
		>1min (slow)				
K-WP5-1	Number of data modalities supported	1 2 >2	R1.5.1, R1.5.2	UCs, D2.1 / Evaluation of the coverage of the TI component in terms of data modalities considered	D5.1/D5.2	2 (NetFlow and syslog). Working on 3rd one (considering IDS alerts, required for Hybrid TI)
K-T53-2	Number of types of threats detected and classified	2 3 >3 (e.g., malware, MitM, volumetric, phishing, ransomware, data breach)	R1.5.3, R1.5.5, R1.5.6	UCs, D2.1 / Evaluation of the coverage of the TI component in terms of types of threats	D5.1/D5.2	3 (botnet, data breach/brute force, volumetric)
K-T54-2	Number of types of remediation actions supported	2 3 >3	R1.5.4	UCs, D2.1	D5.1/D5.2	4 (adding nodes, deleting edges, moving nodes, adding rules to the configuration of a security capability)
K-T52-2	Average performance of anomaly detection models	<80% 80-90% >90%	R2.2.1	DoA / Evaluation of the performance of the MAD module	D5.1/D5.2	MIDAS on CIC-IDS2017: ROC-AUC 99.25%, avg Precision: 98.45% AE on CIC-IDS2017 (botnet only): Precision up to 75% GANomaly on CIC-IDS2017 (botnet only): Precision up to 92%

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	36 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
						GANomaly on USTC-TFC2016: Precision up to 99% Isolation Forest on AIT Log: Precision up to 80%
K-T53-3	Average accuracy of threat classification models	<80% 80-90% >90%	R2.2.1	DoA / Evaluation of the performance of the TCAM module	D5.1/D5.2	Random Forest on USTC-TFC2016: overall 10-classes accuracy 99.39% Random Forest on AIT Log: overall 3-classes accuracy: 99.33%
K-WP5-2	Percentage of complex threats –overlooked by traditional IDPS- that were detected using the hybrid TI methods	<60% 60%-90% >90%	DoA	DoA / Evaluation of the performance of the hybrid TI component	D5.2	N/A
K-T54-3	Percentage of recommended remediation actions that led to the mitigation of propagating threats on PALANTIR's protected network	<60% 60%-90% >90%	R2.2.5, DoA	DoA / Evaluation of the performance of the RR module	D5.2	N/A
K-T54-4	Proactive mitigation measures transferred via the PALANTIR threat sharing mechanism led to the automated mitigation of threats in other PoPs	<5 5-8 >8	R2.2.6, DoA	DoA / Evaluation of the performance of the threat sharing	D5.2	N/A
K-WP5-3	Reduction of false positives and negatives with respect to commercial solutions	<15% 15-40% >40%	R2.2.7, DoA	DoA / Evaluation of the performance of the MAD/TCAM modules	D5.2	N/A
K-WP5-4	% Improvement in Mean Time to Detection (using Hybrid Threat Intelligence versus using no intelligence)	<50% 50-65% > 65%	DoA	DoA / Evaluation of the performance of the hybrid TI component	D5.2	N/A

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
K-WP5-5	% Detection rate (false positives versus false negatives measured in AUROC)	<75% 75-85% >85%	DoA	DoA / Evaluation of the performance of the MAD module	D5.1/D5.2	MIDAS on CIC-IDS2017: ROC-AUC 99.25%
K-WP6-1	Hardware resources provisioned for use cases	A. 20vCPU, 80G RAM, 1TB disk B. 10vCPU, 40Gb RAM, 500MB disk More than A between A and B Less than B	DoA, R.1.2.3	Provide resources as required for proper demonstration	D6.1	N/A
K-WP6-2	Number of vertical use cases supported for deployment of demonstrations	>3	DoA	According to DoA three UCs are anticipated	D2.1/D2.2	N/A
K-T62-1	Total traffic generated in order to simulate attacks	30 Mbps (using various traffic profiles according to the scenarios)	DoA, R3.2.3	Provide synthetic traffic to simulate required attack.	D6.2	N/A
K-T62-2	Number of lightweight CPEs to be developed	>1	DoA	Provide a HW CPE to run Palantir components	D6.2	N/A
K-T63-1	Number of Cloud-based and on-premises CPEs to be developed	> 1	D2.4	Provide a HW CPE to run Palantir components	D6.2	N/A
K-T63-2	Type of attacks specific to CRM to be detected and mitigated	>2	D2.4	Detected attempts to exploit the web server vulnerabilities and mitigated by triggering redirects or security forms	D6.2	N/A
K-T63-3	Number of users involved in co-creation and demonstration	> 4	D2.1, D2.2	Number of employees to be involved, minimal 1 type of employee per actor (e.g., owner, accounting, and at least 1 sale in 2 branches)	D6.2	N/A
K-T63-4	Number of locations to be protected	> 3	DoA, D2.1	Number of employees to be involved, minimal 1 type of employee per actor (e.g., owner, accounting, and at least one salesperson in each of the 2 branches)	D6.2	N
K-T64-1	Number of UEs used for the UC realization	>3	DoA	The number of UEs is limited by the available devices, additional traffic will be simulated	D6.2	N/A

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	38 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

KPI-ID	KPI description	Expected value(s)	Origin	Justification	Mean of validation	Preliminary evaluation
K-T64-2	Number of distinct 5G infrastructures to be used	>2	DoA	NCSRD and TID		N/A
K-T64-3	Number of incidents and attacks to be shared across infrastructures	>3	DoA	detected attacks that are notified and shared to the rest of the 5G infrastructures		N/A

Table 1: PALANTIR KPIs

3. PALANTIR PILOTS

3.1. Pilot 1: Securing private medical practices with lightweight SecaaS

3.1.1 Motivation and Outline

Pilot 1 represents the realization of Use case 1, “*Securing private medical practices with lightweight SecaaS*”. Pilot 1 implements a Lightweight SecaaS for the protection of small businesses from data breaches and ransomware attacks. To this end, the PALANTIR platform will be leveraged in the scope of medical data protection, where relevant activities to safeguard patient data and prevent medical identity theft will be supported. In order to support such pilot and showcase the added value of PALANTIR components, a data leakage scenario will be developed and implemented in a medical practice office to replicate a real-world cybersecurity scenario. Various attack types will be investigated, as to their efficiency and applicability to real world conditions. An indicative set of attacks that will be considered:

- Malware;
- Man in the Middle;
- Ransomware;
- Eavesdropping;
- Spoofing.

The described scenario will be integrated in a pilot deployed in the Athens testbed, where the PALANTIR components will be integrated and will monitor the network. The next step will be to initiate an attack scenario to gain access to the medical data node and start the malicious data transfer. The PALANTIR platform will be able to detect the attack and begin to apply remediation measures, such as application blocking, firewall rule enforcement, etc. The primary goal of this use case is to demonstrate a lightweight cybersecurity solution that can leverage both PALANTIR cloud platform modules that will run remotely, and the local edge modules that will perform the on-site operations, i.e., detection and remediation. Edge operations will receive periodically updated metadata in various forms (weights, models, etc.) that will maintain the platform’s readiness in new attacks, and also provide an efficient lightweight SecaaS solution.

3.1.2 Deployment Model

Pilot 1 will follow the Lightweight SecaaS deployment model. The model refers to standalone devices/services installed at the SME/ME premises, following the model of Customer Premises Equipment (CPE) (i.e., all should be local). The aim of this deployment model is to achieve a low power, low cost, and resource efficient infrastructure so SMEs/MEs can easily replicate and deploy the PALANTIR platform and adjust to the corresponding network conditions. The specific deployment is also suitable for dynamic environments at the edge where the computing capacity is resource constrained and an agile approach on scalability is more fitting to the SME/ME’s requirements.

3.1.3 The Testbed Infrastructure

As described in the previous section, the Pilot 1 will adopt the Lightweight SecaaS deployment model, where most PALANTIR components and services will be installed at the SME/ME's premises. For this purpose, an AIO node has been employed to act as the vCPE hosting the said components and services, as depicted in Figure 15. This node is implemented by a small factor server that can be easily transferred and plugged into the local premises provided by the UC owner.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	40 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

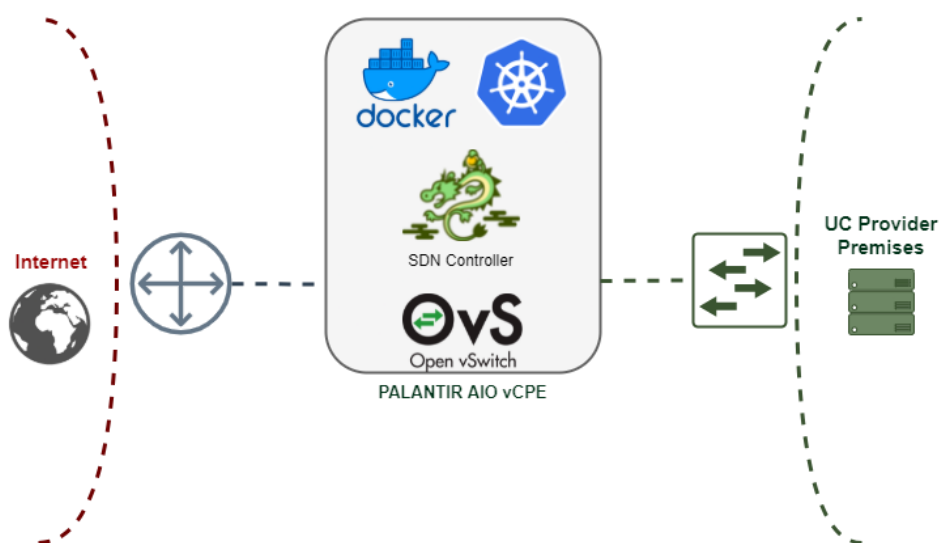


Figure 15: Lightweight SecaaS deployment model in Pilot 1

Figure 16 provides a detailed representation of the vCPE connections, where four physical network interfaces are installed to support the different connectivity requirements of the UC from Figure 15. It is directly connected to the network router, acting as the default gateway of the infrastructure, forcing the traffic to pass through the node. The first interface is reserved for out-of-band management purposes of the node by the system administrator, using a secure protocol connection. The second interface is the one connecting the system to the router, enabling external connectivity. Finally, the two remaining interfaces are used for creating isolated and secure LANs to serve the SME needs. More specifically, one LAN is used for hosting the private infrastructure of the medical practitioner where the critical patient data are securely stored. The second LAN is designed to be used by external users (i.e., the practitioner's clients) via a wireless access point, offering connectivity to the Internet. The nodes from both LANs use the vCPE as the default gateway, allowing the PALANTIR components to monitor all the traffic and apply specific policies where needed.

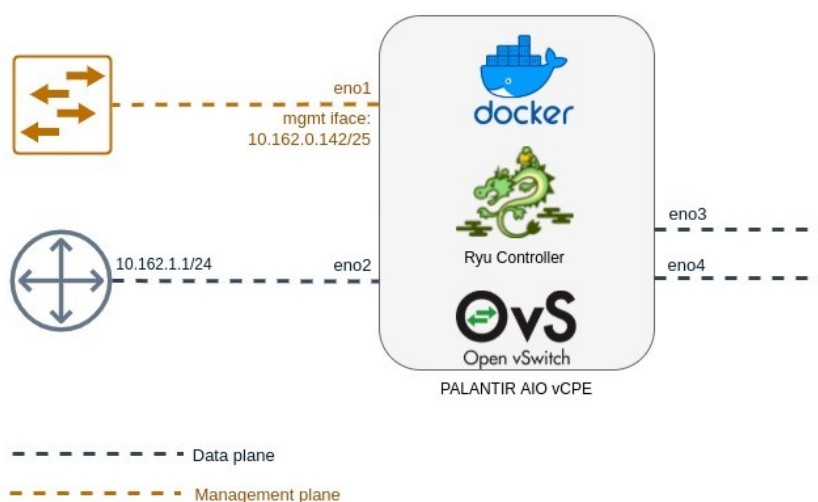


Figure 16: The detailed representation of the vCPE

The following utilities have been installed and configured in order to support the current and future PALANTIR components and allow them to (i) monitor all the ingress/egress traffic of the SME and (ii) apply the necessary remediation actions based on the system conditions:

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	41 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

- **Docker:** Docker is the selected container runtime engine responsible for hosting the containerized PALANTIR components and the deployed SC. It is configured to be used by the Kubernetes cluster when the SO instructs the instantiation of a new SC.
- **Open Virtual Switch (OVS):** OVS is a multilayer virtual switch that enables massive network automation through programmatic extension, supporting standard management interfaces and protocols, such as NetFlow and OpenFlow. OVS is responsible for routing the network traffic to the appropriate network and redirecting the packets to the PALANTIR components for monitoring and inspection purposes.
- **Ryu SDN controller:** Ryu is the selected Software-Defined Networking (SDN) controller for exploiting the OVS programmability features, forcing the network traffic to follow a specific path.
- **Kernel-based Virtual Machine (KVM):** KVM is the employed technology for creating VMs on the vCPE. These VMs are used for hosting the different PALANTIR components and services installed locally on the SME premises, following the Lightweight SecaaS deployment model.

3.1.4 Scope and Threats to be demonstrated in Pilot 1

The Figure 17 Actor Diagram, reported in D2.4 summarizes the intended Pilot 1 operations.

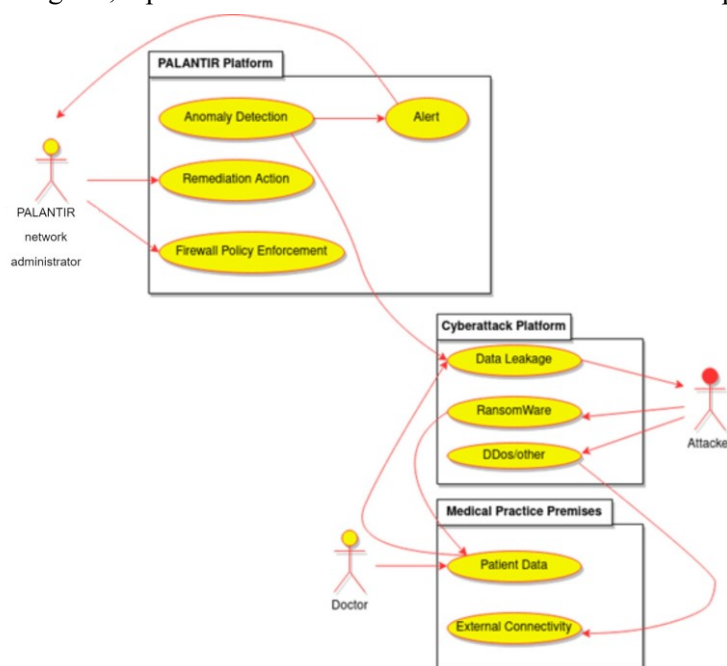


Figure 17: The Actor diagram for Pilot 1 as Reported in D2.4

There are three main stakeholders involved in Pilot 1, **the Doctor** (Healthcare Professional) stores/accesses patient data on-premises (medical practice private data server), the **PALANTIR operator** and the **Attacker**. The goal of the **Attacker** is to manage to get access of the private medical server by disrupting the trusted connection between the healthcare professional and the private server. With this access the attacker can initiate data leakage to a malicious server and/or encrypt sensitive medical data. The PALANTIR Admin leverages the platform's Lightweight SecaaS delivery mode to protect the client's sensitive data. PALANTIR services are deployed on-premises as a Lightweight SecaaS solution and monitor the network traffic. The end-users (e.g. Doctor and PALANTIR Admin), are notified of an ongoing attack and possible encryption of data. A remediation action to block the malicious connection is suggested, e.g., Firewall Policy Enforcement, Backup to secure location, and enforced by the SecaaS components, leading to the disruption of the data leakage attempt.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	42 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

Minimal Pilot specific KPIs to be evaluated within the Pilot are:

KPI-ID	KPI description	Expected value	Justification
K-T62-1	Total traffic generated in order to simulate attacks	30 Mbps (using various traffic profiles according to the scenarios)	Provide synthetic traffic to simulate required attack.
K-T62-2	Number of lightweight CPEs to be developed	>1	Provide a HW CPE to run Palantir components

Table 2: A Draft of PILOT 1 specific KPIs

A more comprehensive list of KPIs is to be reported in D6.2 after the stakeholder workshops.

3.2. Pilot 2: Uninterrupted Electronic Commerce with Cloud SecaaS

3.2.1 Motivation and outline

Pilot 2 represents the realization of Use case 2, “*Uninterrupted Electronic Commerce with Cloud SecaaS*”, as defined under deliverable D2.4. The PALANTIR solution in Pilot 2 provides a holistic cybersecurity protection for a ME in retail and service-oriented setting that uses on-premises equipment and a cloud-based solution that needs to be protected. The strong reliance to online customer services and the lack of security breach technology safeguards provides hackers the opportunity to easily access streams of sensitive corporate and personal data. The PALANTIR solution will be leveraged against attacks targeting disruption of business, getting access to private data of customers, and getting access to sensitive corporate information. In order to support such pilot and showcase the added value of PALANTIR components, diverse attack scenarios will be developed and demonstrated in collaboration with a subcontracted ME who will offer 3 offices located in 3 different cities in Slovenia all daily processing real customer and corporate data. In addition to a replica of local environments and operations, a replica of the e-commerce website and complete suite of cloud-based solutions used by the ME will be made available for replication. Various attack types will be investigated on the corporate infrastructure and the eCommerce platform as to effectiveness of the PALANTIR framework and its applicability to real world conditions. An indicative set of attacks that will be considered:

- Malware;
- DDoS;
- Ransomware;
- Exploits/Injections, including Cross-site-scripting (XSS) and SQL injection attacks;
- Spoofing.

The described scenario will be integrated in an edge pilot deployed in the Maribor testbed, where the PALANTIR components will be integrated and will monitor the network. We will initiate a series of attack scenarios to gain access to i) disturb the connectivity to cloud solution and services and ii) gain access to customer/corporate data on local or cloud infrastructure. We will start with DDoS attacks as a pre-sequel, followed by attacker attempts to exploit the system vulnerabilities of the Cloud CRM (e.g. cross-site scripting, SQL injections, Input Validation Vulnerabilities). The PALANTIR platform will be able to block the attack and begin to apply remediation measures such as, extensive traffic logging (for later inspection), application blocking and request redirection (e.g., automatic rules in *.htaccess*), firewall rule enforcement and backup of data. The primary goal of this Pilot is to exploit PALANTIR’s hybrid deployment mode to secure and to protect a network environment with limited security and multiple managed and unmanaged points from attacks and within the acceptable disruption of critical services (as defined by the owner). To this end, the Incidence Response component will be showcased along with its capacity to design and deploy personalized remediation policies.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	43 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

3.2.2 Deployment model

Pilot 2 will follow the Cloud SecaaS deployment model. The model refers to a hybrid solution where core services are deployed on the cloud and additional standalone services (i.e., clients enabling access to devices) are installed at the SME/ME premises. The aim of this deployment model is to achieve low power, low cost, and resource-efficient infrastructure so SMEs/Mes can easily replicate and deploy the PALANTIR platform on-premises and adjust to the corresponding network conditions. Most SCs are deployed on the CPE (i.e. closer to the client) and minimal on-site deployment is foreseen to ensure access to protected devices and applications. The specific deployment is specifically suitable for environments where business services are distributed between the edge and the cloud.

3.2.3 The Testbed Infrastructure

The test bed infrastructure for Pilot 2 is divided into three main parts. First the Office, which represents the real SME office (e.g. with Windows and Linux-based workstations). These devices are connected to the office router, which is also a gateway to the internet and the Cloud space. Pilot 2 will also leverage agents as monitoring services on the protected infrastructure (e.g. workstations, servers), also exploiting commercial-off-the-self (COTS) and open-source solutions if possible, together with PALANTIR probes which are another type of services that wait for messages from the PALANTIR agents and forward them to the TI. Each device is running the PALANTIR agent as a service that tracks and collects the events and logs and shares those logs to the PALANTIR probe that can be set up locally on the premise or remotely. Figure 18 outlines that PALANTIR probes are deployed locally on the premises to perform monitoring activity. Second part of the testbed represents the Cloud space where the SME's hosts it's mail, web, and data server together with the database. On the Cloud space, PALANTIR agents are deployed and assigned the same task as devices in the SME office. The third part is the Cloud deployment of the PALANTIR Platform which provides the remediation actions for detected cyber threats.

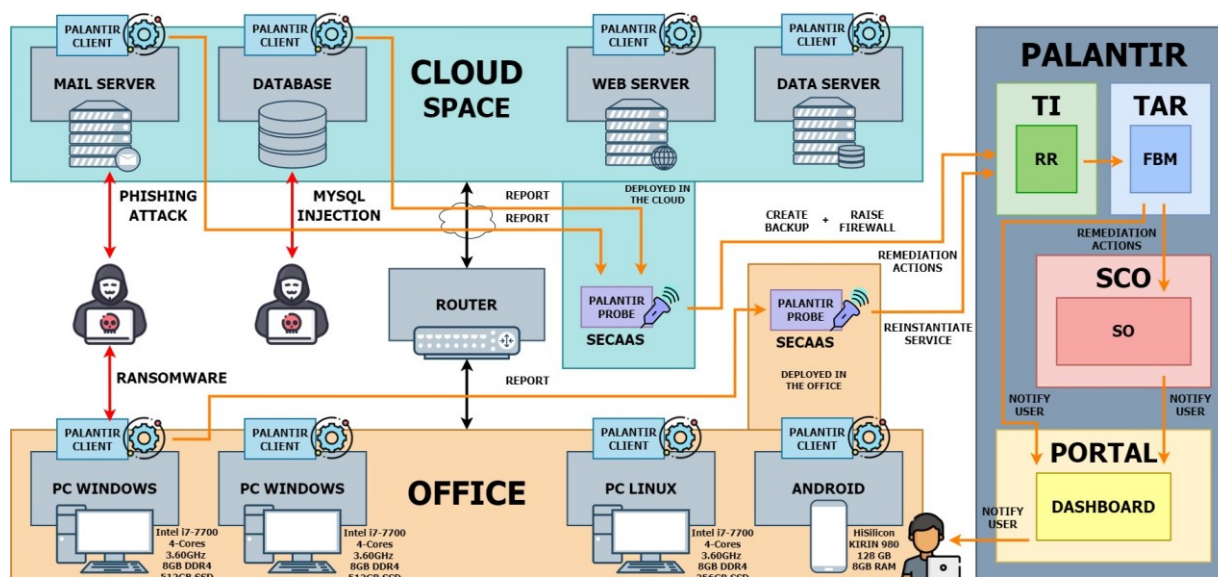


Figure 18: The Pilot 2 Testbed

In case of the attack and detected threats from the PALANTIR Probes which received the data from PALANTIR Clients the PALANTIR Platform apply the remediation steps designed for the type of detected threat. For example, if there is detected the MySQL injection, the FBM component will decide to do the data backup from the database and raise the firewall to prevent the attacker from the access. FBM then informs the user over the PORTAL Dashboard. In case of Ransomware and if there are additional remediation steps, required the FBM will call for remediation actions from the SO and inform the user over the PORTAL Dashboard.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	44 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

3.2.4 Scope and Threats to be demonstrated in Pilot 2

The Figure 19 Actor Diagram, reported in D2.4, summarises the intended Pilot 2 operations.

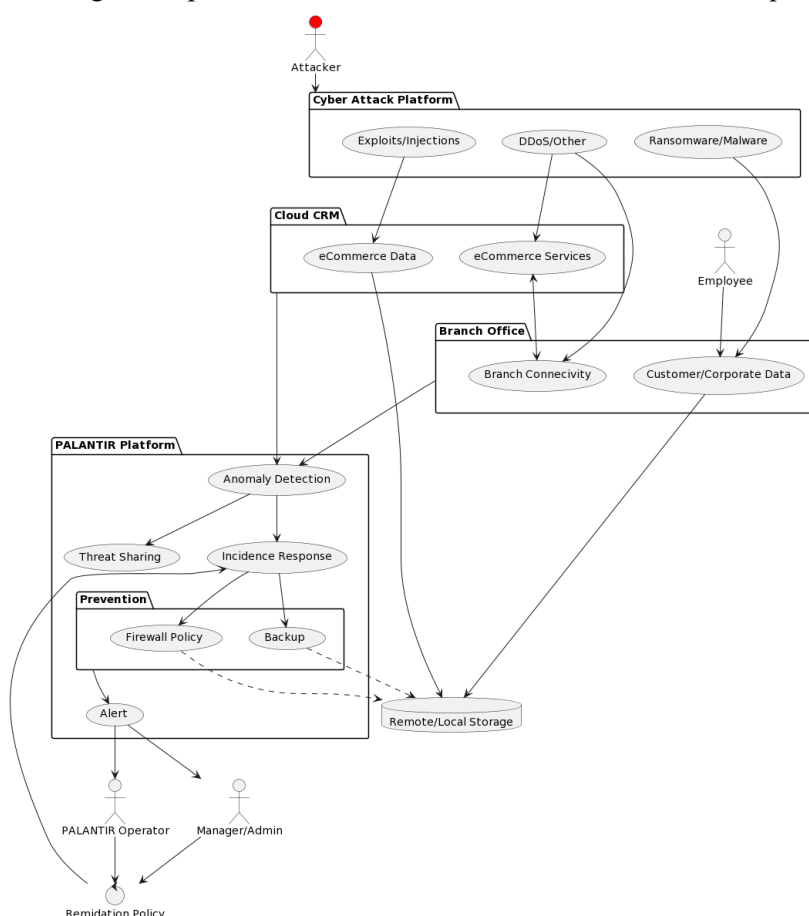


Figure 19: The Actor diagram for Pilot 2 as Reported in D2.4

Four main stakeholders are involved in pilot 2, the **Employee**, who interfaces with Cloud services/APIs and/or customer/corporate data, the **SME's Manager**, who has a high-level access from multiple devices, the **Network Operator** and the **Attacker**. The attacker performs malicious injections/exploits to get access to data or the functionalities of the cloud eCommerce solution and tries to install ransomware/malware to get access to the company infrastructure in order to steal data or hamper services via data encryption. The Network Operator designs a Remediation Policy to best fit the identified risks and priorities of the company using the remediation actions proposed by RAF. The PALANTIR Operator leverages the platform's Cloud SecaaS delivery mode to analyse the network traffic generated by the company's multiple PoPs. The PALANTIR Operator also deploys the on-premises services (i.e., clients and probes). The PALANTIR solution detects the propagating attack and blocks the attacker's access to the network. PALANTIR performs a backup of data (files and database), and stores it in an isolated local or cloud instance for later inspection and post attack restore. After the attack PALANTIR Operator and SME's Manager design a new remediation policy.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	45 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

Minimal Pilot specific KPIs to be evaluated within the Pilot are:

KPI-ID	KPI description	Expected value	Justification
K-T63-1	Number of Cloud-based and on-premises CPEs to be developed	> 1	Provide a HW CPE to run Palantir components
K-T63-2	Type of attacks specific to CRM to be detected and mitigated	= 3 (XSS, SQLi, input validation attack)	Detected attempts to exploit the web server vulnerabilities and mitigated by triggering redirects or security forms
K-T63-3	Number of users involved in co-creation and demonstration	> 4	Number of employees to be involved, minimal 1 type of employee per actor (e.g., owner, accounting, and at least 1 sale in 2 branches)
K-T63-4	Number of locations to be protected	> 3 (Maribor, Ljubljana, Cloud)	Number of employees to be involved, minimal 1 type of employee per actor (e.g., owner, accounting, and at least one salesperson in each of the 2 branches)

Table 3: A Draft of PILOT 2 specific KPIs

A more comprehensive list of KPIs is to be reported in D6.2 after the stakeholder workshops.

3.3. Pilot 3: Live Threat Intelligence Sharing in a large-scale Edge scenario

3.3.1 Motivation and outline

Pilot 3 represents the realization of Use case 3, “*Live Threat Intelligence Sharing in a large-scale Edge scenario*” as defined under deliverable D2.4. This pilot experimentally demonstrates the operational capacity of PALANTIR solution in the 5TONIC and 5GENESIS testbeds. These 5G-enabled testbeds can emulate traffic from multiple SecaaS clients on their edge network as well as parallel complex attacks, in large-scale MEC scenarios. Pilot 3 will incorporate the virtual network infrastructure as well as SDN/NFV infrastructure comprised of high-performance servers for the execution of NFV management software and deployment of SDN controllers. The different elements of the testbed can be flexibly interconnected using OpenFlow switches. 5TONIC provides multi-site capability by incorporating infrastructure and equipment located at TID premises. A part of these labs is the Mouseworld, a configurable generator of labelled network traffic datasets, supporting dynamic network topologies (by means of an NFV infrastructure), experiment scheduling to configure and run predefined scenarios, and dataset labelling from the knowledge derived from the scheduled experiments.

The PALANTIR coordination efforts will be focused on deploying the PALANTIR components on various levels of the utilized virtual networks, while SSE will deploy realistic cyberattack scenarios of propagating attacks (e.g., DDoS, WannaCry ransomware) that will be simultaneously directed to multiple the clients of the PALANTIR solution. In this context, we plan to leverage PALANTIR by:

- Detecting the common threats addressed to multiple clients;
- Publishing the incident to a knowledge sharing platform via PALANTIR’s threat sharing functionalities;
- Retrieving relevant threat intel information in order to produce an appropriate mitigation plan;

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	46 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

- Relaying high-level mitigation policies through the PALANTIR provider to the other SecaaS clients.

3.3.2 Deployment model

Pilot 3 investigates a large-scale 5G scenario Edge SecaaS, where the Communication Service Provider (CSP) deploys the SecaaS on the network edge, following the Multi-Access Edge Computing (MEC) paradigm, an essential building block of 5G deployments. In this scenario, the PALANTIR components, services, and instantiated SCs are pushed to the network edge, closer to the client, rather than congest the CSP core network. As a result, in the edge case, the SecaaS provides protections on multiple "tenants" of the CSP, while it is capable of detecting threats faster and more efficiently.

3.3.3 The Testbed Infrastructure

Figure 20 presents a typical CSP infrastructure consisting of a core data centre, a remote edge location, and the transport WAN connecting the two regions above. On top of the physical infrastructure, various virtual and isolated network slices are created to offer multi-tenancy end-to-end connectivity services, tailored to match the specific requirements of the vertical industries. A set of control-plane components are responsible for the management, orchestration, and distribution of the underlying physical resources, as well as the overall supervision and end-to-end configuration of the deployed services. The infrastructure has been designed to offer cloud-computing capabilities at the network's edge regions that can be used by operators, third parties, and vertical industries, offering ultra-low latency and high bandwidth capabilities for all kinds of applications and services, which can be quickly and flexibly deployed.

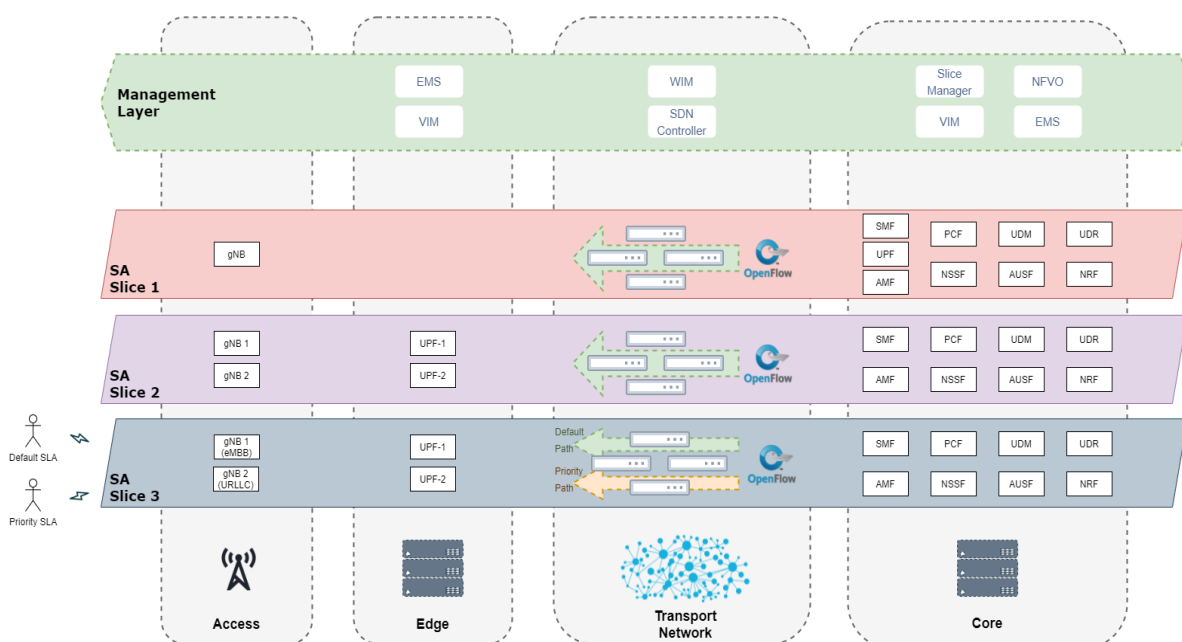


Figure 20: The testbed infrastructure for Pilot 3

As depicted in Figure 21 the 5G edge location consists of a flexible, small scale edge data centre, located close to the 5G new radio components. The edge cloud computing supports containers using the appropriate enabler technology, i.e., OpenStack and Kubernetes clusters. These technologies support the deployment of all the PALANTIR components and services to the edge region of the CSP, as designed by the edge SecaaS deployment scenario. In addition, the SO can force the instantiation of the necessary SCs at the edge, in order to be as close to the client's applications as possible. The CSP

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	47 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

management and orchestration components are responsible for creating the required traffic paths that allow the SecaaS to monitor the traffic and apply the necessary remediation actions to the system.

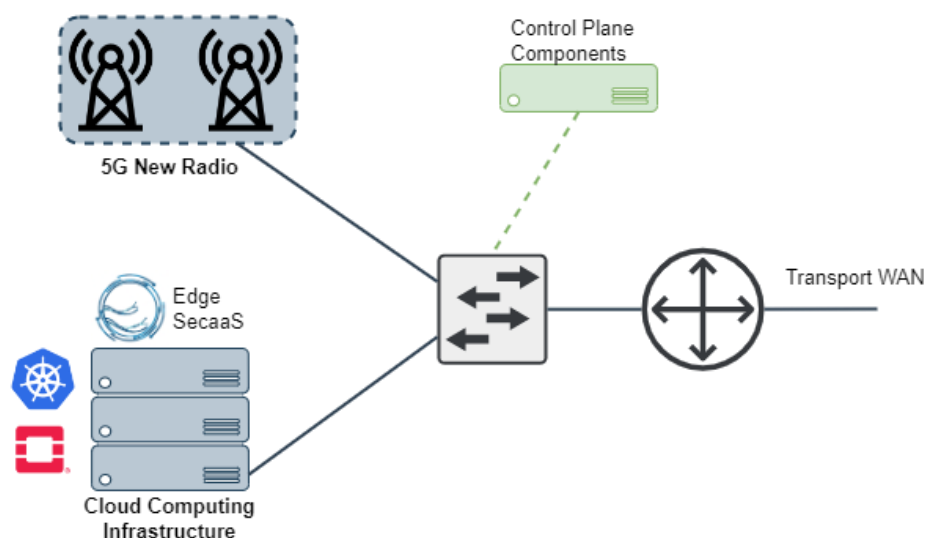


Figure 21: The 5G edge location for Pilot 3

3.3.4 Scope and Threats to be demonstrated in Pilot 3

The Figure 22 Actor Diagram, reported in D2.4, summarises the intended Pilot 3 operations.

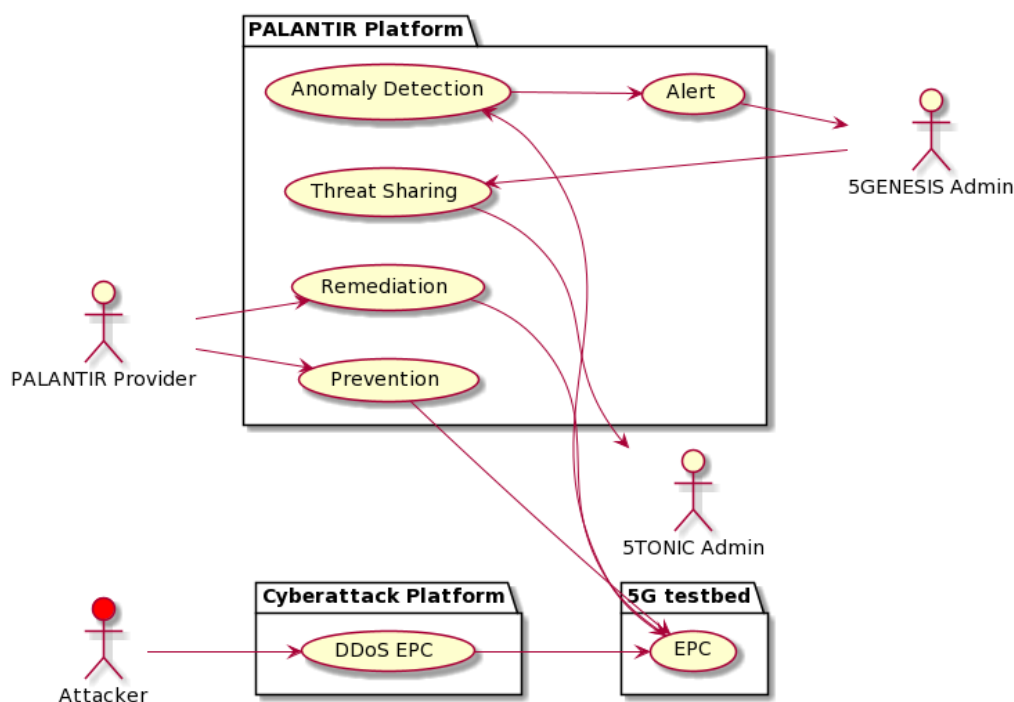


Figure 22: The Actor diagram for Pilot 3 as Reported in D2.4

Four main stakeholders are involved in pilot 3, The **5GENESIS Admin**, the **5TONIC Admin**, as end users administrating the testbeds, **PALANTIR Provider**, responsible for the operation of the PALANTIR platform and an **Attacker** who deploys propagating attacks to the 5GENESIS testbed which is protected by the PALANTIR Edge SecaaS solution. The PALANTIR provider is a CSP that deploys the SecaaS on the network edge following the MEC paradigm, offering an umbrella of

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	48 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

protection to multiple tenants in large-scale edge scenarios. The attack on the 5GENESIS tenant is detected by PALANTIR as an anomaly and an alert is issued to the testbed administrator along with a suggested remediation policy, which is enforced to the current tenant. The threat data is also published to another tenant (5TONIC testbed) via the threat sharing functionality, resulting in a proactive policy enforcement that prevents the further propagation of the attack.

Minimal Pilot specific KPIs to be evaluated within the Pilot are:

KPI-ID	KPI description	Expected value	Justification
K-T64-1	Number of UEs used for the UC realization	>3	The number of UEs is limited by the available devices, additional traffic will be simulated
K-T64-2	Number of distinct 5G infrastructures to be used	>2	NCSRD and TID
K-T64-3	Number of incidents and attacks to be shared across infrastructures	>3	detected attacks that are notified and shared to the rest of the 5G infrastructures

Table 4: A Draft of PILOT 3 specific KPIs

A more comprehensive list of KPIs is to be reported in D6.2 after the stakeholder workshops.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	49 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

4. PALANTIR Testbed and Innovation Labs

This section provides the first initial draft of the manual for PALANTIR Testbed and Innovation Labs to be mentioned and will be finalized under D6.2. This section will be used as the baseline for the PALANTIR workshops with stakeholders, with the aim of allowing them to experiment with the PALANTIR testbed and get accustomed to its usage. It is envisaged that such workshops will funnel requirements and pilot specific KPIs and provide validation feedback from the international community and will contribute significantly to the wider adoption of the PALANTIR results. Using this guide the participating stakeholders will get all the knowledge they need to go forward and develop/experiment on PALANTIR tools and propose innovative applications, use-cases, and further business models, products, and services for the platform.

4.1. Installation guide

4.1.1 Dashboard – Portal

Requirements on the operation environment

The technical requirements for the accommodation of all the portal subcomponents in a single machine are as follows:

- Minimum hardware requirement: A machine with 2 vCPUs, 4GB of RAM, 16 GB of HDD;
- Recommended hardware requirement (for all services in one machine and a relatively small DB size for the deployment): 4 vCPUs, 8GB of RAM, 32 GB of HDD;
- Software environment: The machine has to support the Java Runtime Environment version 11, or a more recent version, node.js and Maven;
- Proven OS: Ubuntu server 20.04 LTS is a proven operating system for the Portal, but any other OS running JRE 11 should suffice;
- Container-based instance management: If the Portal services are to be facultatively managed through Docker-compose, Docker Engine 20.10.16 and Docker Compose 1.29.1 have to be installed.

Setting up hosts, for integration purposes

Make sure that either `/etc/hosts` file on the host machine, *or the DNS* service the machine works with in general, contains the following mappings of domain names to valid IP addresses (name → IP address):

- `sco-scc` → valid SCC IP
- `sco-so` → valid SO IP

This is required in order for APIs of other PALANTIR components to be reachable by the Frontend.

Similarly for Kafka. Make sure that the following domain name mapping is in place:

- `kafka` → valid Kafka broker IP

The above step is necessary for proper operation of the backend.

Note that more such mappings will be required as integration with other components of the platform becomes more complete.

Source retrieval

From a CLI, the source code of the software can be fetched with the following commands:

```
$ git clone https://github.com/palantir-h2020/dashboard-portal.git
```

Then change current working directory to proceed with the remaining tutorial:

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	50 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

```
$ cd dashboard-portal/
```

Parameters

Before moving on, make sure you have set the required environment variables [5] which are needed for running Docker. You can do this by creating a `.env` file and export the environment variables. There is an example environment file named `.env.example` under the parent project's root directory (current) for the backend and accompanying services, and one more under `src/main/dashboard-webapp` for the frontend. The corresponding `.env` file is to be placed in the same directory as `.env.example`.

Portal Services

The services required for the portal to function have to be deployed before the main backend and frontend are. For development and testing purposes, Docker and Docker Compose can be used to deploy all required services, such as Postgres, pgAdmin, Keycloak etc., as shown below:

```
$ docker-compose up
```

Similarly, spinning up the services in production environment:

```
$ docker-compose --profile monitoring up
```

Keycloak (user management)

Once the services are up, go to `http://localhost:8090/auth/` (URL relative to the machine) and login in the Administration Console using the credentials defined in the `.env` file (e.g., admin/palantir).

The realm to select by default is `palantir`. In order to import the default settings of the PALANTIR realm, use the `realm/realm.json` file.

Then go to *Clients* and open the `backend-service` client ID. On the *Credentials* tab generate a new secret and update the value of `KEYCLOAK_CLIENT_SECRET` in `.env` with it.

Finally, go to *Users* and add a user with a *non-temporary* password. Once you save the user, go to *Credentials* tab for the user, to set a password, and *de-select the temporary* switch. The go to *Role Mappings* and add role to the user. Each user must have ONE role! By default, the mapping is `default-roles-quarkus`. Remove that from the assigned roles and add another one. The supported ones are:

- `network_operator`
- `sme_manager`
- `sc_developer`

This way the users are appropriately set up.

Compiling and running backend in development environment

```
$ ./mvnw clean compile quarkus:dev
```

Compiling and running frontend in development environment

Before running the frontend, make sure you've installed the required dependencies needed for the frontend app:

```
$ cd src/main/dashboard-webapp
$ npm install
```

Then in order to compile and run the frontend with hot-reload for development purposes:

```
$ npm run serve
```

Packaging and running the entire Portal application

When it comes to packaging and running the application in a staging or production environment, the front-end application is bundled with the backend.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	51 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

There are two options for packaging and running the application, backend with the frontend server, while in project root directory. In order to run them, the other services which are spun up via docker compose should be up and running:

- Create a jar executable and run it:

```
$ ./mvnw clean package -Pvue
$ java -jar target/portal-${release}-runner.jar
```

The \${release} is the same as in pom.xml

- Create a docker container and run it

```
$ ./mvnw clean package -Dquarkus.container-image.build=true -Pvue
$ docker run --env-file .env --network=host palantir/dashboard-portal:${release}
```

The \${release} is the same as in pom.xml

4.1.2 Dashboard - Service Matching

In this section, we detail how to retrieve, install, and execute the SM component.

Requirements on the operation environment

The technical requirements for the service matching are quite modest. Mainly, they relate to the provision of a machines supporting the java Runtime environment.

- Hardware requirement: A machine with 2 vCPUs, 4 GB of RAM, 10 GB of HDD;
- Software environment: The machine has to support the Java Runtime Environment version 11, or a more recent version;
- Proven OS: Ubuntu server 20.04 LTS is a proven operating system for the SM, but any other one running JRE 11 should suffice;
- Container-based instance management: If the SM instance is to be facultatively managed through Docker-compose, Docker Engine 20.10.16 and Docker Compose 1.29.1 have to be installed.

Source retrieval

From a command line interpreter, the source of the software can be fetched with the following commands:

```
$ git clone https://github.com/palantir-h2020/dashboard-servicematching.git
```

We can then change our current working directory to proceed with the remaining of this tutorial

```
$ cd dashboard-sm/
```

Compilation

The SM requires OpenJDK version 11 and can be built with Maven using the following command at the root of the project:

```
$ mvn compile
```

Testing and package generation

To launch unit tests and generate an executable JAR file, the following command can be used:

```
$ mvn package
```

This will produce an executable at ./target/cmr-VERSION.jar.

Launching the SM

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	52 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

Executing the SM with its default configuration is quite straightforward:

```
$ java -jar target/cmr-VERSION.jar
```

In the default configuration, the CMR expects:

- A PostgreSQL database located locally, containing a `cmr` database, accessible to the user `cmr` with the password `cmr`,
- A kafka instance accessible at `kafka:9092`.

To customize the deployment, you can edit a property configuration file according to the following template:

```
spring.main.banner-mode=off
spring.datasource.url=jdbc:postgres://localhost/cmr
spring.datasource.username=cmr
spring.datasource.password=cmr
spring.datasource.driver-class-name=org.postgresql.Driver
logging.level.root = INFO
kafka.bootstrapAddress=kafka:9092
spring.kafka.consumer.key-
deserializer=org.springframework.kafka.support.serializer.ErrorHandlingDe
serializer
spring.kafka.consumer.value-
deserializer=org.springframework.kafka.support.serializer.ErrorHandlingDe
serializer
spring.kafka.properties.spring.deserializer.key.delegate.class=org.apache
.kafka.common.serialization.StringDeserializer
spring.kafka.properties.spring.deserializer.value.delegate.class=org.spr
ingframework.kafka.support.serializer.JsonDeserializer
```

To use the CMR with the personalized configuration file, you can execute the following command:

```
$ java -jar target/cmr-VERSION.jar --
spring.config.location=YOURCONFIGURATIONFILE.properties
```

Containerisation & production-grade environment

The SM can be containerised into an OCI container using docker-ce with the following command:

```
$ docker build -t SM-SNAPSHOT .
```

If we opt for exploiting the provided PostgreSQL DBMS image, its container can be built in a similar fashion:

```
$ cd postgresql/
$ docker build -t postgresql-cmr .
$ cd ..
```

In this case, it is possible to directly rely on the provided `docker-compose.yml` file to instantiate the SM:

```
$ docker-compose up
```

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	53 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

4.1.3 SecaaS – Security Capabilities

The official GitHub repository <https://github.com/palantir-h2020/sc-secaas> incorporates all information and code related to the implementation of the SecaaS as Security Capabilities (SCs). This repository contains the different descriptor packages for the SC instantiation, the juju charms with the actions available into the specific SC, and the files needed to generate the docker image used by such SC.

In the case of SCs, the minimum/expected requirements to perform a correct execution are dependent of the specific SC instance because the services and frameworks running in each SC can differ in requirements and required resources. For that information, the specific README file into each SC juju charm includes the information related with the requirements that such SC needs to be deployed.

To the actual deployment of SCs, a specific environment with a K8s cluster and the Security Orchestrator is required, as well as the docker images publicly available into a Docker Hub repository. With this environment, the following steps are used to install a Security Capability:

- Clone the repository;
- (If needed) Access to the “deployment-images” folder, generate the docker images needed for the SCs and upload to the Docker Hub repository;
- (If needed) Access to the “juju-charm” folder and generate the SC.charm file to be included into the SC package;
- Access to the “descriptor-packages” folder, upload the SC package with the xNF and NS descriptors and instantiate it into the Security Orchestrator.

Each of the SCs implemented contains the specific information and steps to be deployed into the README file allocated in each SC folder.

4.1.4 Security Capabilities Orchestration - Security Capabilities Catalogue

This section outlines the retrieval, installation, and execution of the Security Capabilities Catalogue. It has to be noted that some requirements and deployment steps will be altered in the final version.

Requirements on the operation environment

The technical requirements for the accommodation of all the portal subcomponents in a single machine are as follows:

- Minimum hardware requirement: A machine with 2 vCPUs, 4GB of RAM, 16 GB of HDD;
- Recommended hardware requirement (for all services in one machine and a relatively small DB size for the deployment): 4 vCPUs, 8GB of RAM, 32 GB of HDD;
- Software environment: The machine has to support the Java Runtime Environment version 11, or a more recent version;
- Proven OS: Ubuntu server 20.04 LTS is a proven operating system for the Portal, but any other OS running JRE 11 should suffice;
- Container-based instance management: If the Portal services are to be facultatively managed through Docker-compose, Docker Engine 20.10.16 and Docker Compose 1.29.1 have to be installed.

Setting up hosts, for integration purposes

The connection to the Security Orchestrator has to be set up, by changing the hosts file of the OS to point to the `sco-so`: On Linux add for example `10.101.41.168 sco-so` to the hosts file or do an equivalent mapping to the name `sco-so`. Alternatively, the `sco-so` domain name must resolve to the appropriate host of the Security Orchestrator.

Source retrieval

From a command line interpreter, the source of the software can be fetched with the following commands:

```
$ git clone https://github.com/palantir-h2020/sco-scc.git
```

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	54 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

We can then change our current working directory to proceed with the remainder tutorial:

```
$ cd sco-scc
```

Parameters

Before moving on, make sure you have set the required environment variables [5] which are needed for running Docker. You can do this by creating a `.env` file and export the environment variables. There is an example environment file named `.env.example` under the parent project's root directory (current directory). The corresponding `.env` is to be placed in the same directory as `.env.example`.

SCC Database Services

The services required for the SCC to function have to be deployed before the main service is. For development and testing purposes, Docker and Docker Compose can be used to deploy all required services, MongoDB and MinIO, as shown below:

```
$ docker-compose up
```

Similarly, spinning up the services in production environment:

```
$ docker-compose --profile monitoring up
```

Running SCC in development mode

In order to compile and run the Security Capabilities Catalogue immediately, use the following:

```
$ ./mvnw compile quarkus:dev
```

The SCC has the Quarkus Dev UI enabled in dev mode, which is available in dev mode only at `http://localhost:8080/q/dev/`. The SCC also has a Swagger UI enabled in dev mode, where the API can be inspected, which is available in dev mode only at `http://localhost:8080/q/swagger-ui/`.

Packaging and running SCC as a Jar file

In order to package the SCC:

```
$ ./mvnw package
```

It produces the `quarkus-run.jar` file in the `target/quarkus-app/` directory. Be aware that it is not an *über-jar* as the dependencies are copied into the `target/quarkus-app/lib/` directory.

The catalogue is now runnable using:

```
$ java -jar target/quarkus-app/quarkus-run.jar
```

Alternatively, if an *über-jar* should be built, execute the following command:

```
$ ./mvnw package -Dquarkus.package.type=uber-jar
```

The catalogue, packaged as an *über-jar*, is now runnable using:

```
$ java -jar target/*-runner.jar
```

Packaging and running SCC as a native executable

A native executable can be created using the following:

```
$ ./mvnw package -Pnative
```

Or, if GraalVM is not installed, you can run the native executable build in a container using:

```
$ ./mvnw package -Pnative -Dquarkus.native.container-build=true
```

The native executable can be run with the following:

```
$ ./target/security-capabilities-catalogue-1.0.0-SNAPSHOT-runner
```

For more information about building native executables, please consult Quarkus [6], as SCC currently uses the Quarkus maven tooling.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	55 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

4.1.5 Security Capabilities Orchestration - Security Orchestrator

The official GitHub repository <https://github.com/palantir-h2020/sco-so.git> hosts the modules that are part of the orchestrator and are partially developed already, namely the northbound API, the database schemas, the life-cycle management, the monitoring and alerting, as well as the package handling.

Two deployment modes are currently provided: a virtual environment (requiring Python's venv), for development; and a micro-service-based virtualisation for production (through Docker). A common deployment script is given to ease the deployment of all modules or selected ones, and where both general and per-module requirements are installed automatically.

The minimum requirements for the Security Orchestrator, at this point in time, are:

- A machine with 2 vCPU, 2 GB of RAM, 7 GB of HDD running in an Ubuntu server 20.04 LTS, where Docker Engine 20.10.16 and Docker Compose 1.29.1 are installed.

Similarly, the expected minimum requirements for the Security Orchestrator in its final form are:

- A machine with 2 vCPU, 4 GB of RAM, 10 GB of HDD running in an Ubuntu server 20.04 LTS, where Docker Engine 20.10.16 and Docker Compose 1.29.1 are installed.

The following steps are required to install the Security Orchestrator and its modules:

- Clone the repository;
- Copy sample configuration files into the final ones. Review the common configuration files under the general folder (i.e., “./cfg”) and, if needed, the module-specific configuration (e.g., “./logic/modules/mon/cfg”);
- Move to the deployment folder and run the specific deployment script (e.g., “./docker-deploy.sh”), selecting the specific module to run or none (to deploy all of them).

Each of the modules in use is referenced from the main README file, where specific instructions for their deployment and examples of usage are provided in their own README file.

4.1.6 Trust Attestation & Recovery - Attestation Engine

The Github repository (<https://github.com/palantir-h2020/tar-ae-polito.git>) is where the core components of the AE from PoliTo are hosted: Trust Monitor, Whitelist Service, IMA patch and a customized version of the Keylime attestation framework. In the following the minimum requirements for the installation:

- 1 VM equipped with 2 vCPU, 4 GB of RAM, 60 GB of HDD, Ubuntu server 20.04 LTS, where Docker Engine 20.10.12 and Docker Compose 1.29.2 have been installed; on this node the following components are run: Trust Monitor, Whitelist Service, Keylime Verifier, Keylime Registrar, Keylime Tenant Webapp;
- 1 bare-metal node equipped with a TPM 2.0 chip, Ubuntu server 20.04 LTS and a kernel version >= 3.10 with the applied IMA patch; this node has to be configured as a worker node for the Security Orchestrator and needs of Docker Engine 20.10.12; here the Keylime Agent component is executed.

The following steps are required to install the Trust Monitor, Whitelist Service and the Keylime framework:

- Clone the repository from Github;
- Install and configure the Keylime framework following the instructions contained in the markdown guide;
- Install and configure the Trust Monitor cloud-native application following the instructions contained in the markdown guide; it relies on Docker Compose, which automates the deployment and configuration of the required containers;
- Install and configure the Whitelist Service following the instructions contained in the markdown guide.

The following steps are required to install the IMA patch and Keylime Agent (Worker node):

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	56 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

- Clone the repository from Github;
- Apply the IMA patch to the kernel and recompile it;
- Check the availability of the TPM 2.0 and the extension of the correct PCR banks (e.g., SHA-256);
- Install tpm2-tss and tpm2-tools, for the interaction with the physical TPM 2.0 chip;
- Install and configure the Keylime Agent following the instructions provided by the markdown guide.

4.1.7 Trust Attestation & Recovery - Fault and Breach Management

The Github repository <https://github.com/palantir-h2020/tar-fbm.git> consists of a Spring State Machine Papyrus component which is used to generate the Finite State Machine as a JAR file of the provided Eclipse Papyrus UML files. Generated JAR file is then executed on demand by other component called Spring State machine UML. Spring State Machine UML provides Kafka and REST (OpenAPI) endpoints over which the prepared JAR file/s can be executed. Spring State Machine is always in the listening state even while executing previously given tasks.

FBM contains IR and RS component. Both IR and RS are Java built-in endpoints that contain and execute specifically predefined policies as FSMs.

The minimum requirements for the installation:

- 1 VM equipped with 2 vCPU, 2 GB of RAM, 32 GB of HDD/SSD, Linux 64-bit.

The following steps are required to install the FBM together with IR and RS:

- Clone the repository from Github;
- Follow the instructions contained in the README.md guide.

4.1.8 Threat Intelligence - Anonymization Service

The GitHub repository <https://github.com/palantir-h2020/ti-as> includes the RESTful anonymization service which has two core functions. The first is the anonymization of an IP address, given an original IP address, returning its obfuscated version. The second is the inverse procedure, the deanonymization of an IP address, given an obfuscated IP address, returning its original version.

The minimum requirements for the installation:

- 1 VM equipped with 2 vCPU, 1 GB of RAM, 4 GB of HDD/SSD, Linux 64-bit.

The following steps are required to install the ti-as component:

- Clone the repository from GitHub;
- The README.md file provides the instructions to install and run the components together with their requirements.

4.1.9 Threat Intelligence - Anonymization Service

The GitHub repository <https://github.com/palantir-h2020/ti-dc> includes the distributed collection component which consists of five modules. These modules are the OpenDistro for Elasticsearch among Kibana, the Elasticsearch Sink Kafka connector, the Registry service, the Collector Load Balancer, and finally, the NetFlow Source Kafka connector.

The minimum requirements for the installation:

- 1 VM equipped with 4 vCPU, 16 GB of RAM, 64 GB of HDD/SSD, Linux 64-bit.

The following steps are required to install the ti-dc component:

- Clone the repository from GitHub;
- The README.md file provides the instructions to install and run the components together with their requirements.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	57 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

4.1.10 Threat Intelligence - Data Pre-processing

The GitHub repository <https://github.com/palantir-h2020/ti-dp> includes the data pre-processing component. This component takes and pre-processes the raw NetFlow data from a Kafka topic. The pre-processed data is written to a new corresponding Kafka topic. This application is implemented in the Spark programming language using the Spark Streaming framework.

The minimum requirements for the installation:

- 1 VM equipped with 4 vCPU, 8 GB of RAM, 16 GB of HDD/SSD, Linux 64-bit

The following steps are required to install the ti-dp component:

- Clone the repository from GitHub;
- The README.md file provides the instructions to install and run the components together with their requirements.

4.1.11 Threat Intelligence - Multi-Modal Machine Learning - Anomaly Detection

The GitHub repository <https://github.com/palantir-h2020/ti-mmml-ad> includes a set of Anomaly Detection modules tailored to two data modalities, i.e., network traffic flows (NetFlow) and system logs. Specifically, it includes three algorithms for the NetFlow modality (AutoEncoder, Isolation Forest and MIDAS) and one for the system logs (Isolation Forest).

The minimum requirements for the installation:

- 1 VM equipped with 32 vCPU, 32 GB of RAM, 64 GB of HDD/SSD, Linux 64-bit

The following steps are required to install the MMML component:

- Clone the repository from GitHub;
- Each folder includes a dedicated README.md file with the instructions to install and run the components together with their requirements (e.g., Docker, Kafka) and a small dataset sample

4.1.12 Threat Intelligence - Remediation Engine

The GitHub repository <https://github.com/palantir-h2020/ti-re> includes the Recommendation and Remediation Engine, which is composed by a set of Python 3 modules.

The minimum requirements for the installation:

- A machine equipped with 4 vCPU, 8 GB of RAM, 60 GB of HDD/SSD, Linux 64-bit.

The following steps are required to install the RE component:

- Clone the repository from GitHub;
- Follow the instructions in the README.md file to deploy and run the Remediation engine as a Kubernetes pod.

4.1.13 Deployment models

The following table presents the positioning of each PALANTIR (sub)component for each delivery model. To facilitate the interpretation of the table the following notation is used:

R: indicates remote deployment (i.e. outside SME/ME infrastructure). This is the preferred deployment option for the Cloud and Edge delivery models, however some parts of the PALANTIR platform may be required to remain closer to the user premises (e.g. to enable network-based remediation activities).

L: local deployment to the SME/ME site. This is the preferred deployment option for the Lightweight SecaaS, however there are exceptions for some (sub)components which remain remote in all delivery models, to enable centralized management (e.g. for security reasons).

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	58 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

Component			Delivery models			Must reach/be reached by...
Acronym	Name	Owner / Maintainer	Lightweight SecaaS (vCPE / PiaB)	Cloud SecaaS	Edge SecaaS (WAN-Edge)	
Dashboard>AD	Dashboard > Accounting Dashboard	UBITECH	R	R	R	SM, Billing and SLA
Dashboard >CM	Dashboard > Cybersecurity Dashboard	UBITECH	R	R	R	Kafka, SM
Dashboard >SM	Dashboard > Service Matching	i2CAT	R	R	R	RAF>RM SCO>SO, SCO>SCC TAR>IR
Dashboard >TS	Dashboard > Threat Sharing	UBITECH	R	R	R	Kafka, TI
RAF>RA	RAF > Risk Assessment	ORION	R	R	R	
RAF>RM	RAF > Risk Management	ORION	R	R	R	SM
TAR>AE	TAR > Attestation Engine	POLITO/HP ELB	L(POLIT O) N/A (HPELB)	R (POLIT O) R (HPELB)	R (POLIT O) R (HPELB)	SCHI, RS, SCO Portal, TI
TAR>FBM	TAR > Fault & Breach Management	SFERA	L	L	R	Portal, TI, AE, SCO
TAR>IR	TAR > Incident Response	SFERA	L	L	R	Portal, TI, AE
TAR>RS	TAR > Recovery Service	SFERA	L	L	R	Portal, SCO
TI>AS	TI > Anonymisation Service	INFILI	L	R	R	TI>DP TI>MmML>T C
TI>DC	TI > Distributed Collection	INFILI	L	L	R	Kafka SC
TI>DP	TI > Data Pre-processing	INFILI	L	R	R	Kafka TI>AS
TI>MmML>AD	TI > Multi-modal Machine Learning > Anomaly Detection	NEC	L	R	R	(TI>DP, TI>MmML>T C) Kafka TI>Storage

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	59 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

TI>MmML>T C	TI > Multi-modal Machine Learning > Threat Classification	NEC	L	R	R	(TI>MmML> AD, TI>RR) Kafka TI>AS Portal TI>Storage
TI>RR (a.k.a. TI>RE)	TI > Remediation & Recommendation (a.k.a. TI > Remediation Engine)	POLITO	L	R	R	
SC	SecaaS	UMU	L	L	R	Kafka SCO>SO
SCO>SCC	SCO > Security Capabilities Catalogue	UBITECH	R	R	R	SCO>SO, Dashboard, SM
SCO>SO	SCO > Security Orchestrator	I2CAT	R	R	R	D>SM Kafka SCHI>MAN O, SCHI>VIM SCO>SCC TAR>AE
Third parties						
SCHI>MANO	SCHI > Management and Orchestration	I2CAT	L	R	R	SC SCHI>VIM SCO>SO
SCHI>MQ	Message bus (Kafka)		R	R	R	TI>DC, TI>DP, TI>MmML SC SCO>SO
SCHI>VIM	SCHI > Virtual Infrastructure Manager (e.g., Kubernetes)	I2CAT/UMU	L	R	R	SC SCO>SO

Table 5: PALANTIR Deployment models

4.2.PALANTIR MVP 1.0, demonstration

4.2.1 Storyline 1: Botnet Attack

In the first storyline, PALANTIR is leveraged to detect botnet activity using flow-level features and to isolate the infected machine.

Figure 23 reports the high-level architecture of the components involved in the first demonstration, and is divided into two main planes:

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	60 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

The **control plane** (i.e PALANTIR SecaaS Logic Plane) consists of the PALANTIR framework component conducting any decision-making process in for the evaluation of a security posture of a system and the selection of the adequate mitigation. In the context of the PALANTIR MVP 1.0, the components of this plane have been deployed on a specific Kubernetes cluster, and features the RAF, SM, SO, SCC, Dashboard, DCP, MAD, TCAM and RR component for the first storyline.

- The **data plane** models the instances security capabilities in charge of enforcing the security decisions issued from the control plane and gathering technical information fuelling this plane. In practice, this plane comprehends a second Kubernetes cluster, located on the customer premise, simulating the settings of the Lightweight SecaaS delivery model, and is expected to interact with a network controller to alter the network topology by, for instance, blocking the access of a botnet node.

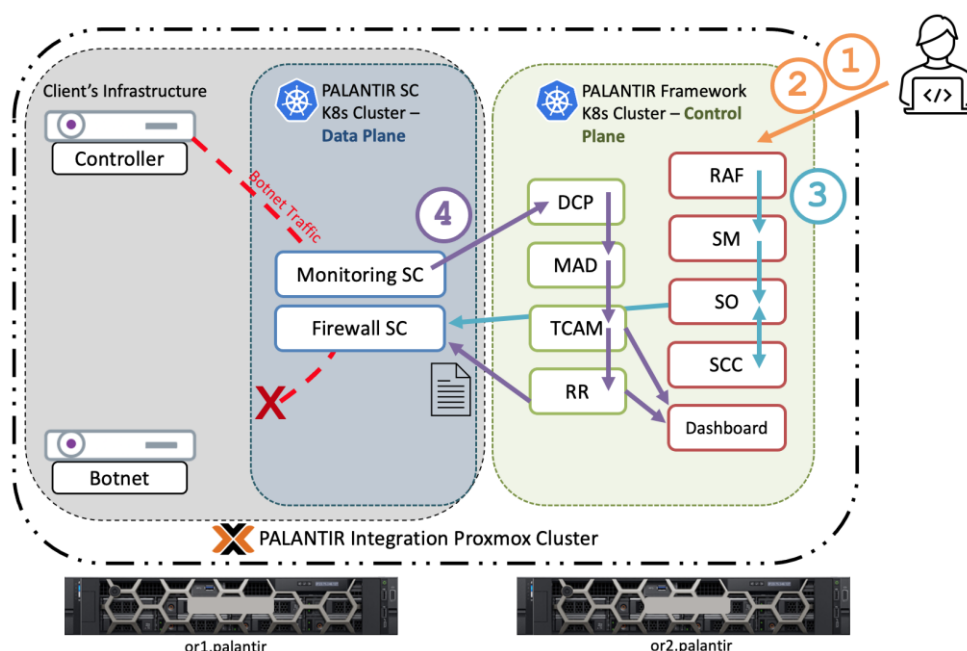


Figure 23: Storyline 1 high-level workflow

4.2.1.1 Risk Assessment - Risk Profile Selection

In the first phase, the Risk Assessment Framework (RAF) provides a simplified view of risk management. The doctor is requested to fill a questionnaire about the infrastructure of his medical centre. A sample of questions is shown in Figure 24. There are 55 questions in total and they are simplified enough to be easily understandable to non-expert users without a deep knowledge about networking. Questions are grouped in 8 chapters (firewall & gateways, secure configuration, software patching, user accounts, administrative accounts, malware protection, risk assessment and awareness of password weaknesses) and the results are weighted based on the potential impact on the business. At the end of assessment, a risk profile, categorized as low, medium, or high, is returned to the user based on the provided answers. Depending on the score, a remediation action is recommended to the user.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	61 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

9. Is this because software firewalls are not commonly available for the operating system you are using?

☒ Yes

☐ No

10. Where you are able to do so, have you removed or disabled all the software that you do not use on your laptops, computers, servers, tablets and mobile phones?

☐ Yes

☐ No

11. Have you ensured that all your laptops, computers, servers, tablets and mobile devices only contain necessary user accounts that are regularly used in the course of your business?

☐ Yes

☐ No

12. Have you changed the default password for all user and administrator accounts on all your laptops, computers, servers, tablets and smartphones to a non-guessable password of 8 characters or more?

☐ Yes

☐ No

13. Do all your users and administrators use passwords of at least 8 characters?

☐ Yes

☐ No

Next

Figure 24: Risk Profile Selection questionnaire sample

4.2.1.2 Risk Assessment - Asset Identification

In this phase, a second questionnaire helps the user to identify the assets in the company. Assets are split in four categories (systems, network, applications, and people). By specifying, for example, the model and the version of the equipment, the exposure to vulnerabilities can be evaluated and remediation actions for minimizing the risk can be suggested to the user. At the end of the questionnaire, the user receives a summary including a set of suggestions to achieve a better security level. For example, as reported in Figure 25, known vulnerabilities for specific software versions are addressed by proposing the download of a security update. In the demonstration, the RAF proposes the deployment of a Security Capability firewall based on the client's feedback (attack surface analysis and asset identification).

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	62 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

Summary

Asset Category	Asset Type	Vendor	Model	Vulnerabilities	Suggestions
System	Server	Dell	PowerEdge R340	CVE-2021-0157	update BIOS to version 2.8.3
System	Workstation	-	-	-	make sure the system gets the updates
System	Laptop	-	-	-	make sure the system gets the updates
System	Backup	Acronis	-	CVE-2020-25593	no patch available yet
Network	Router	Cisco	RV 160	CVSS-2022-20700	You can download the security update through the software downloads page or get the update from Cisco TAC.
Network	Wireless Access Point	Cisco	Catalyst 9130 Access Points	CVE-2020-24586 CVE-2020-24588 CVE-2020-26146	No available patch - Low Risk
Network	Switch	Arista	7280CR3-36S	CVE-2021-28503	The recommended resolution is to upgrade to a remediated software version at your earliest convenience.
Applications	Data	-	-	-	You should use a more reliable resource
People	Training	-	-	unsupervised infrastructure	you could refer to an external company or counsil a specialist

Figure 25: Asset Identification questionnaire sample

4.2.1.3. Service Matching

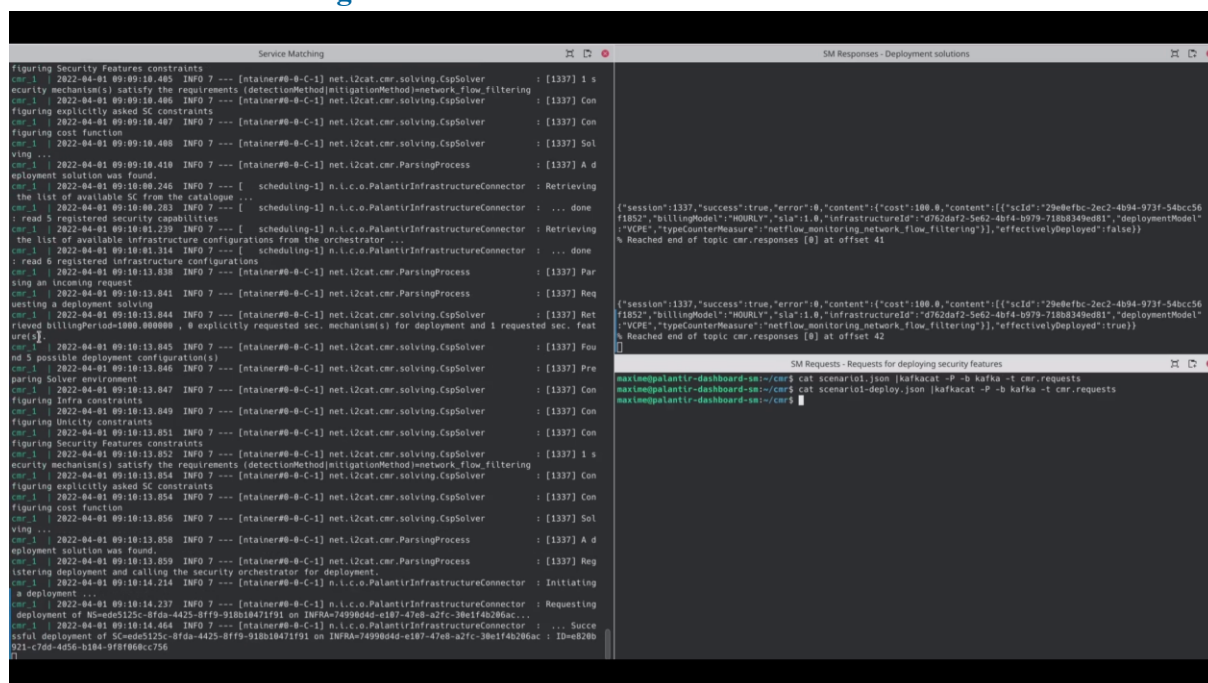


Figure 26: Service Matching Screenshot in Storyline 1

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	63 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

In the third step, the SM subcomponent receives the request for deploying a SC providing firewalling feature. This request is expected to be provided by the RAF component when the project integration will be fully achieved. In Figure 26, the injected request appears in the bottom right of the screenshot.

Then, the SM analyses the properties of the infrastructure accessible to the tenant, as communicated by the SO, the implementations of the SCs available from the catalogue (SCC) and their costs to find a compromise between the billing models and the security features to be leveraged. The several steps are reflected in the left column of Figure 26, and mainly consist in:

- Retrieving the information about the infrastructure properties and available security capabilities,
- Populating a model for the constraint Satisfaction Problem based on the possible solution,
- Enforcing the constraints on the solution to maintain the compatibility between SC and infrastructure, ensuring the capacity of the salter is not exceeded, and accounting the security feature to be deployed,
- Defining a cost function to hierarchise the different possible deployment solution,
- Proceed with the effective solving,
- Structuring the solution.

Therefore, the SM returns a possible solution associating specific SCs to deploy, the infrastructures to host them, and the delivery models to apply along with a price quotation to be reviewed by an operator. This information is technically transmitted over a Kafka channel to be listened.

If the operator agrees with the computed deployment plan, he can trigger an effective deployment and the SC is instantiated by the SO, as confirmed by a notification on the Dashboard.

4.2.1.4. Threat Intelligence

This phase involves all the four modules part of the Threat Intelligence, the Monitoring and Firewall SCs and the Dashboard. Figure 27 reports the seven terminals associated to the different components.

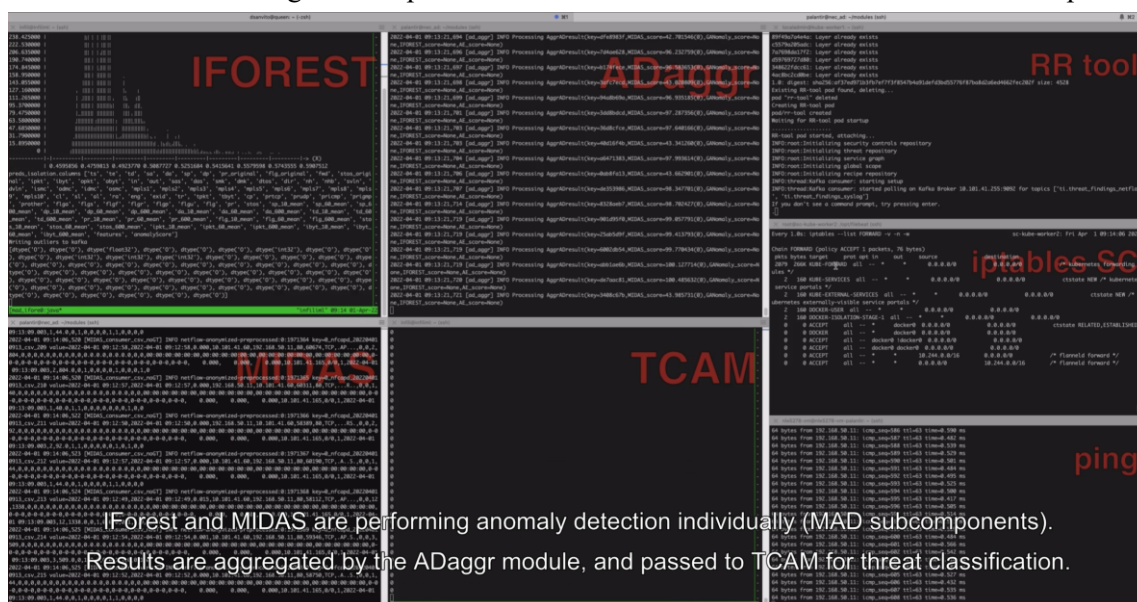


Figure 27: Logging output of all Threat Intelligence components

At the bottom right, a ping command is continuously run from the Controller host to the Botnet host to verify whether the attacker has connectivity towards the victim. Botnet traffic from a benchmark dataset is replayed into the simulated network along with normal traffic. The Monitoring SC forwards the traffic towards the DCP module which pre-processes it (e.g., by anonymizing the IP addresses) before passing it to the MAD module.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	64 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

The MAD includes a set of Anomaly Detection algorithms running in parallel (Isolation Forest and MIDAS, reported on the left) analysing traffic for anomalies starting from flow-level features. Results are passed to the ADaggr module (shown in the upper centre) which aggregates individual results before passing them to the TCAM, reported at the lower centre.

The TCAM is in charge of associating a specific threat label to the detected outliers (using a supervised Random Forest model) and to generate an attack report to be consumed by the RR engine, reported in the upper right corner.

The RR engine computes a remediation measure to deal with the specific threat detected starting from the attack report and from the current network landscape. In the case of the botnet, it suggests a re-configuration of the Firewall SC by adding a couple of policies to block the traffic among the Controller and the Botnet. The terminal on the middle right reports the set of rules currently installed in the iptables Firewall SC.

As soon as the policies are added, iptables reports an increasing number in packets and bytes counters associated to the newly installed rules, confirming that the block mitigation policy is matched by the traffic. A further confirmation can be found at the bottom right corner: the interruption of the ping outputs acknowledges that the attacker has no longer connectivity towards the victim once the remediation has been put in place. The user is informed in real time about the threats and remediations through the Portal.

Figure 28 and Figure 29 show the notifications received by the user when a threat is detected and once the generated remediation has been correctly applied in the network.

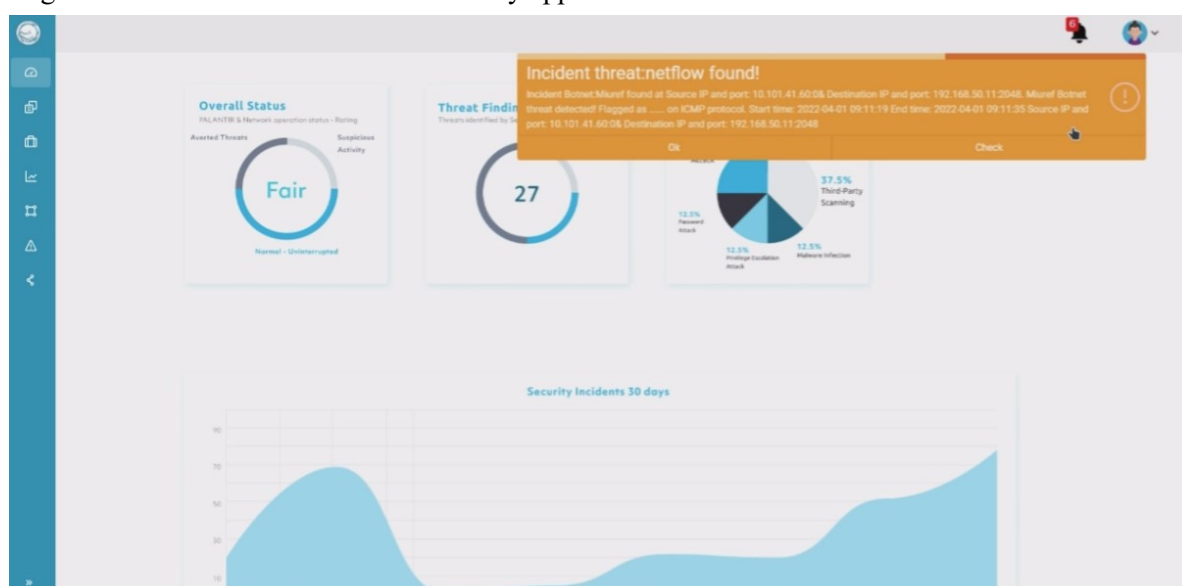


Figure 28: Dashboard notification after a new threat is detection

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	65 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

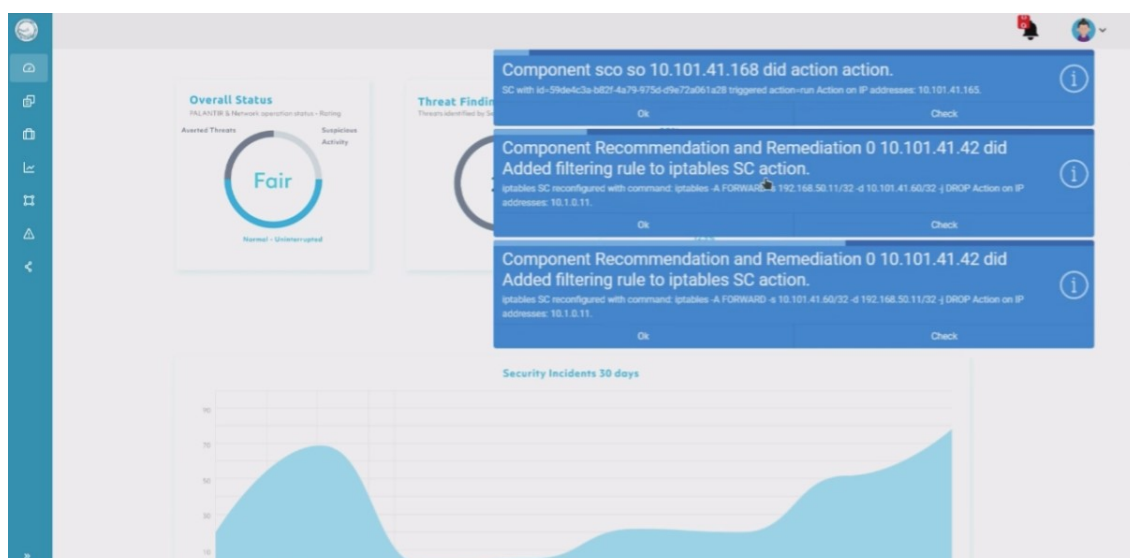


Figure 29: Dashboard notifications after the remediation has been correctly applied

4.2.2 Storyline 2: Data Breach Detection and Recovery via AI-based Log Analysis

In this scenario, PALANTIR is showcased as an end-to-end SecaaS solution which monitors the logs of critical enterprise infrastructure in order to successfully detect and recover from a data breach attempt. The scenario involves the Dashboard, two Security Capabilities (for log monitoring and firewalling respectively), the Data Collection and Pre-processing (DCP), Multimodal Anomaly Detection (MAD), Threat Classification & Alarm Management (TCAM), Recommendation and Remediation (RR) subcomponents of the Threat Intelligence (TI) component and the Incident Response (IR) subcomponent of the Fault and Breach Management (FBM) component. The aforementioned subcomponents are depicted in Figure 30 below:

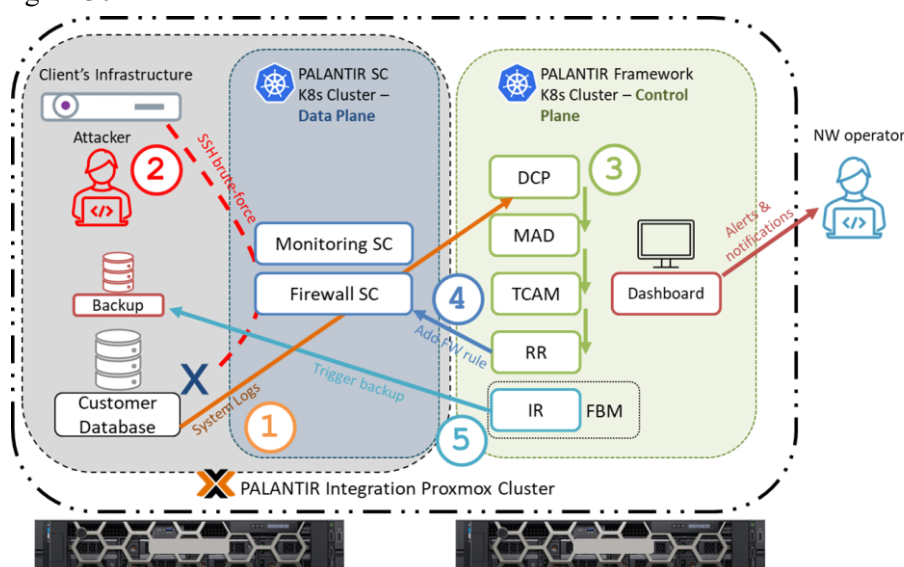


Figure 30: Infrastructure topology for Storyline 2

The scenario comprises the following steps:

1. **Live log monitoring:** System logs from a customer server (database) containing sensitive data are periodically transferred to the DCP. MAD is tasked with performing anomaly detection based on complementary ML models.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	66 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

2. **Data breach simulation:** An SSH brute-force attack (simulating a data breach attempt) is performed live on the database server protected by PALANTIR (Figure 31).

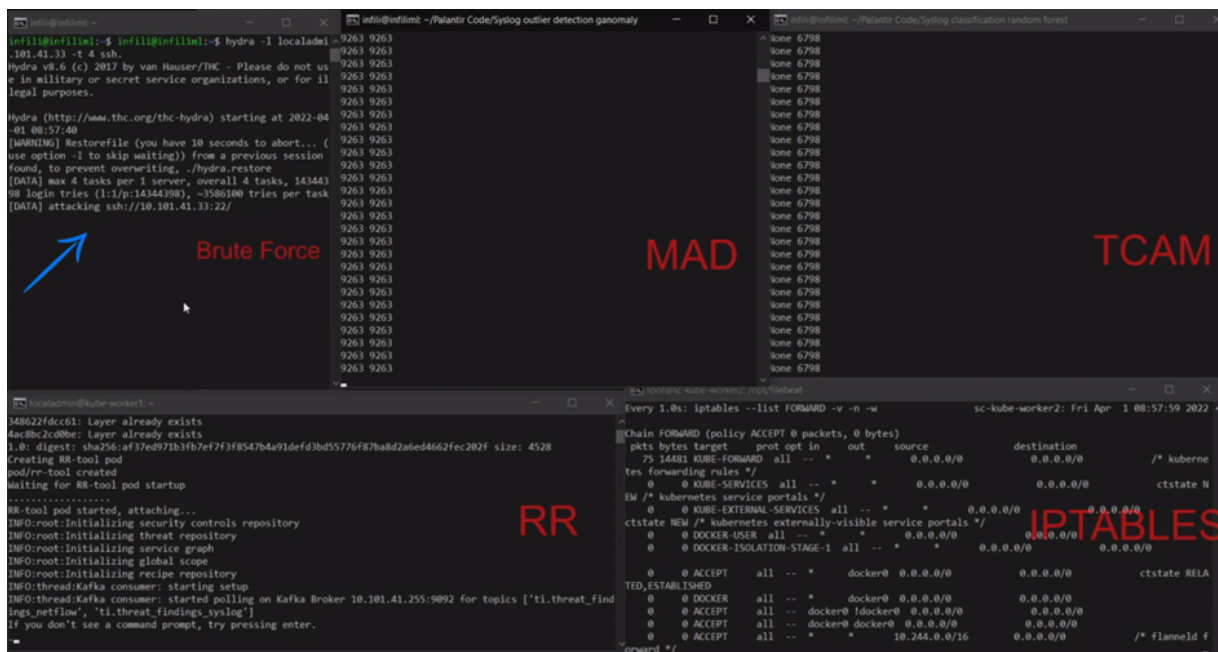


Figure 31: Brute-force attempt on the protected infrastructure

3. **Threat Intelligence:** The TI component pre-processes the syslog in real time, detects the anomalous behaviour (MAD) (Figure 32), which is classified as a brute-force attack (TCAM) (Figure 33). The network operator is notified about the attack (Figure 34).

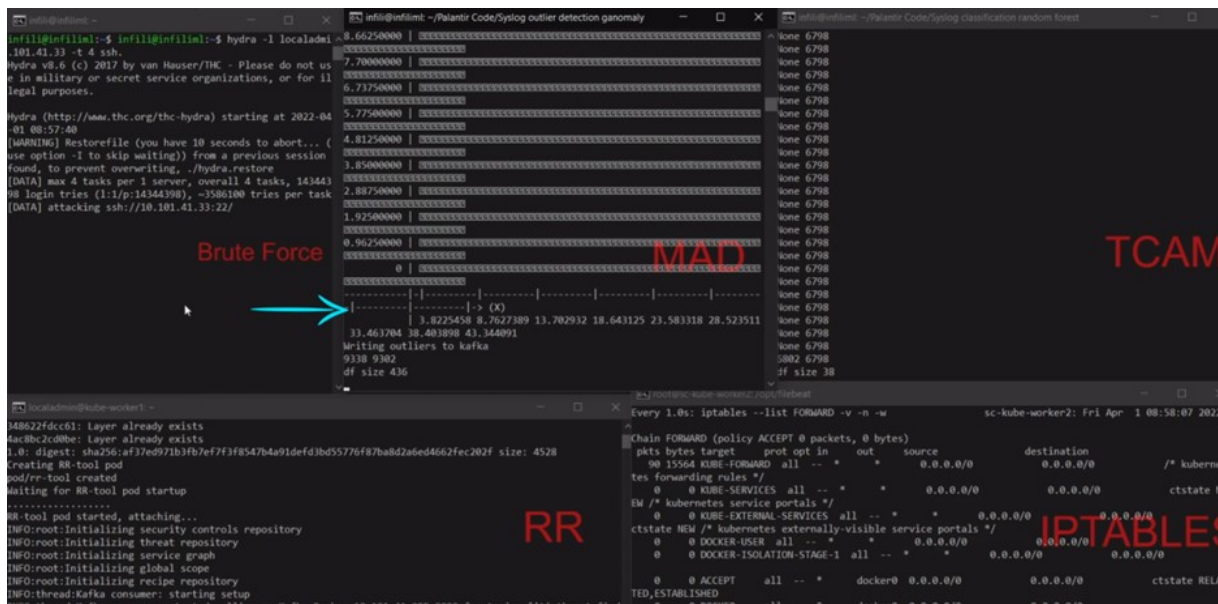


Figure 32: The logs generated by the brute-force attack are successfully identified as outliers by MAD

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	67 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final

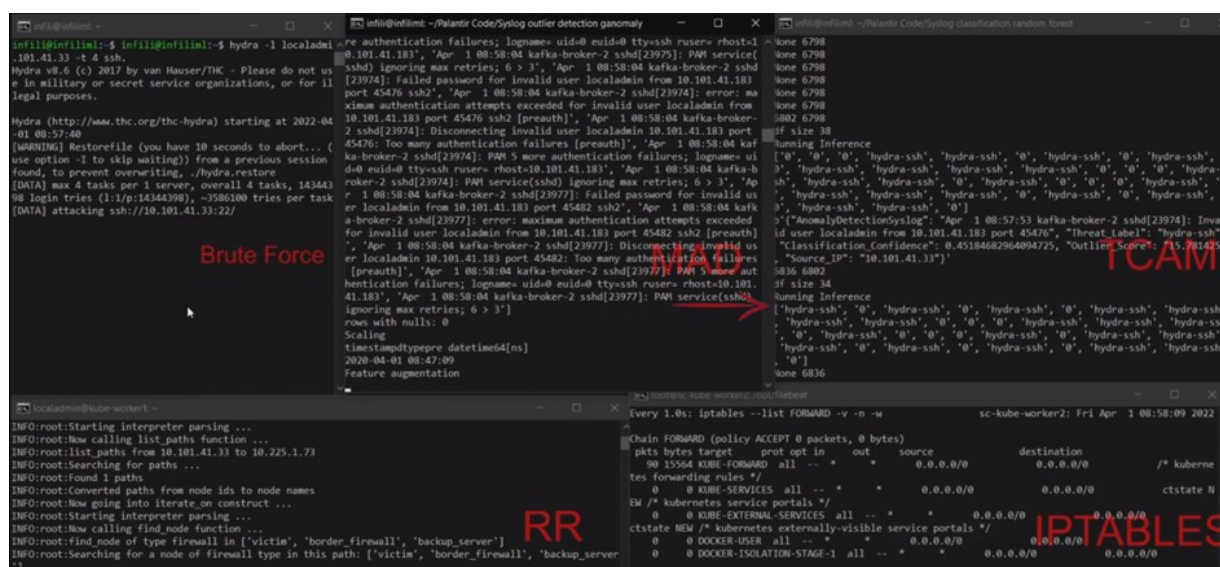


Figure 33: TCAM labels the outliers as “hydra-ssh” attack

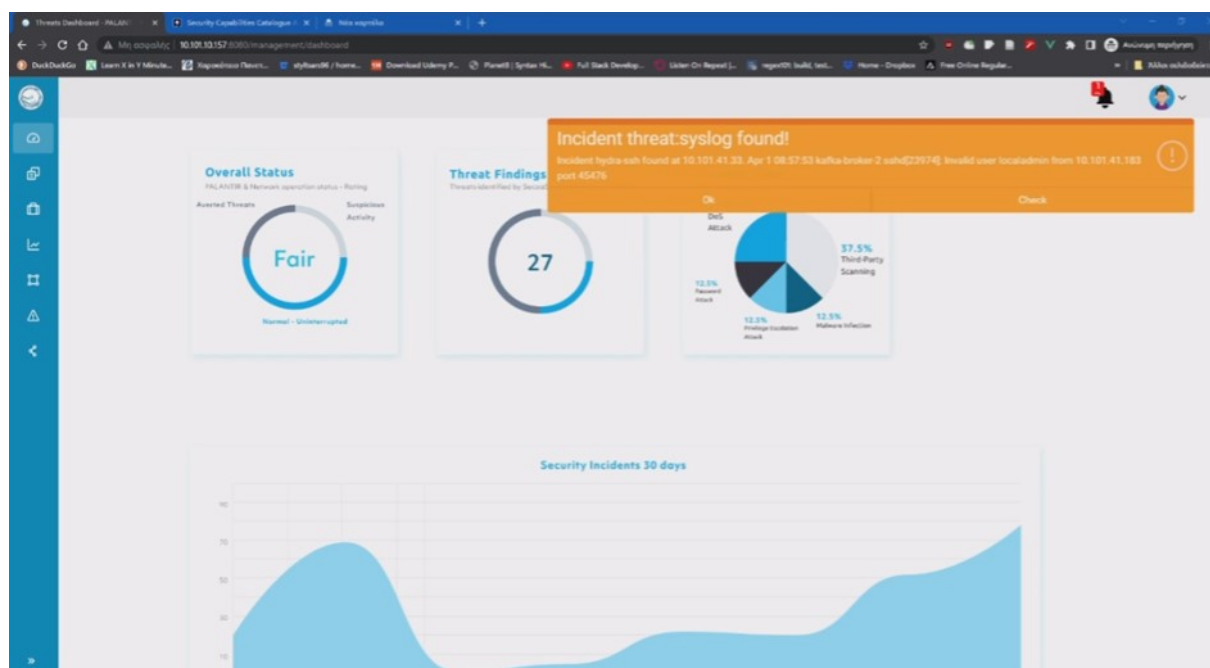


Figure 34: The network operator is notified about the data breach incident via the Portal

4. **Mitigation:** The Recommendation and Remediation subcomponent (RR) proposes two policies based on the threat findings: a) apply a FW rule to isolate the server, and b) perform an instant backup of the victim server’s data (Figure 35).

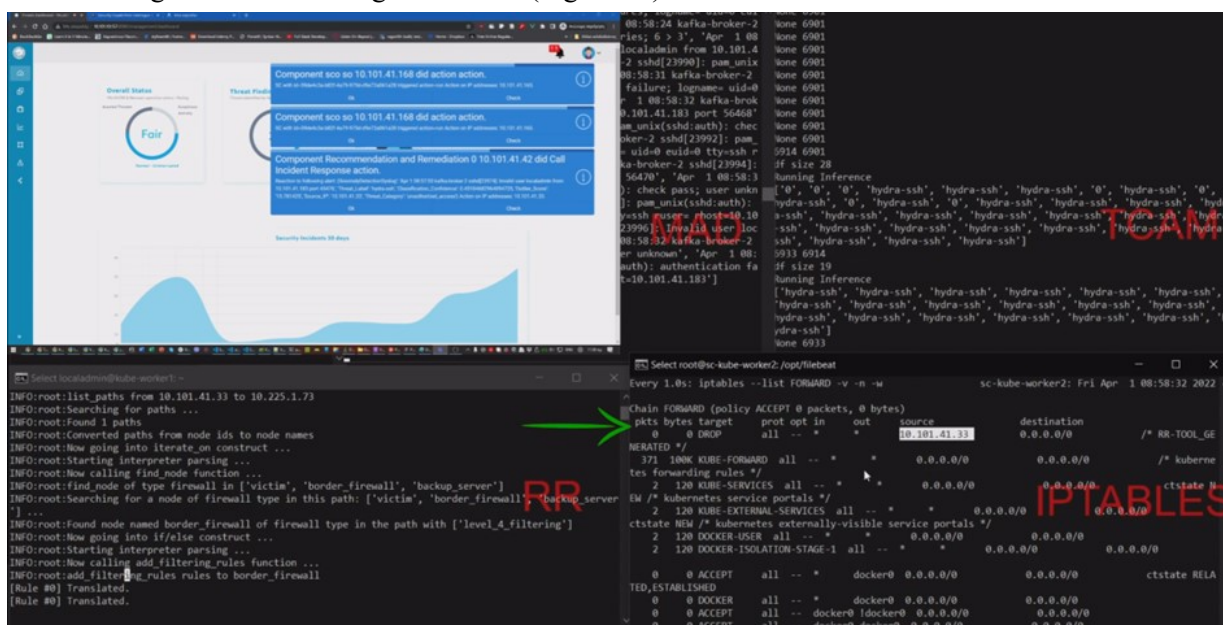
Document name:	Integration & Validation Report: Use case results and playbook (first prototype)	Page:	68 of 77
Reference:	D6.1	Dissemination:	Public
Version:	1.0	Status:	Final


```

Select localadmin@kube-worker1: ~
INFO:root:list_paths from 10.101.41.33 to 10.225.1.73
INFO:root:Searching for paths ...
INFO:root:Found 1 paths
INFO:root:Converted paths from node ids to node names
INFO:root:Now going into iterate_on construct ...
INFO:root:Starting interpreter parsing ...
INFO:root:Now calling find_node function ...
INFO:root:find_node of type firewall in ['victim', 'border_firewall', 'backup_server']
INFO:root:Searching for a node of firewall type in this path: ['victim', 'border_firewall', 'backup_server'] ...
INFO:root:Found node named border_firewall of firewall type in the path with ['level_4_filtering']
INFO:root:Now going into if/else construct ...
INFO:root:Starting interpreter parsing ...
INFO:root:Now calling add_filtering_rules function ...
INFO:root:add_filtering_rules rules to border_firewall
[Rule #0] Translated.
[Rule #0] Translated.
  
```

Figure 35: A double mitigation policy is proposed by RR

- Blocking attacker access:** A running SC (firewall) is reconfigured with newly added rule based on the RR policy to block the brute-force attack. The network operator is notified about this mitigation action through the Portal (Figure 36).



The figure consists of three main components:

- Security Dashboard:** Shows an 'Overall Status' of 'Fair' and a 'Threat Profile' section with a 'Component Recommendation and Remediation' for '10.101.41.42 did Call Incident Response action'.
- Terminal Window:** Displays a brute-force attack log for 'pam_unix(sshd:auth): check pass; user unknown; pam_unix(sshd:auth): authentication failure; logname=uid=0...'. A green arrow points from this terminal to the iptables window.
- IPTABLES Configuration:** Shows the 'Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)' with rules for blocking traffic from '10.101.41.33' to '10.101.41.42'.

Figure 36: The firewall SC interrupts the brute-force attack

- Preventing data loss:** The Incident Response (IR) subcomponent of Fault and Breach Management (FBM) successfully translates the RR mitigation policy (Figure 37) and performs a backup of the SQL dump to a secure node (Figure 38). The user is notified about the backup action through the Portal.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	69 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

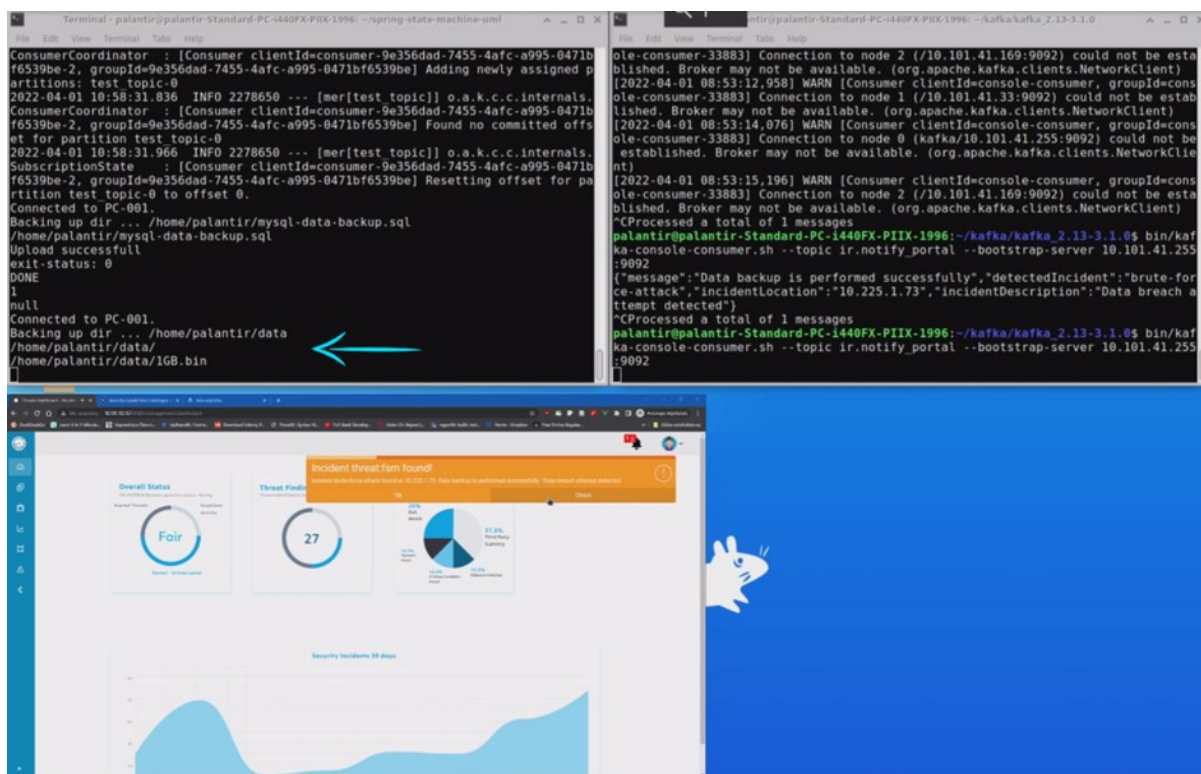


Figure 37: IR initiates the backup process after the RR trigger

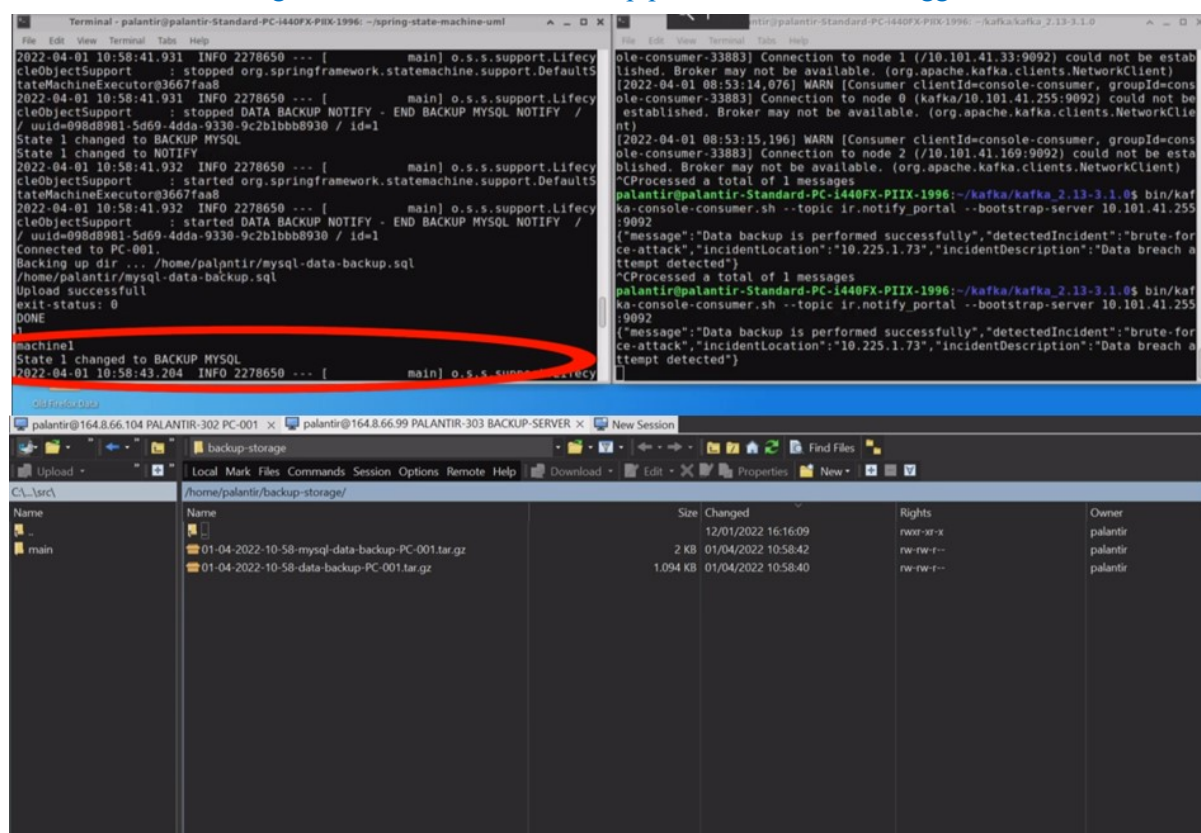


Figure 38: IR initiates the backup process after the RR trigger

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	70 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

4.2.3 Storyline 3: Attestation and Attestation Failure

This scenario involves the Dashboard, the Security Capabilities Orchestrator, and the Catalogue, as well as the Attestation Engine, the Recovery Service, and the Snort Security Capability. Figure 39 depicts these involved subcomponents along with the detailed set of steps expected to occur when performing the scenario.

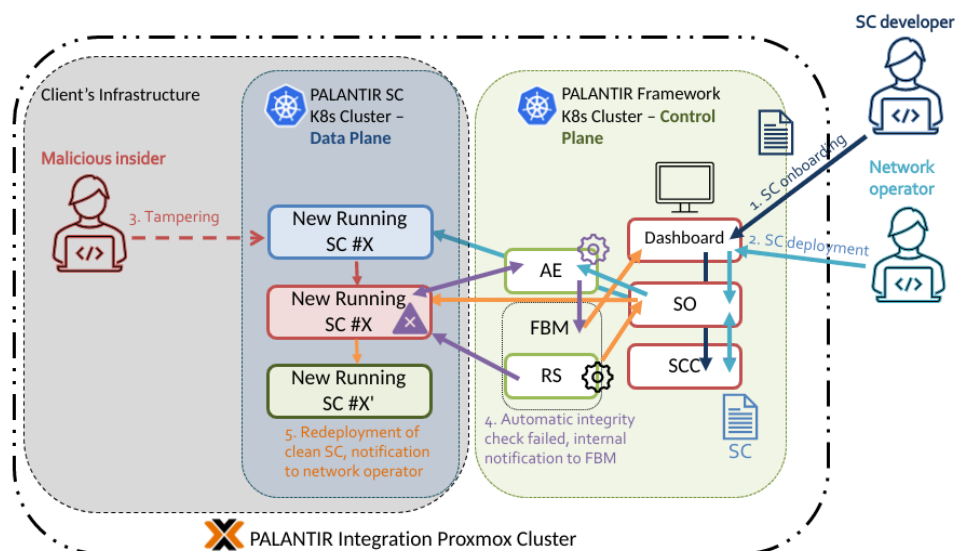


Figure 39: Workflow for Storyline 3

Initially, during the first step, the SC developer defines the logic of the security service (Snort, in this case). When onboarding the SC (i.e., registering it in the SCC), the developer fills in all relevant metadata to store in the catalogue, which will be used later on to search for security capabilities based on the given parameters. Upon the onboarding process, the SC metadata is stored at the SCC and the SC packages are onboarded and registered into the NFVO, through the SO.

After some time, the second step kicks off with the network operator specifying an explicit SC to be instantiated. From the Dashboard, the operator selects the capability and deploys it. During this procedure, the SC spins up at the k8s cluster for the data plane, as well as fetching relevant container runtime information to provide to the AE in order to use in future stages.

The third step occurs when a malicious insider accesses the client's infrastructure, then the k8s cluster for the data plane, and finally gaining access to the container where the previously deployed Snort SC instance is running. The malicious user tampers a binary in the container, for instance disguising some malware as one of the commonly used binaries in the common system directories.

After few moments, in the fourth step, the background attestation process will return a failed integrity check, effectively setting the container into an untrusted state and returning a notification to the network operator through the Dashboard, as observed in Figure 40.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	71 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

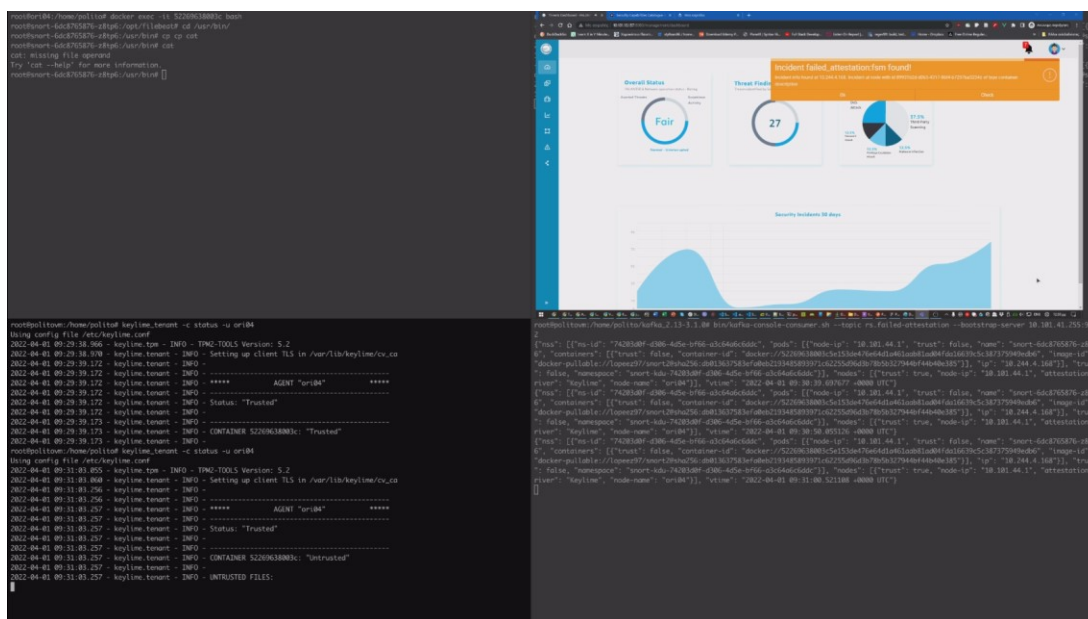


Figure 40: Identification and notification of a compromised container in Storyline 3

It will also notify FBM. Inside the latter, the RS generates a relevant policy that will immediately request the SO for terminating the compromised, untrustworthy container with the Snort SC and also issue the redeployment of a new clean instance of the same type.

Finally, in the fifth step, the redeployment of the new SC instance finished, and the network operator is notified of the successful recovery.

4.2.4 HPE-Attestation Engine Standalone Demo

Along with the AE presented by POLITO in storyline 3, HPE presented their standalone version of AE. The demonstration runs in an HPE-internal test bed, consisting of the following minimum requirements:

- AE: a machine equipped with 4 CPU, 8GB of RAM, 60 GB of HDD, OpenSUSE Leap 15.3;
- Attestation Agent: bare-metal node equipped with a TPM 2.0 chip, HPE's iLO, OpenSUSE Leap 15.3.

HPE's AE aims at attesting the following: Hardware Attestation, Firmware Attestation, Load-time Attestation of the OS, Kernel Runtime Attestation with HPE DIME. In addition, HPE's Attestation Engine provides Reference Measurement Management associated with those attestation capabilities.

HPE's AE first standalone demonstration comprises of the following steps:

- **OS Verification:** HPE's AE leverages Linux IMA feature for measuring files loaded by the operating system at load time which guarantees that a program file is measured before it is executed, so malicious program cannot tamper its measurement. A scenario (Figure 41) is depicted where a new binary file is added to the system by malicious actor which will be create an IMA violation and will be detected by the Attestation Engine while loading the attestation agent service.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	72 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final


```

sug@serverx:~$ curl --no-proxy "*" -k https://hss-ui.palantir-ae.local/api/v1/devices -H "Cookie: Auth=$token" | jq
% Total    % Received % Xferd  Average Speed   Time    Time     Time    Current
           Dload  Upload   Total     Spent    Left     Speed
100 4403    0 4403    0    0    214k      0 --:--:-- --:--:-- --:--:-- 214k
{
  "members": [
    {
      "uid": "39373638-3935-5a43-4a31-323030373831",
      "name": "hsepal1",
      "productName": "ProLiant DL360 Gen10",
      "type": "appliance",
      "status": "critical",
      "mode": "monitored",
      "monitoredDate": "Mon, 28 Mar 2022 18:52:19 UTC",
      "tags": [],
      "nextAttestationExpirationTime": "Wed, 15 Jun 2022 11:17:22 UTC",
      "isAttestationExpired": false,
      "lastAttestation": {
        "summary": "https://hss-apigw.palantir-ae.local:443/api/v1/devices/39373638-3935-5a43-4a31-323030373831/attestations/1988?severity=critical,warning",
        "log": "https://hss-apigw.palantir-ae.local:443/api/v1/devices/39373638-3935-5a43-4a31-323030373831/attestations/1988",
        "date": "Wed, 15 Jun 2022 10:17:22 UTC",
        "highlightedIssues": [
          {
            "description": "hash mismatch",
            "type": "os",
            "severity": "critical",
            "count": 1
          },
          {
            "description": "missing path",
            "type": "os",
            "severity": "warning",
            "count": 2
          },
          {
            "description": "filtered event",
            "type": "os",
            "severity": "unknown",
            "count": 189
          }
        ]
      }
    }
  ]
}

```

Figure 41: HPE-OS Verification: Failed attestation alert due to IMA policy violation

- Firmware Attestation:** HPE's AE ensures that the platform initializes in a known-good state from hardware up to the firmware level (Figure 42). Measured boot is used so that each component measures the next to create a secure chain of trust beginning from UEFI components continued with the bootloader followed by the operating system kernel. These measurements are recorded in Platform Configuration Register (PCR) in TPM for tamper resistance. A scenario is assumed where a malware is able to disable Secure boot configuration in a platform which is detected by the Attestation engine and alerts a failed attestation.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	73 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

```

sug@serverx:~$ curl --no-proxy "" -k https://hss-ui.palantir-ae.local/api/v1/devices -H "Cookie: Auth=@token" | jq
% Total    % Received % Xferd  Average Speed   Time    Time     Time    Current
           Dload  Upload   Total     Spent    Left     Speed

100 4413    0 4413    0    0    195k    0 --:--:-- --:--:-- --:--:-- 187k

{
  "members": [
    {
      "uuid": "39373638-3935-5a43-4a31-323030373831",
      "name": "hsepal1",
      "productName": "ProLiant DL360 Gen10",
      "type": "appliance",
      "status": "critical",
      "mode": "monitored",
      "monitoredDate": "Mon, 28 Mar 2022 18:52:19 UTC",
      "tags": [],
      "nextAttestationExpirationTime": "Wed, 15 Jun 2022 12:27:11 UTC",
      "isAttestationExpired": false,
      "lastAttestation": {
        "summary": "https://hss-apigw.palantir-ae.local:443/api/v1/devices/39373638-3935-5a43-4a31-323030373831/attestations/1992?severity=critical,warning",
        "log": "https://hss-apigw.palantir-ae.local:443/api/v1/devices/39373638-3935-5a43-4a31-323030373831/attestations/1992",
        "date": "Wed, 15 Jun 2022 11:27:11 UTC",
        "highlightedIssues": [
          {
            "description": "PCR lockdown mismatch",
            "type": "uefi",
            "severity": "critical",
            "count": 4
          },
          {
            "description": "missing path",
            "type": "os",
            "severity": "warning",
            "count": 32
          },
          {
            "description": "filtered event",
            "type": "os",
            "severity": "unknown",
            "count": 21
          }
        ]
      }
    }
  ]
}

```

Figure 42: HPE-FirmAtt: Failed Firmware Attestation alert

- **Reference Measurement Management:** a RM is the baseline set of measurement that are known to be correct, for performing attestation. Proper attestation lifecycle management requires these golden measurements to be updated whenever there is an authorised patch in the OS in order to have a correct RM for correct attestation to occur. HPE's AE demonstrates capability to update these RM when required (Figure 43).

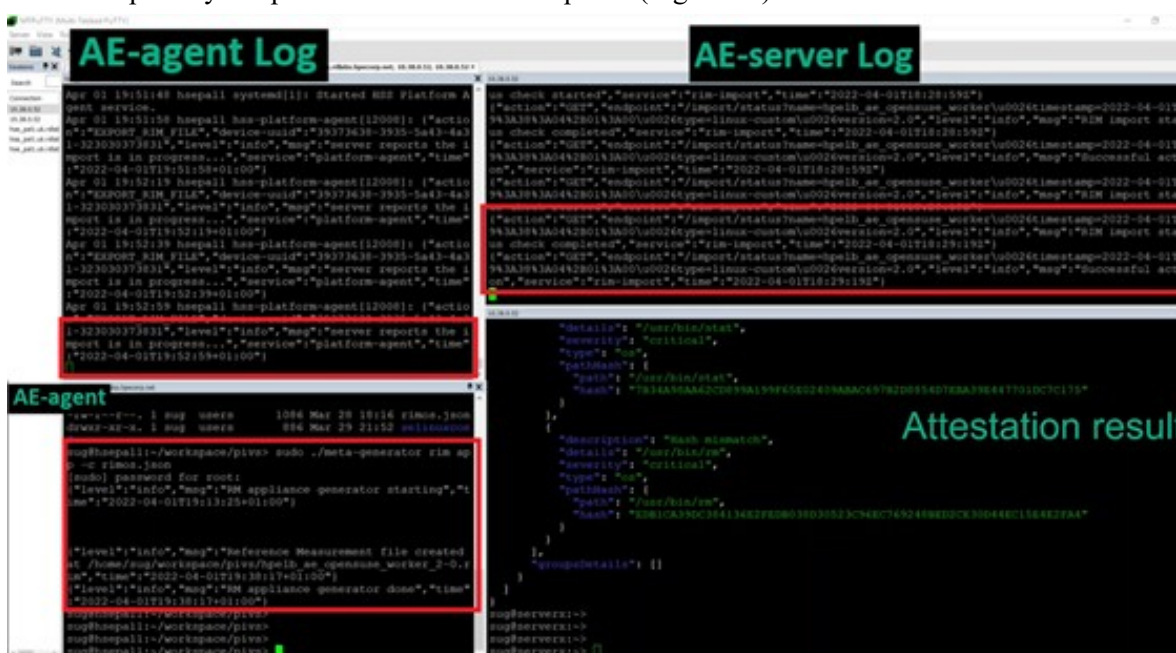


Figure 43: HPE-RMmangt: Updating new Reference Measurement

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	74 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

- Runtime Attestation with HPE DIME:** HPE DIME is an OS intrusion detector that implements a kernel memory inspection capability of the platform to detect any unexpected change of code or data already loaded in memory (Figure 44). HPE DIME continuously monitors and verifies portions of OS kernel using a scanning engine in iLO – a Baseboard Management Controller (BMC) for remote server management embedded on a system board. Since the kernel is the most privileged and sensitive part of the OS running on a server, it is often targeted by attackers by inserting rootkits into the kernel to provide them with a hidden back door. These rootkits give the attacker a persistent access to the system as it acquires a root or administrator access to the system. One of the publicly available examples of such rootkit is Diamorphine [7], a kernel space rootkit, which generally adds a malicious kernel module to the system allowing broadest user privileges and control to all system processes. The DIME driver in Attestation Agent takes an initial snapshot of the kernel and periodically validates it to the scanning engine in iLO which can detect these behaviours of the rootkits and alert a failed attestation to the AE.

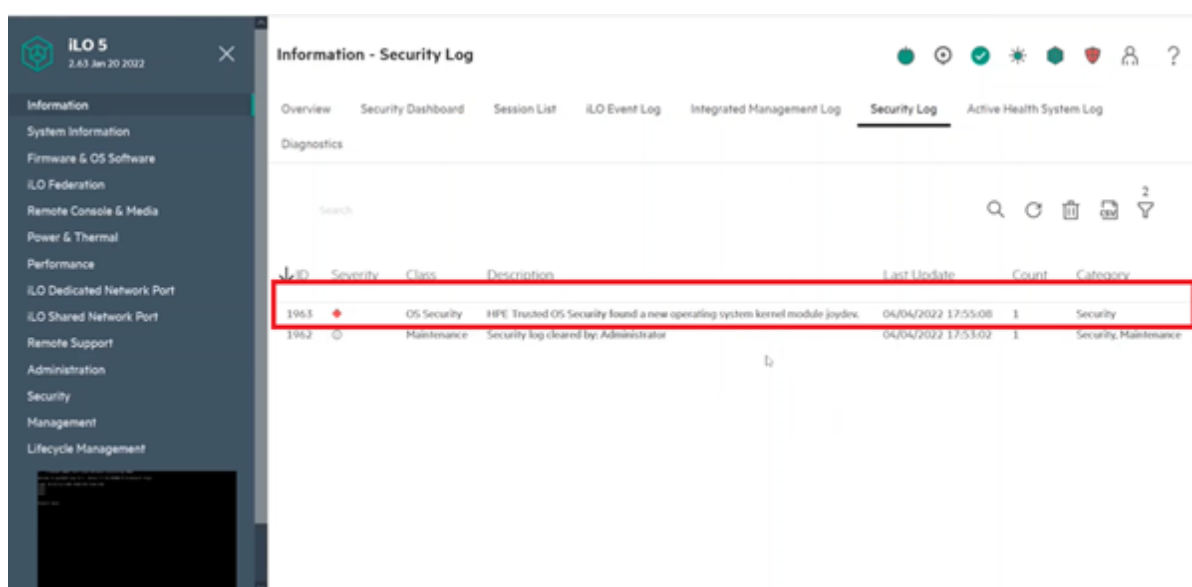


Figure 44: HPE-DIME: New kernel module added alert in iLO

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)					Page:	75 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status:	Final

5. Conclusions

This deliverable provides the initial details on the PALANTIR Minimal Viable Product (MVP) - the first version, the Pilots to evaluate the PALANTIR Platform and the first version of the manual describing and showcasing the PALANTIR Platform, Testbeds and Innovation Labs.

The document provides the current, up-to-date aggregated architecture of the PALANTIR Platform, including all technological advanced carried out under WP3, WP4, and WP5 and integrated into the PALANTIR Solution as PALANTIR MVP. The PALANTIR Testbed and Innovation Labs outline the installation guide to deploy the components and test the solution and first demonstrations of the PALANTIR MVP. Finally, the document outlines the three Pilots designed to address use cases and requirements defined and presented in D2.4 and evaluate the PALANTIR MVP in four distinct cases.

This document will be followed by a second iteration, "D6.2 Integration & Validation Report: Use case results and playbook (final prototype)" later on the project with the evaluation of the final PALANTIR Solution, i.e., the MVP 2.0, incorporating all final developments and results of co-creation with external stakeholders. To this end, the Pilot Specific KPIs is also expected to be refined and extended.

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)				Page:	76 of 77
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final

6. References

- [1] “5TONIC,” *5TONIC*. <https://www.5tonic.org/> (accessed Jun. 27, 2022).
- [2] “5GENESIS – 5th Generation End-to-end Network, Experimentation, System Integration, and Showcasing.” <https://5genesis.eu/> (accessed Jun. 27, 2022).
- [3] “Papyrus.” <https://www.eclipse.org/papyrus/> (accessed Jun. 27, 2022).
- [4] “S.M.A.R.T-Way-Management-Review.pdf.” Accessed: Jun. 27, 2022. [Online]. Available: <https://community.mis.temple.edu/mis0855002fall2015/files/2015/10/S.M.A.R.T-Way-Management-Review.pdf>
- [5] “Environment variables in Compose,” *Docker Documentation*, Jun. 24, 2022. <https://docs.docker.com/compose/environment-variables/> (accessed Jun. 27, 2022).
- [6] “Building applications with Maven.” <https://quarkus.io/guides/maven-tooling> (accessed Jun. 27, 2022).
- [7] V. R. Mello, *Diamorphine*. 2022. Accessed: Jun. 27, 2022. [Online]. Available: <https://github.com/m0nad/Diamorphine>
- [8] “Spring State Machine” Accessed: Jun. 27, 2022. [Online]. Available: <https://spring.io/projects/spring-statemachine>
- [9] ISO 9001:2015 Accessed: Jun. 27, 2022. [Online]. Available: <https://www.iso.org/standard/62085.html>

Document name:	Integration & Validation Report: Use case results and playbook (first prototype)			Page:	77 of 77	
Reference:	D6.1	Dissemination:	Public	Version:	1.0	Status: Final