SECURING AGAINST INTRUDERS AND OTHER THREATS THROUGH A NFV-ENABLED ENVIRONMENT

[H2020 - Grant Agreement No. 700199]

Deliverable D4.2

Updated specifications, design, and architecture for the usable information driven engine

Editor F. Ferreira (Ubiwhere)

Contributors G. Gardikis, S. Pantazis, G. Kolonias (Space Hellas), R. Preto (Ubiwhere), C. Fernandez, B. Gaston (i2Cat), P. Adamidis, A. Litke, N. Papadakis, D. Papadopoulos (INFILI), G. Dimopoulos (Talaia), M. De Benedictis (POLITO), O. Segou (ORION), J. N. Mendoza (TID).

Version 1.0

Date March 31st , 2018

Distribution PUBLIC (PU)

Executive Summary

Following the work done in D2.1/D2.2, where the requirements of the SHIELD platform were elicited and the high-level design and architecture of the platform was exposed, a detailed study of the different components has been done in order to obtain the low-level architecture and design (subcomponent granularity), the specifications (transformation of the user requirements into technical requirements/specifications) and the implementation guide (technologies to use). This work has been divided into the two technical development work packages of SHIELD namely WP3 and WP4. Hence, this deliverable covers the two components developed within WP4, i.e. the data analysis and remediation engine (DARE) and the security dashboard.

From the point of view of the low-level architecture, we expose some changes in subcomponents from the technical architecture exposed in the Description of the Action (DoA) of the project. Firstly, we have simplified some subcomponents with the objective to avoid using technologies with specifications that SHIELD does not use. For example, the Enterprise Service Bus (ESB) has been replaced by a streaming service, since it is only needed to communicate information from the Virtual Network Security Functions (vNSFs) to the central DARE and hence, no bidirectional flow or multi-cast is needed. It is worth mentioning that SHIELD already identified several phases for the information treatment and valorisation, the main task of the WP4. These phases are defined in detail from the bottom (data ingestion) to the top (data visualisation) of the data valorisation stack. Between them, we define the data analysis phase to detect anomalies in the network and classify them, and the cybersecurity topologies phase, which recommends specific remediations to the detected threats in the form of new network services (sets of vNSFs) or reconfiguration of existing ones.

In terms of design, we have identified the five user stories behind the three use-cases of SHIELD, namely: i) vNSFs deployment, ii) vNSF withdrawal, iii) anomaly detection, iv) recommendation deployment and v) monetisation definition. From these user stories, we have exposed and updated the workflow between subcomponents and also between the subcomponents of the WP4 and the components of WP3. These workflows identify the flow of information and tasks that define and compose every user story.

One of the main aspects exposed in this deliverable is the transformation of the user requirements into specifications or technical requirements. Since SHIELD's engineering process is based on two iterations of the requirements elicitation, the final specifications and design of these components is herein provided, based on the preliminary work in D4.1 and the updated requirements in D2.2. This work concludes the transformation of user requirements into a high-level design and architecture, that later evolves to technical specifications. The deliverable is also updated with ethical and regulatory compliance (ERC) requirements and a Regulatory Compliance section that focus on maintaining the platform's alignment with the EU regulatory landscape, mainly regarding the General Data Protection Regulation (GDPR). The possible implications of these updated requirements in every phase and subcomponent have been identified and the requirements have been translated from the business language used in D2.1/D2.2 to the technical language needed for the developments. As a further evolution from D3.1, this deliverable also focuses on the Dashboard's Role-Based Access Control (RBAC) and on the implemented billing framework.

Finally, following the detailed research on the most relevant technologies in the fields of cybersecurity, big data and data visualisation, our conclusion remains that Apache Spot [1] is currently the best choice for the technical base for the DARE. In the context of SHIELD, Spot is currently being used as the starting point for the DARE developments. The necessary extensions to the core Spot platform in order to fulfil the SHIELD requirements are currently under development and involve functionalities such as: distributed data collection; near-real time operation; alternative anomaly detection methods; threat classification and labelling; mitigation capabilities; multi-user support and visualisation; optimised operation in a Network Function Virtualisation (NFV) environment and enhancement of the data model to support for more types of information. All technical advancements and specification updates since D4.1 can be found in Annex A.

Table of Contents

1. INTRODUCTION

1.1. SHIELD project overview

This document presents the low-level architecture, design and specifications of the components involved in the usable information-driven engine, within WP4. This deliverable starts from the high-level architecture, design and requirements presented in D2.2 [2], and provides specific details of the components' design, definition and their suitability regarding the SHIELD requirements.

SHIELD, as a use case driven project, aims to cover the functionality required by the following three use cases (already defined in D2.2 but briefly recalled here for the sake of completeness):

- *Use case 1*: An Internet Service Provider (ISP) using SHIELD to secure its own infrastructure. This UC involves the ISPs deploying vNSFs in their network to detect incidents (Figure 1).

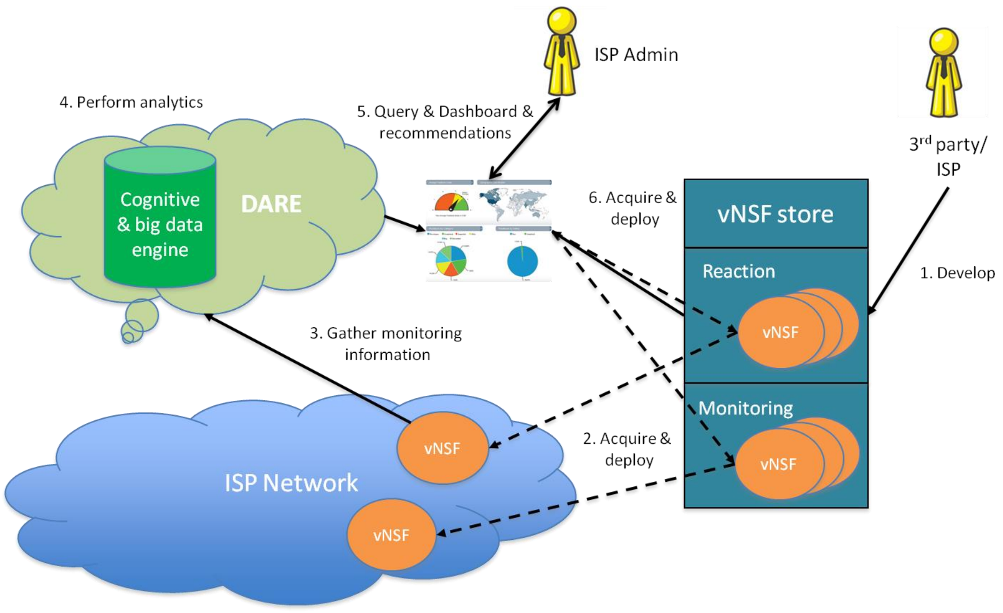

Figure 1: High-level picture of the use-case 1

- *Use case 2:* An ISP leveraging SHIELD to provide advanced SecaaS services to customers. This UC assumes that network security services (consisting of vNSFs), along with real-time incident detection and management, are offered as-a-Service to ISP clients, such as enterprises, public bodies, etc. (Figure 2).

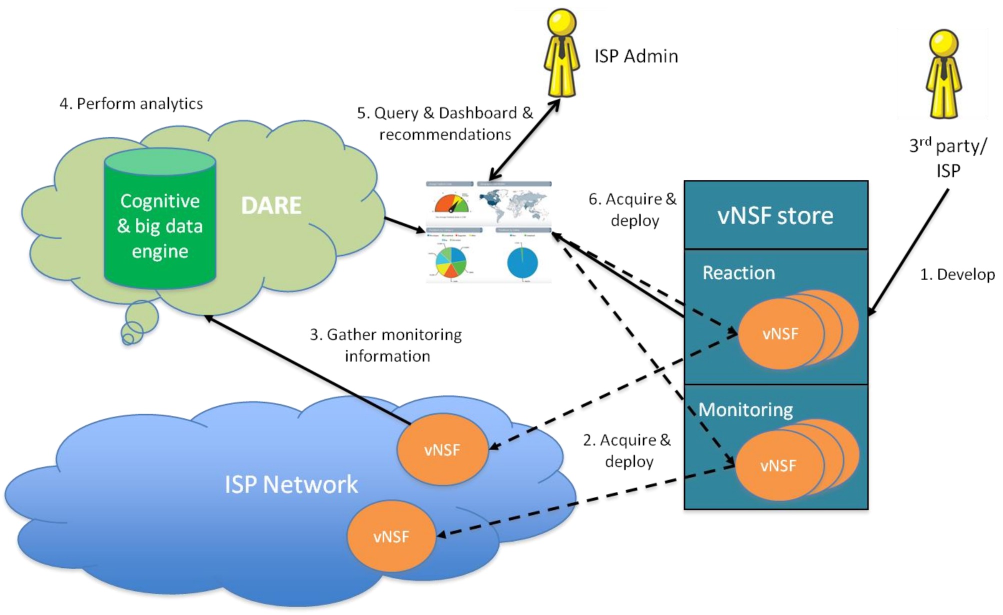

Figure 2: High-level picture of the use-case 2

- *Use case 3:* Contributing to national, European and global security. This UC assumes that incident information is exposed, in a secure and private manner, to public cybersecurity authorities (Figure 3).

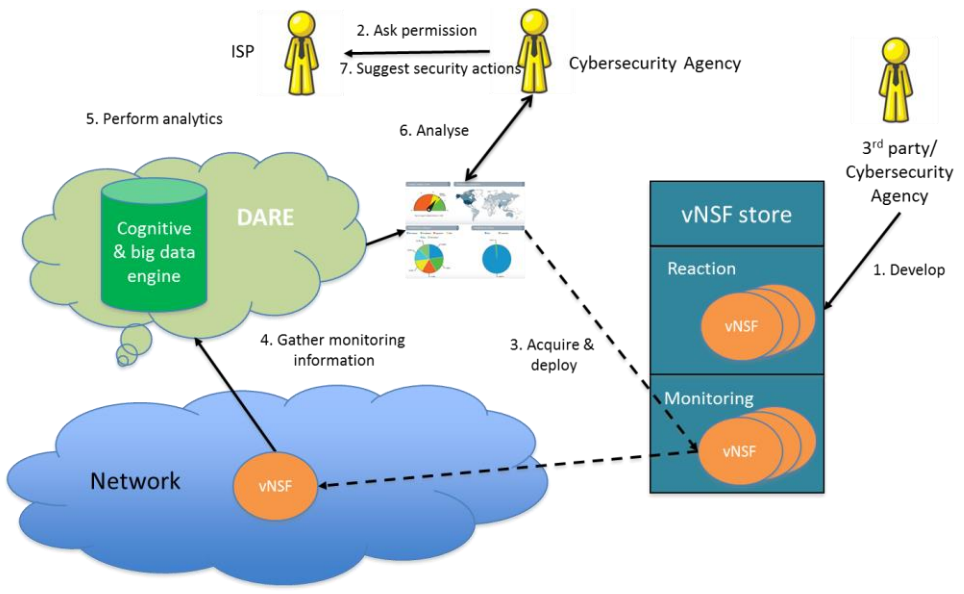

Figure 3: High-level picture of the use-case 3

Although the three use cases act as the basis of the analysis, the resulting architecture, design, specifications and implementation have been elaborated to produce a unified and universal solution i.e. a single cybersecurity solution that can be used for multiple purposes. To this intent, the SHIELD platform enables the actors in the different use cases with different views and roles on the network. For example, while an ISP (use case 1) can view the big picture of the cybersecurity analysis and can deploy a vNSFs in any location of the network, the ISP client (use case 2) only has access to a limited vision of the cybersecurity picture (information that is offered by the ISP and/or paid by the client) and can deploy vNSFs in specific places of the network (i.e. to its gateways).

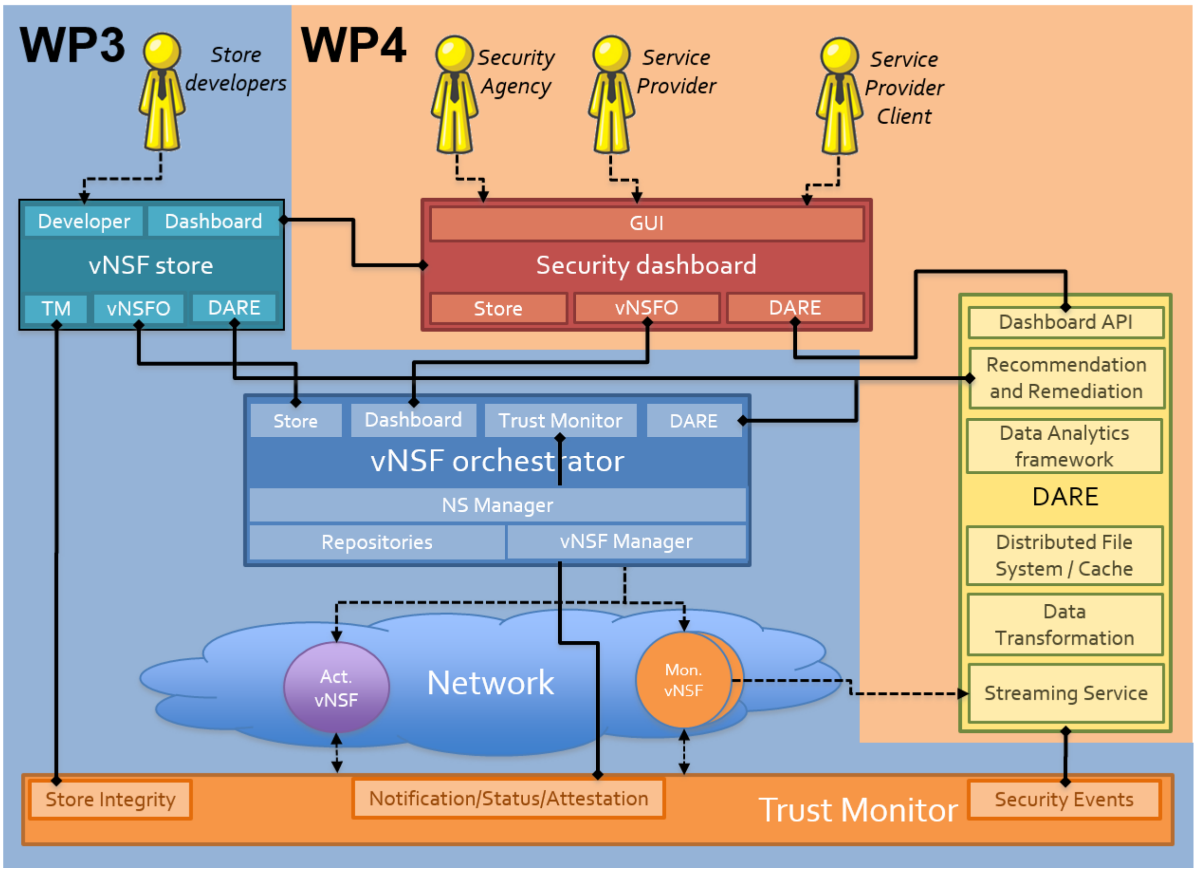

Figure 4: High-level architecture of SHIELD, with components per WP

Based on these use cases and the requirements highlighted in Deliverable D2.2 [2], the designed high-level architecture for the SHIELD platform is articulated around six different components, illustrated in Figure 4 (vNSFs, Trust monitor, vNSF Orchestrator, vNSF Store, Security overview dashboard and DARE). The components corresponding to WP4 are described at a low level in this deliverable. From the point of view of the usable information-driven engine (WP4), the DARE stores and analyses the security logs and events provided by the network services (NSs) and vNSFs running in the network and these results are presented to the operator in the Security overview dashboard. These components collaborate with the vNSF ecosystem (WP3), specifically with i) the vNSF Store, which holds a registry of NS and vNSF-related information; ii) the vNSF Orchestrator, which deploys and manages the lifecycle of the NSs and vNSFs; iii) the monitoring vNSFs which produce the information to detect the threats; and iv) the Trust Monitor, which verifies that both NSs and vNSFs, as well as other nodes from the infrastructure, are trusted at all times.

The high-level design presented in D2.2 states that the network infrastructure provides a trusted environment for supporting the execution of vNSFs. For attestation purposes, the network infrastructure interacts with the Trust Monitor to authenticate the integrity of each network component. The network infrastructure is interconnected with the vNSF Orchestrator through the vNSF Manager Engine. This interaction allows the deployment of vNSFs, the vNSF lifecycle management and the collection of monitoring information. Moreover, in case of attestation incidents, the Trust monitor can act on the network to solve the issue and inform to the DARE of the arisen situation. Monitoring vNSFs inspect captured data and provide valuable information to the DARE. The network status is reported periodically since events, not detectable by individual vNSFs, are inferred by centralising all the information in the DARE. Then, the data analytics framework analyses all the heterogeneous network information previously collected via monitoring vNSFs and Trust monitor. It features cognitive and analytical components capable of predicting specific vulnerabilities and attacks. Finally, the remediation engine provides recommendations in the form of medium level policies, i.e. configurations of vNSFs that compose the different Network Services available in the catalogue (for a specific user), to remediate the detected threats. These recommendations and the attack information is given to the intuitive and appealing graphical user interface implemented in SHIELD, which allows authenticated and authorised users to access SHIELD's functionalities. From this dashboard, operators have access to monitoring information showing an overview of the security status. Furthermore, this dashboard allows operators as well as SecaaS clients to take actions and react to any detected vulnerability.

In D3.2 [3], a detailed view of the vNSFs ecosystem components is provided. However, with the aim to provide self-explained deliverables, we summarise their main functionalities:

There are two types of vNSF functionalities in SHIELD, monitoring and reacting vNSF. Monitoring vNSFs are configured to send, in an efficient manner, the collected traffic to the DARE, while reacting vNSFs are configured to stop an ongoing attack or to remediate a detected vulnerability. However, this classification is not strict; many vNSFs are exposing both capabilities (i.e. monitoring and reacting).

The vNSF Store acts as a nexus between the vNSF Orchestrator and third-party vNSF providers/developers, who can register and manage vNSFs to be available through the SHIELD platform. The Store handles all the vNSF data related with the service, the software images and the information required to validate the integrity of itself.

The vNSF Orchestrator, is responsible for managing the lifecycle of vNSFs. Among others, this allows to deploy (instantiate and place) vNSFs in specific points of the network infrastructure. The vNSF Orchestrator interacts with each of the other modules to obtain data on the vNSFs, to receive deployment requests or to convey information of specific vNSFs to enable analysis processes.

The Trust Monitor is the component in charge of monitoring the trust of the SHIELD infrastructure. This is achieved by a combination of authentication and integrity: each node joining the infrastructure must be properly authenticated and provide also a proof of the integrity of its software stack, by leveraging Trusted Computing (TC) mechanisms.

1.2. Scope of this document

SHIELD dedicates WP4 ("Usable information-driven engine") to the technical work required towards achieving the following key goals: (a) to develop acquisition and storage capabilities for data associated with cyberattacks; (b) to develop data analytics capabilities for anomaly detection, by employing machine learning techniques for traffic classification; (c) to develop cyberattacks mitigation capabilities, providing remediation policies to be deployed in specific places of the network; and (d) to develop a graphical user interface with operations and management capabilities, making SHIELD a cybersecurity platform able to dynamically deploy network security services for a SecaaS client.

This document (D4.2 "Updated specifications, design, and architecture for the usable information driven engine") details the final design choices regarding the DARE and the Dashboard components. During M1-M19, SHIELD has developed the foundation and proof-of-concept for the data acquisition and storage, the analysis and anomaly detection, and the recommendation and remediation modules. A GUI was also developed to convey to the user a remediation, in the form of a set of security policies, to mitigate an ongoing cyberattack, along with the option to have such policies automatically applied on the vNSFs, at the touch of a button. Such features were first demonstrated during the Y1 review.

D4.2 draws inputs from the following deliverables:

- D2.1 "Requirements, KPIs, design and architecture" [4] defines high-level requirements for the SHIELD platform and the overall architecture, including the KPIs to use in evaluation phase. D2.2 "Updated requirements, KPIs, design and architecture" [2] is the final, updated version of D2.1, which was drafted concurrently with this document.
- D4.1 "Specifications, design and architecture for the usable information-driven engine" [5] contains the first version of the design and specifications for the SHIELD DARE and Dashboard. This document builds upon D4.1 and provides the finalized specifications and design.
- D3.1 "Specifications, design and architecture for the vNSF ecosystem" [6] contains the detailed design and specifications for SHIELD's DARE components, including analysis and remediation.
- D5.1 "Integration results of SHIELD HW/SW modules" [7] provides guidelines for the integration and testing of vNSF ecosystem components.

1.3. Organisation of this document

This document is organised as follows:

- **Section 1** (present section) serves as a basic introduction to this document and its scope.
- **Section 2** provides an overview of the design and architecture of the usable information-driven engine.
- **Section 3** lists the specifications and implementation details for the usable information-driven engine.
- **Section 4** discusses the regulatory and ethical compliance specifications for the usable information-driven engine and is a new addition to this document.
- **Section 5** concludes the document and lists future WP4 work.
- **Annex A** lists the technical updates and the history of changes from D4.1 to D4.2.

2. Design and Architecture

In this section, the design and architecture of SHIELD are presented for those components defined within WP4, i.e. the DARE and the Security dashboard. This description is more detailed than in D2.2 [2], as it specifically addresses low-level details, as the subcomponents for the vNSF environment, their detailed workflows and relation between these and the other components in the SHIELD platform.

2.1. Guiding principles

The WP4 components to develop for SHIELD must embrace certain considerations and follow some common principles in order to facilitate integration, interaction and operational issues, as well as to foster future platform enhancements and evolutions. This section highlights specific top-level and cross-component aspects to be considered and, where applicable, to be taken into account during the component design and implementation phases.

2.1.1. Multi-user aspects in SHIELD

SHIELD targets a multi-user environment, where several users have access to the same platform but with functionalities and views tailored to their needs. SHIELD foresees three main categories of users:

- Security clients: These are clients that have requested specific cybersecurity services to the ISP. According to the SHIELD SecaaS model, the client requests a specific service, e.g. protection against a Distributed Denial of Service (DDoS) attack, but the implementation of this services is on the ISP side. Hence, apart from the basic account information and administration of cybersecurity services (i.e. service acquisition, removal, etc.), the client will only be granted with limited visualization services. This visualization will allow him/her to see what has happened to the network service (and only in his/her network services) according to the services hired. For example, if a client has bought DDoS protection, he/she will see information like the number of attacks rejected, the severity of such attacks, the IP addresses involved, etc.
- Cybersecurity agencies: These are expert users from cybersecurity agencies that, after an agreement with the ISP, can access limited visualizations of the information. In this case, the information will not be about one client, but aggregated of all the clients of the ISP and the ISP itself. For example, if a cybersecurity agency has agreed with the ISP access to 0-day threat propagation, it will have access to information like the number of infected devices, the number of new infections per minute, propagation maps, etc.
- ISP's security department: These are security expert users from the ISP which will have administration roles. They can:
    - Accept or reject a SecaaS service requested by a client.
    - Set up or withdraw cybersecurity services for clients, as per the previous requests.
    - See all the cybersecurity information, being able to filter by user, by range of IPs, or by other characteristics.

- Apply recommendations either to clients or to the ISP itself.
- Give or remove permissions to other users.
- Consult network topology and status per vNSF.
- Retrieve information on trust status per node in the infrastructure.
- Access to any historical logs.

In general, these will be the administrators of the system and hence, they will be able access to all the functionalities needed to manage the platform.

2.1.2. Security concerns

There are some inter-component and user-interaction security concerns that should be considered, namely:

- Any connection between the different components (VNFs, Orchestrator, Store, Trust Monitor, DARE and Dashboard) should be encrypted using an adequately strong algorithm such as Rivest–Shamir–Adleman (RSA), employing encryption keys of at least 256 bits.
- Information sharing is considered from SHIELD framework only to the defined users inside the platform. Hence, no information can be downloaded or further processed outside of SHIELD.
- Users will be authenticated with password and registration which must be explicitly accepted by the ISP.
- Information of the users and the attacks should be considered private allowing access to it only to authenticated and authorized users.

2.1.3. Billing rationale

The platform SHALL be compatible with a billing framework for the use of the SecaaS Services. The clients will select one cybersecurity service (e.g. DoS protection, malware protection, etc.) which will be internally linked to one or more Network Services (NS). Hence the user will not be able to access directly to the network or to the vNSFs but to access to the functionalities defined by their payment modality and to the data associated with this service (number of threats blocked, attacker IPs, etc.).

The SecaaS Client will acquire a SecaaS service through the services store and then the SecaaS Client will be able to have access to this new SecaaS Service. The billing will be an offline method. In the scope of the project, we consider a billing model of a fixed amount, per cybersecurity service, per month. Moreover, we also consider the paying model to the vNSF developer of a fixed amount, per proprietary vNSF deployed, per month.

For this reason, before the ISP can do the onboarding of a SecaaS Service, a client must purchase it beforehand. The client will be able to access a list of available services through the dashboard and buy the ones it wants to use. The ISP will then be responsible for the deployment, maintenance and management of the network services that perform the functionality associated with such services.

The bill will be shared with an invoicing framework (external ones will be possible to use). This bill will identify the SecaaS Client, the SecaaS Service with its description, its begin and end dates for usage, and the amount to invoice. This data will be provided through a REpresentational State Transfer (REST) service.

2.1.4. Untrusted nodes policy

Untrusted nodes shall be isolated from the rest of the infrastructure at the network level. The same process could be applied to untrusted vNSFs, although it might not be sufficient in case the attacker manages to harm the host by exploiting vulnerabilities at the virtualisation level (e.g. privileges escalation in Docker). To do so, the Trust Monitor will interact with the vNSF Orchestrator (vNSFO) to notify the trust status of the node. In addition, the platform should be able to re-deploy the Network Services formerly deployed on the untrusted node with the proper configuration.

2.2. Low-level Architecture

As explained in Section 1, the high-level design elaborated in D2.2 has stated that SHIELD is composed by 6 components. Four of them belong to WP3 (vNSFs, Orchestrator, Store and Trust Monitor), while two of them belong to WP4 (DARE and Dashboard). Although D2.2 was only exposing high-level architecture and design, there were some references to the envisioned subcomponents. Namely, the high-level architecture was exposing a Data Services Centre for data format transformation, an Enterprise Service Bus for data transportation, etc. However, as D2.2 was high-level architecture, these subcomponents, which are actually part of the low-level design, were not described in detail.

During T4.1 the consortium has studied conscientiously the requirements elicited in D2.2 together with the specifications needed to fulfil them. Moreover, several technologies that are currently used for threat detection and remediation have also been considered. The results of this detailed study extensively explained in Section 3, imply some adjustments in the low-level architecture of the modules and hence, some changes in the envisioned components from D2.1.

2.2.1. The DARE

The Data Analysis and Remediation Engine (DARE) is one of the three central innovation pillars of SHIELD, together with the vNSF ecosystem and the hardware attestation (both in WP3). The DARE centralises the management information of SHIELD and exchanges information with all the other components of the solution, as described below:

- the vNSFs, since the DARE centralises the information obtained from the monitoring vNSFs;
- the vNSF Orchestrator, since the DARE needs information of the network per SecaaS client (e.g. ISP, ISP clients using SecaaS or a Cybersecurity agency) to provide accurate and complete recommendations;
- the Trust Monitor, since the DARE needs to know if a vNSF, or even a complete node, has been compromised;

- the Dashboard, since the DARE notifies to the dashboard the detected network anomalies and one or more recommendations of network services (set of vNSFs) with their appropriate high-level policies for configuration;
- the Store, since the DARE needs to know the vNSFs and NS for deployment or reconfiguration.

The data flow diagram of the DARE is shown in Figure 5.

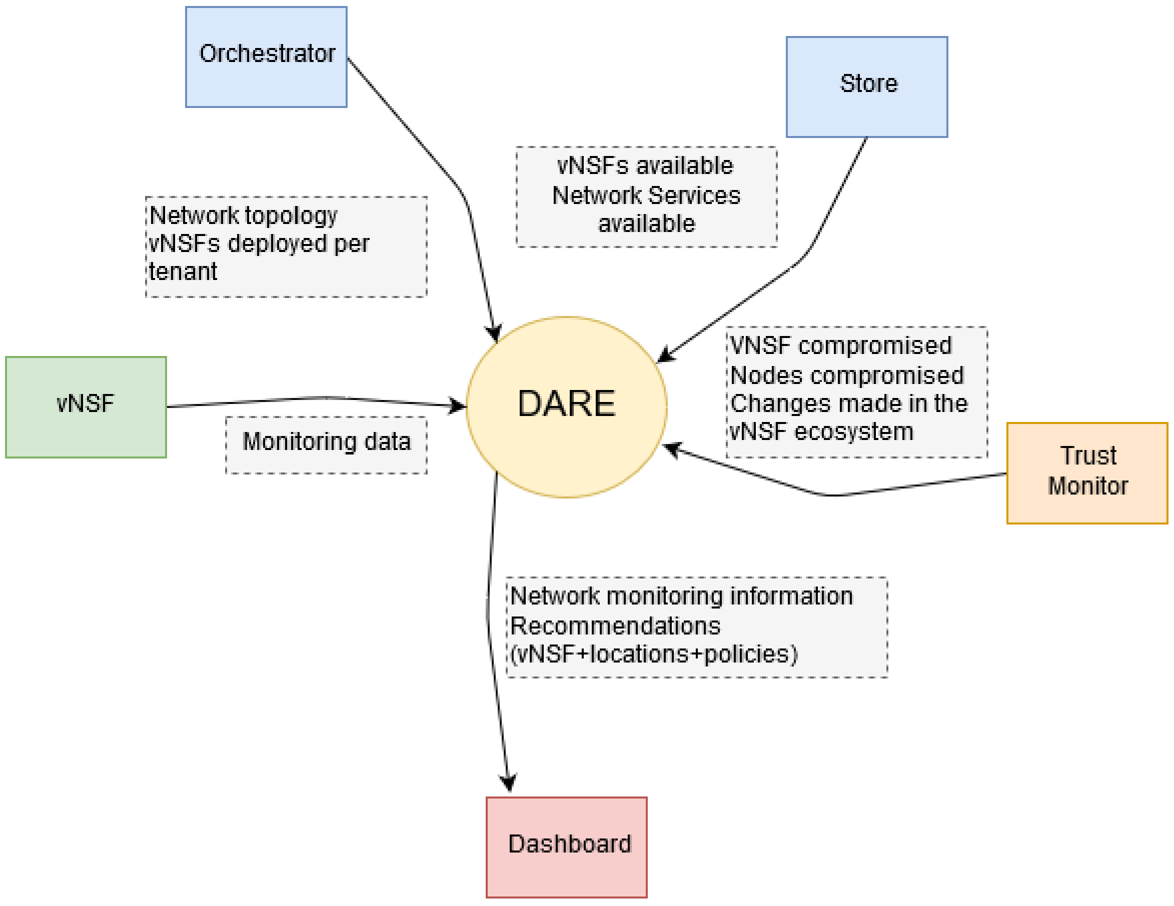
Orchestrator
Store
vNSFs available
Network Services available
Network topology
vNSFs deployed per tenant
vNSF
DARE
VNSF compromised
Nodes compromised
Changes made in the vNSF ecosystem
Monitoring data
Trust Monitor
Network monitoring information
Recommendations (vNSF+locations+policies)
Dashboard

Figure 5: Data flow diagram of the DARE

The DARE will be composed by a central data analytics engine and a distributed set of data collection components. It is worth mentioning that it has been designed following a Big Data approach where the data value elicitation is divided into three different phases, as shown in Figure 6:

1. Data acquisition and storage
2. Data analysis
3. Cybersecurity topologies

In this section, each one of the subcomponents of the DARE will be explained and the differences with the ones envisioned in D2.2 will be detailed.

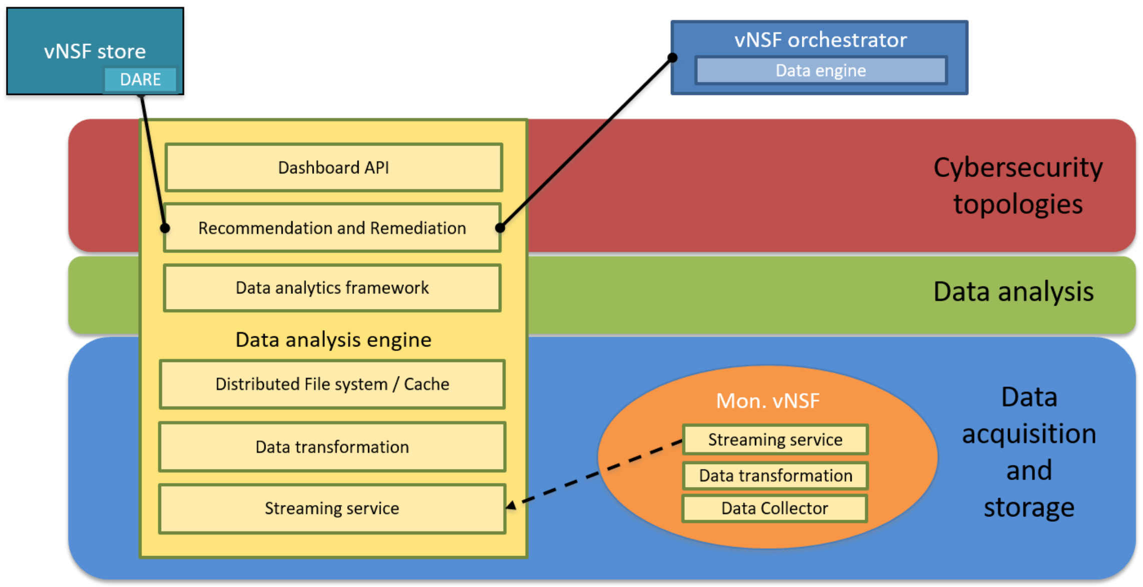

Figure 6: Architecture of the DARE

Data Collector: it is responsible for acquiring the data generated in a monitoring vNSF, using the specific format of the technology provided by such vNSF. The collector is part of each monitoring vNSF and integrated into the diagram for clarification purposes.

Data transformation: it is responsible for transforming the format-specific data into a generic format.

Streaming service: it sends the information from the monitoring vNSF to the data analytics central engine, assuring reliability on the communication.

Distributed File System / Cache: it is responsible for storing the collected data for both, batch (i.e. hard disks) or real-time (i.e. cache) processing.

Data analytics framework: it is responsible for classifying the traffic for anomaly detection using machine learning techniques.

Recommendation and remediation: it proposes, given a specific anomaly or threat detected, a set of vNSFs with the appropriate policies to be deployed in specific places of the network.

Dashboard API: it pushes all the generated information to the Dashboard.

2.2.2. The Security Dashboard

The Dashboard has been designed to be the unique interface with the users of the platform. Hence, the Dashboard must unify all the needs of the users for all the SHIELD use-cases (ISP, SecaaS, and cybersecurity agency).

The low-level architecture presents the different subcomponents that compose the Dashboard as shown in Figure 7.

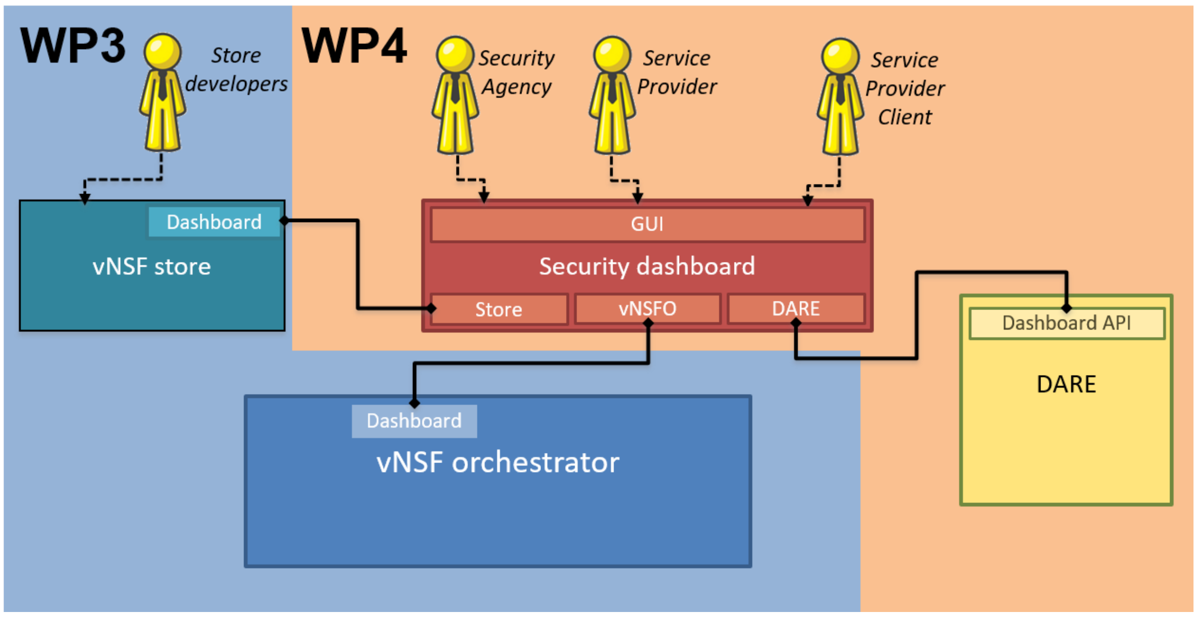

Figure 7: Subcomponents of the Security Dashboard and interactions

2.3. Design

SHIELD is committed to solve three relevant use-cases. Firstly, an ISP which uses SHIELD to secure its own infrastructure. Secondly, an ISP using SHIELD to provide SecaaS to their clients. Thirdly, a cybersecurity agency collaborating with an ISP to research on attacks and vulnerabilities.

This means that the SHIELD must take into consideration the following user interactions:

1. Allow users to deploy network services (after payment, if necessary).
2. Allow users to manage and withdraw network services.
3. Provide insights about what is happening in the network to detect anomalies.
4. Provide a system to display and dispatch DARE recommendations in the form of network services including (completely or partially):
    a. The detailed set of vNSFs recommended.
    b. The description of the specific places of the network where they will be deployed.
    c. The policies that will be used to configure the vNSFs.
5. Allow privileged users to implement monetisation methods.

These features can be modelled as user use-cases (called user cases to avoid confusion with the top-level SHIELD use cases), showing also the components that are related to each one of them (Figure 8). Note that the three SHIELD use-cases share most of the envisioned features. The difference between them is the level of access (e.g. the ISP will have access to all the infrastructure and vNSFs while the ISP client has access only to their subnetwork and to the paid services).

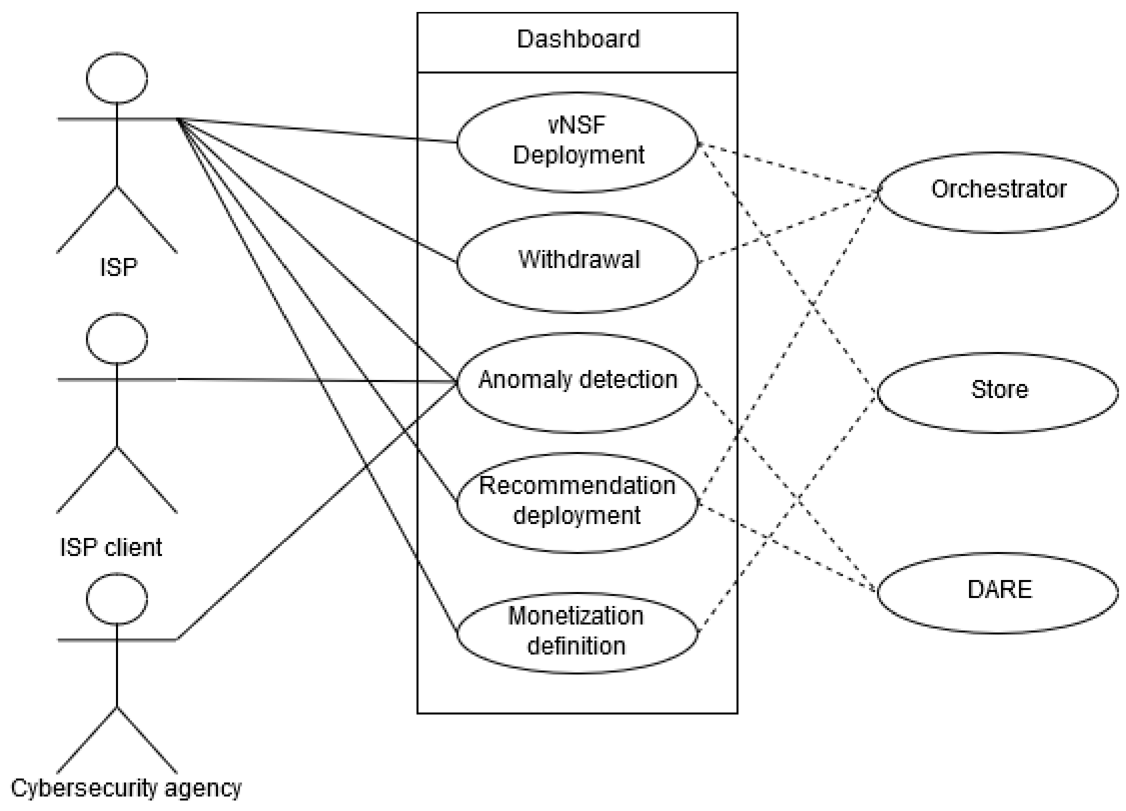

Figure 8: UML user case diagram.

In this section all these user cases will be explained in detail and the differences between the SHIELD use cases will be also stated.

2.3.1. vNSF Deployment and Withdrawal

The processes of deployment and withdrawal of NSs are explained in D3.2 [3] since such tasks are covered by the Store and the Orchestrator components developed in WP3. Since the Dashboard also plays a role in this process (as it interacts with the aforementioned components), we will explain them only from the WP4 perspective.

All the SHIELD use-cases expect the ISP to be able to deploy or withdraw any available NS in any available location in the Network. From the WP4 perspective, the process is very simple, and it is shown in Figure 9.

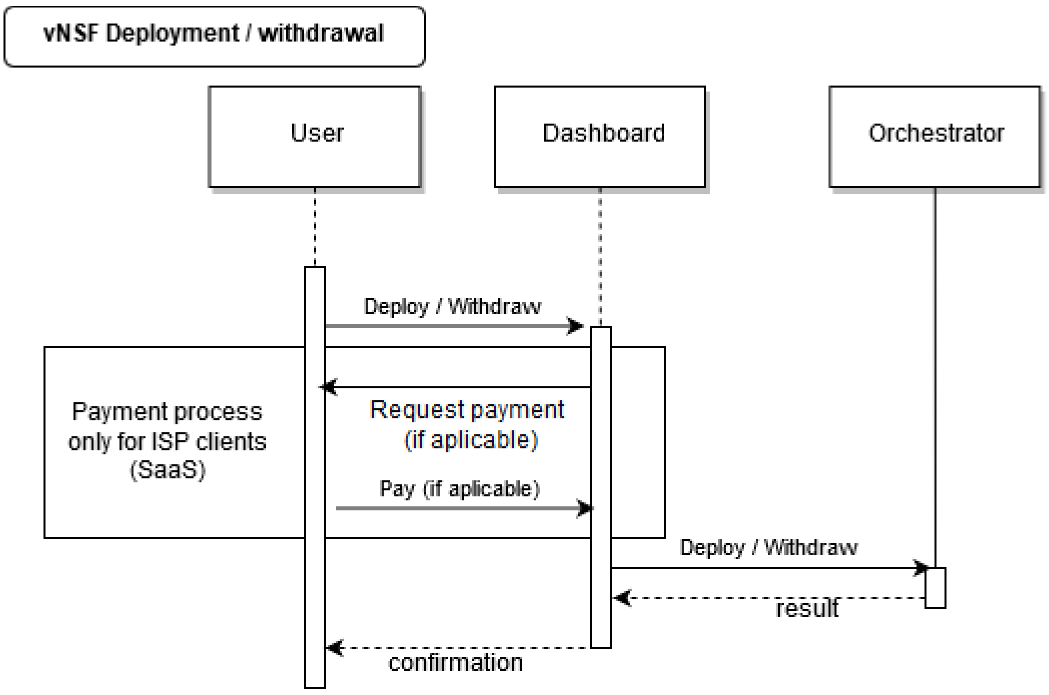
vNSF Deployment / withdrawal
User
Dashboard
Orchestrator
Deploy / Withdraw
Payment process only for ISP clients (SaaS)
Request payment (if aplicable)
Pay (if aplicable)
Deploy / Withdraw
result
confirmation

Figure 9: Flow diagram of the vNSF deployment / withdrawal task.

2.3.2. Anomaly Detection

The process of anomaly detection is divided in two different phases. In phase one, information is collected from the monitoring vNSFs. This information is stored in the Distributed File System subcomponent for further analysis. In the case of real-time analytics, the information will be loaded into cache instead of stored in hard disks, but the workflow is identical. As already explained, the Data Transformation subcomponent can either be distributed, where data is transformed to a generic format (e.g. CSV) before being sent to the (Figure 10); or centralised, so specific formats (e.g. PCAP files) are sent to the central engine (Figure 11).

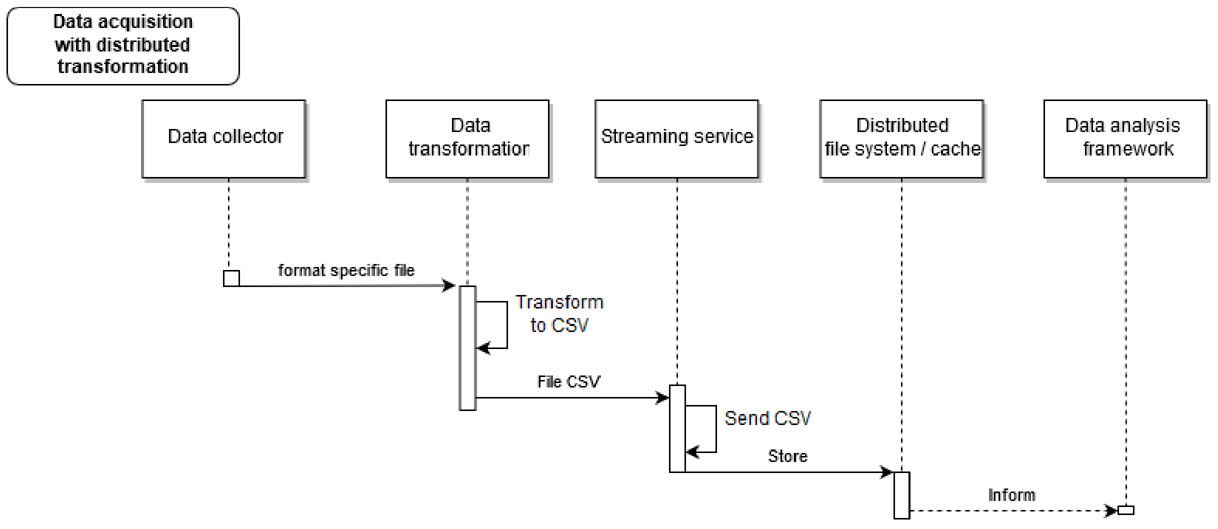

Figure 10: Flow diagram of the data acquisition phase with distributed transformation.

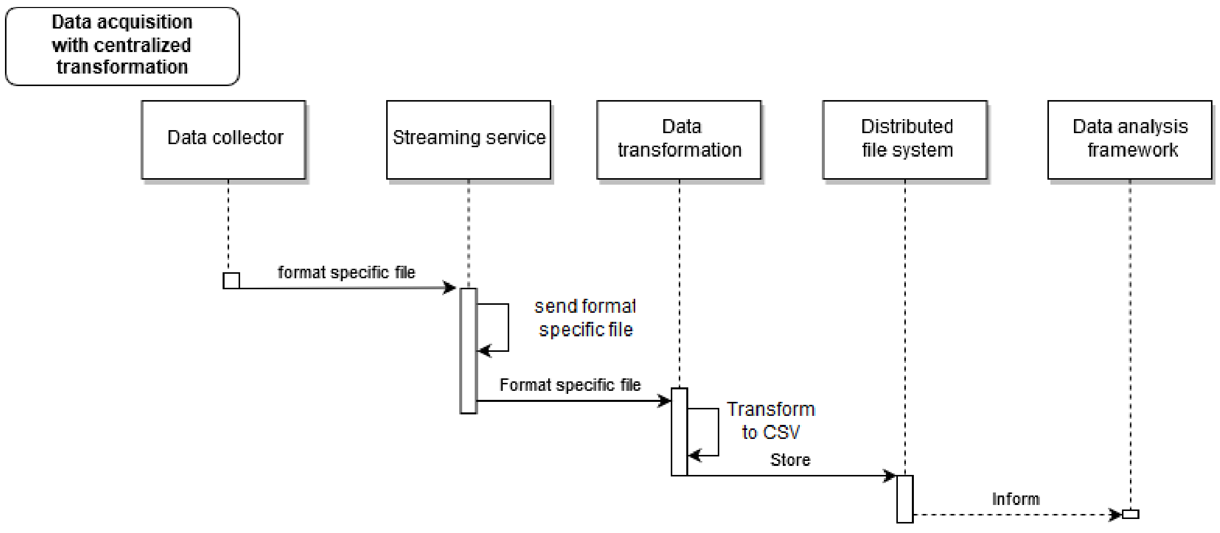

Figure 11: Flow diagram of the data acquisition phase with centralised transformation.

Once the data has been loaded into the distributed file system (either in batch or real time) the phase two (Figure 12) implies that the data analysis framework will use the machine learning algorithms to detect anomalies and inform the remediation and recommendation subcomponent. At its turn, this subcomponent will use this information, together with the information gathered from the vNSF Orchestrator, to provide recommendations to the user in the form of specific network services with the configurations and the locations to be deployed.

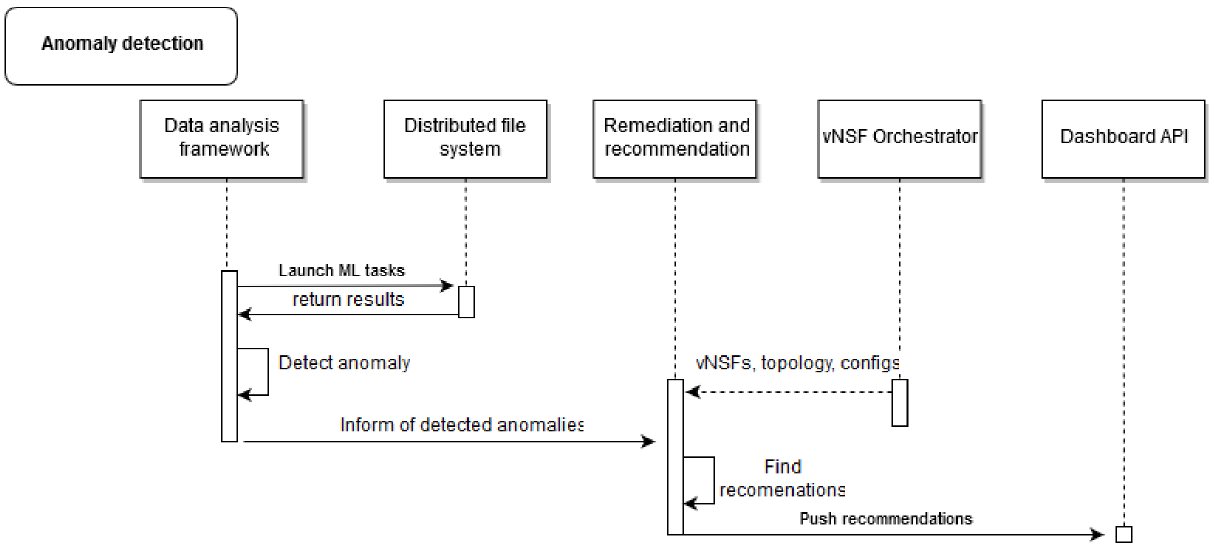

Figure 12: Flow diagram of the anomaly detection phase.

2.3.3. Recommendation deployment

The process of recommendation deployment is driven by the user. Whenever an anomaly is detected in the client infrastructure, a recommendation is generated according to the type of attack and severity. This recommendation must be deployed by the user. The anomaly context information (e.g. type of attack, severity and timestamp) are forwarded to the dashboard and presented to the user along the recommendation. The process is shown in Figure 13.

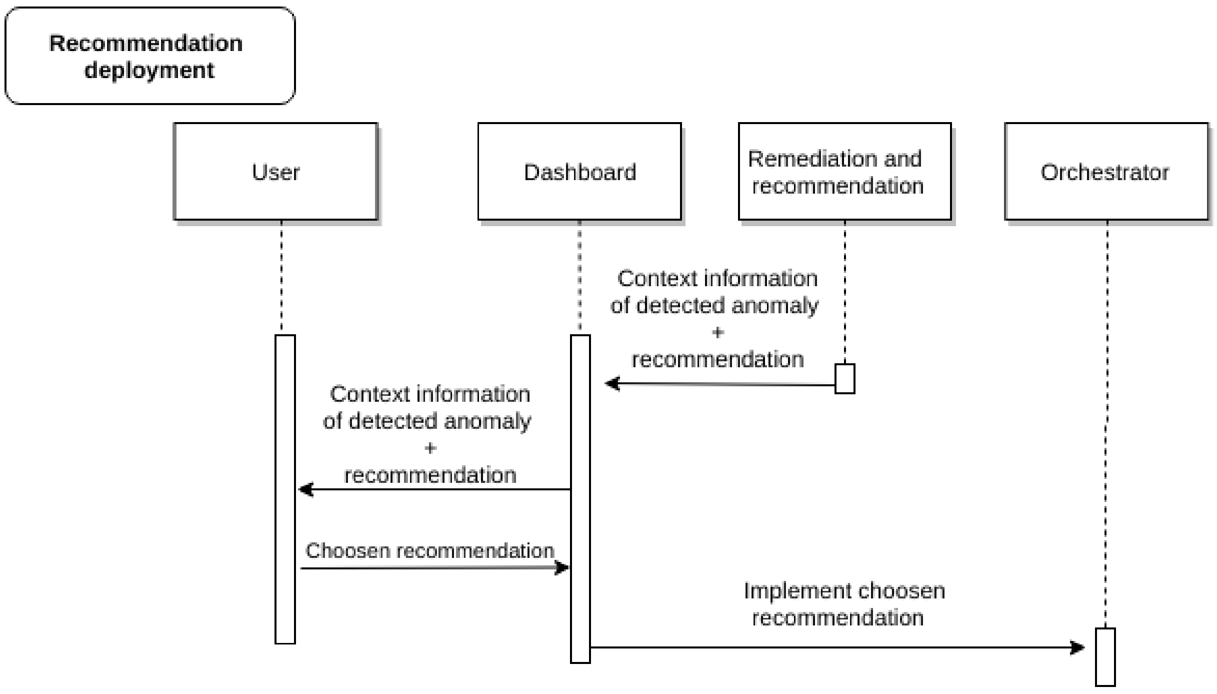
Recommendation deployment
User
Dashboard
Remediation and recommendation
Orchestrator
Context information of detected anomaly + recommendation
Context information of detected anomaly + recommendation
Choosen recommendation
Implement choosen recommendation

Figure 13: Flow diagram of the recommendation implementation.

2.3.4. Monetisation definition

In the SecaaS SHIELD use-case, the ISP client can buy specific SecaaS Services to be deployed in the gateway that joins the client network with the ISP network. It is important to note that, apart from the direct access to the information gathered from its own monitoring service, the ISP will run the monitoring and the remediation actions, allowing the SecaaS Client to monitor the security events.

The monetisation system will be variable. It can be based on a "pay per use", price per SecaaS Service, or "flat rate" that includes all the needed monitoring and mitigation SecaaS to solve a security incident. Business models have been exposed in D2.3 [8] where several billing models have been discussed. In SHIELD, the implementation will focus on the fixed billing model, where a client is charged according to a fixed price per cybersecurity service per month. Moreover, the vNSF developer will be also paid per proprietary vNSF deployed per month.

Even if we will implement only one business model, it is important to include as part of the design some metadata required for possible future improvements. Several parameters must be included. Price, license types, terms of use, period of validation are some potential options. All these parameters will be supported as optional and included in the metadata of the vNSF, within its package. Some examples can be the price defined by the vNSF developer and added as metadata in the vNSF Store (WP3). Another example would include the price defined by the ISP, who can charge the user by adding fixed price to a vNSFs and/or define a subscription system for using the SecaaS. The workflow of this task is shown in Figure 14, BSS is an acronym for Business Support Service, an Application Programming Interface (API) used to implement the business models.

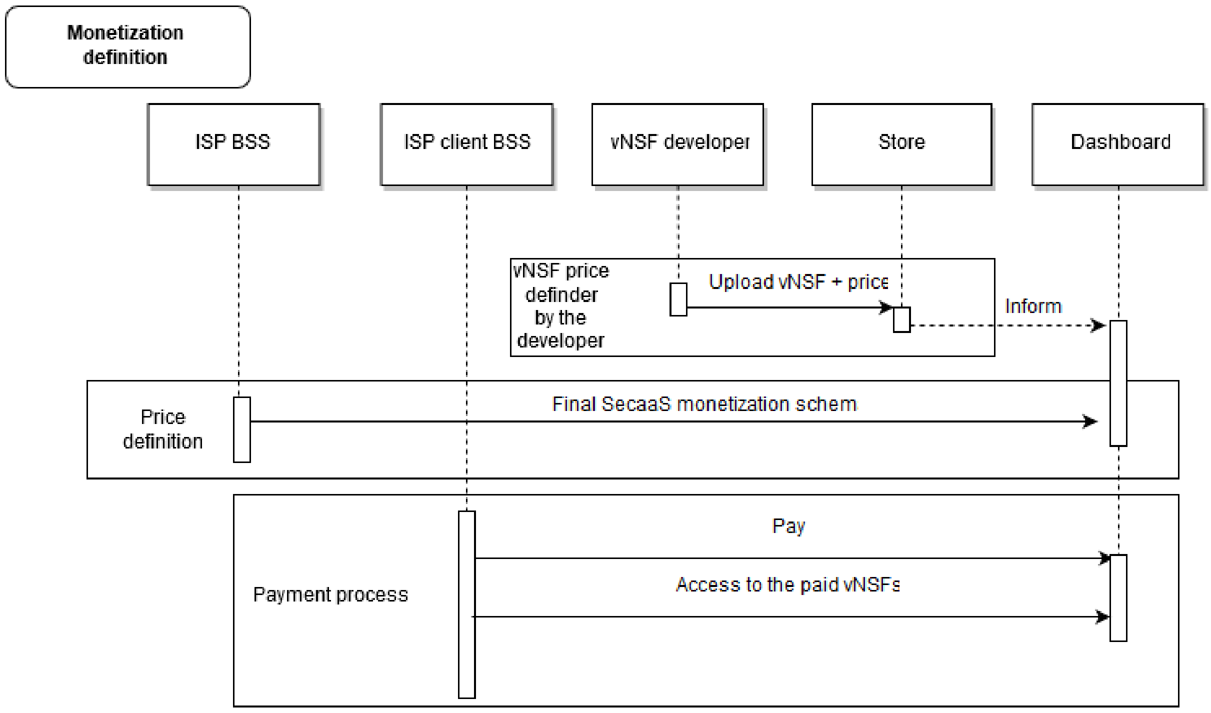

Figure 14: Flow diagram of the monetisation definition.

2.4. Data acquisition and storage Phase

The Data Acquisition phase is responsible for the efficient and reliable capture and storage of various heterogeneous data. It will involve mechanisms and methods to capture and transfer files generated by network tools to the central data analytics engine. This phase is of high importance for ensuring the integrity of the data and their quality in further processing steps.

Heterogeneous network information is captured via specialised vNSFs, which collect overall networking events that are relevant to threat detection. This information is transferred to the central data analytics engine, where it is stored for further processing. This phase will gather, transform and store the acquired network data to a format that can be processed by analytics components.

There are two (2) options for describing the low-level architecture of the Data Acquisition and Storage phase:

- Option 1 – Centralised architecture (Figure 15): only the collection of the data is distributed, while all the other functionalities are centralised in the Data analysis phase.
- Option 2 – Distributed architecture (Figure 16): the data collection and the data transformation are distributed per vNSF and hence, the data is sent to the central engine in a standard format (e.g. CSV).

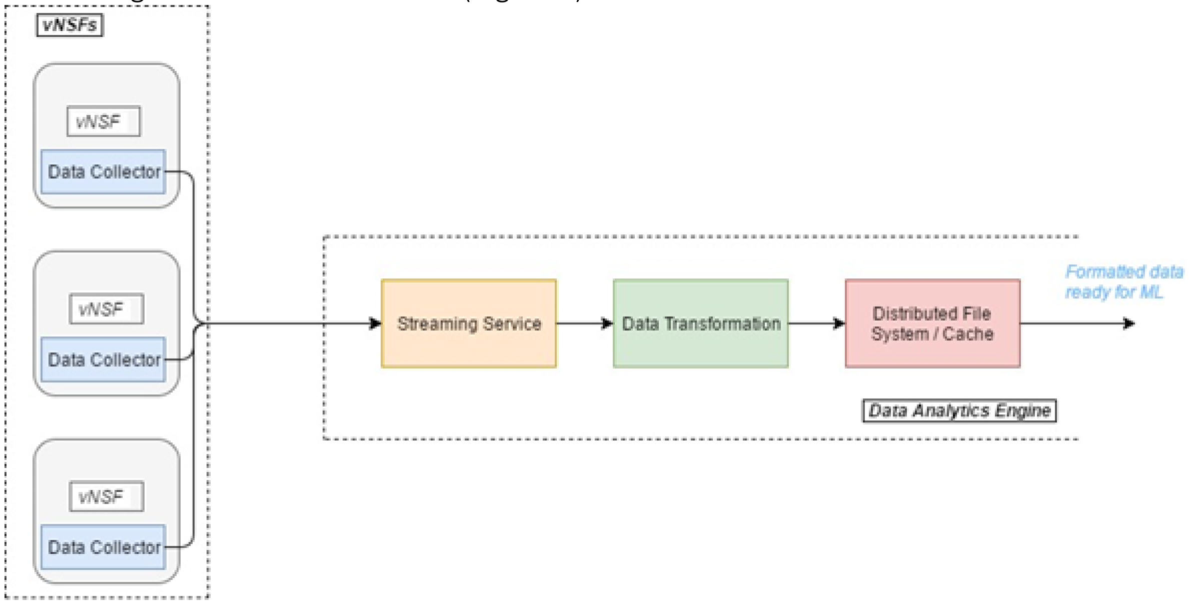

Figure 15: Architecture for centralised acquisition

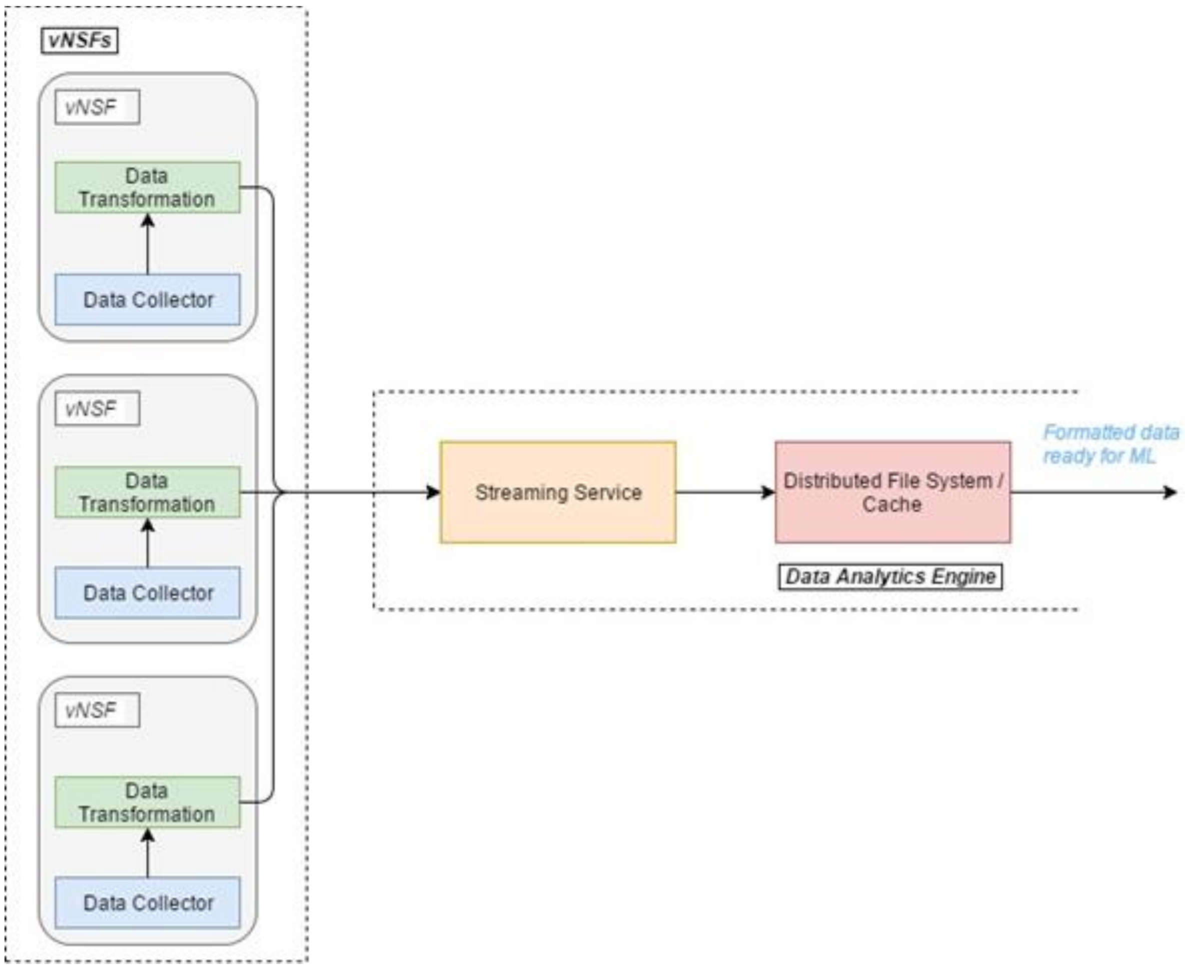

Figure 16: Architecture for distributed acquisition.

The objective of the data acquisition phase is to gather all the network information produced by the vNSFs, transform it into a generic format, ingest it into the data analytics central engine, and store it for further processing. The low-level architecture of the Data Acquisition and Storage phase is divided in four main subcomponents.

Data Collector

The data collector subcomponent is the only one which is distributed (one collector per vNSF) in both options. Each vNSF uses a daemon, called data collector, which is responsible for monitoring the vNSF and detecting new files produced by it.

Streaming Service

The acquisition of network data is achieved via a distributed streaming service that splits the network data into smaller specific topics and smaller partitions, while creating a data pipeline for each topic. It must be reliable and fault tolerant for ensuring the integrity of the data and their quality in further processing steps.

Data Transformation

This is the subcomponent that determines the chosen option for the low-level architecture. If the data transformation is centralised, the architecture will be considered to address option 1, while if it is distributed, the architecture will be considered to address option 2.

In the case of option 1, each pipeline created by the streaming service transfers the stored data to specific daemons which exist inside the central data analytics engine. These daemons are subscribed to a specific topic and partition of the streaming service and transform the raw network data into a human-readable format, by using dissection tools. They are tasked with reading, parsing and storing the data in a specific distributed format to be consumed by the machine learning algorithms.

In the case of option 2, each network data file that is captured by the Data Collector, is sent to specific daemons which exist inside each vNSF (distributed). These daemons transform the raw network data into a human-readable format, by using dissection tools.

Distributed File System / Cache

Once the network data has been transformed, the input is stored in a distributed file system in both the original and modified formats (in the case of option 1) or only the modified (in the case of option 2). The distributed file system is responsible for storing the collected data and making them available, so that it can be accessible by search queries.

2.5. Data analysis phase

The data analysis phase is composed only by one subcomponent, the data analytics framework. This subcomponent features cognitive and analytical functionalities capable of detecting network anomalies that are associated with specific vulnerabilities or threats, offering batch and streaming incident detection. The processing and analysis of large amounts of data is carried out by using Big Data analytics and machine learning techniques. By processing data and logs from vNSFs deployed at specific strategic locations of the network, the data analytics framework can link traffic logs that are part of a specific activity in the network and detect any possible anomaly. In case malicious activity is detected, it informs the remediation and recommendation engine.

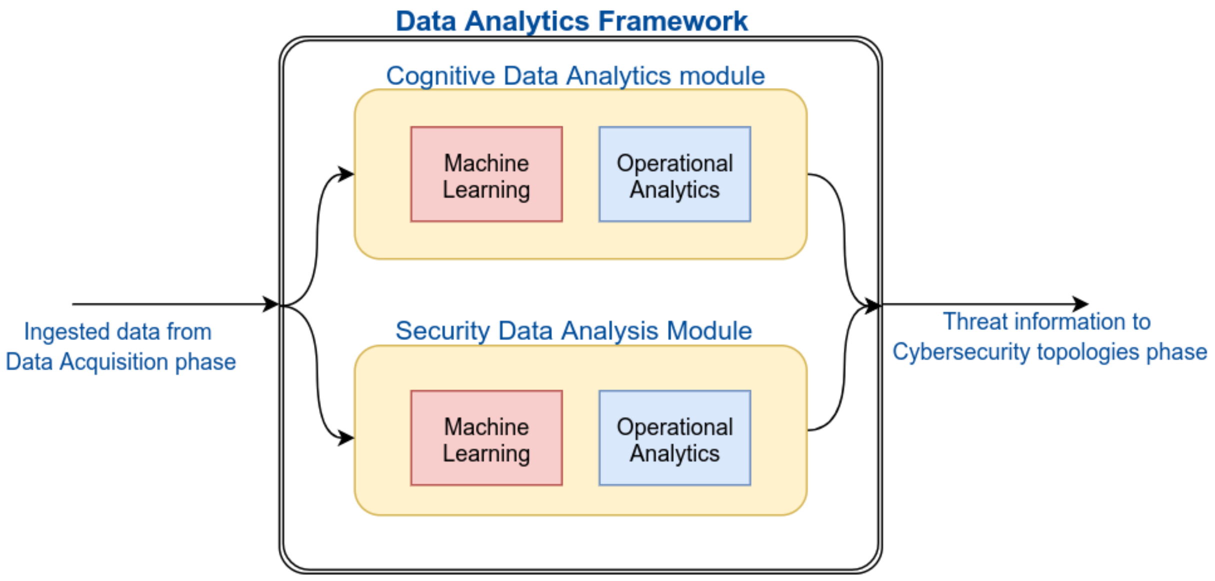
Data Analytics Framework
Cognitive Data Analytics module
Machine Learning
Operational Analytics
Security Data Analysis Module
Machine Learning
Operational Analytics
Ingested data from Data Acquisition phase
Threat information to Cybersecurity topologies phase

Figure 17: Data Analytics Framework overview.

The Data Analysis Engine leverages two different data analytics modules (Figure 17) (while remaining open for the inclusion of others in the future), that use a wide range of complementary detection techniques along with open source frameworks and solutions, a cognitive Data Analytics module based on open-source technologies and a proprietary Security Data Analysis module. The individual functionalities of each module are described in detail.

Since the network data provided by the Data Acquisition phase is required for the functionality of both the cognitive Data Analytics module and the proprietary Security Data Analysis module in the same distributed format, the accessing method is shared between these two modules.

Below we describe the two main Data Analytics modules that comprise the DARE, namely the **cognitive Data Analytics module** and the **Security Data Analysis module**.

Cognitive Data Analytics module

The cognitive Data Analytics module is an entity of elements that can produce packet and flow analytics by using scalable machine-learning techniques. To this end, it involves state-of-the-art Big Data solutions as well as the latest distributed computing technologies to allow batch and stream processing of large amounts of data, scalability and load balancing, utilisation of open data models (ODM) and concurrent running of multiple machine-learning applications on a single, shared, enriched data set. The above technologies will ideally allow for tailor-made security analytics and will lead to predicting attacks by correlating network anomalies to specific threats. The cognitive Data Analytics module consists of two main entities that comprise the overall detection procedure. These discrete entities are shortly referred to as machine learning and operational analytics and are configured either as separate computational nodes (physical or virtual machines) or as a part of a larger distributed computing system. A description of each entity is given below:

- **Machine learning**
  The machine learning entity (Figure 18) is responsible for the detection of anomalies in network traffic that will lead to the prevention or mitigation of potential threats. For this purpose, DARE uses a combination of open-source tools to run scalable machine learning algorithms. The machine learning entity works not only as a filter for separating bad traffic from benign, but also to characterise the unique behaviour of network traffic. It contains routines for performing suspicious connections analytics on flow, DNS, proxy logs, security event data and metrics gathered from the Data Acquisition phase and the built-in Distributed storage system subcomponent. These analytics consume a collection of network events to produce a list of the events that are considered to be the least probable, and these are considered the most suspicious. Below are listed the main types of analytics that are utilised by the machine learning component inside a cluster computing framework.
  Anomaly Detection algorithms: The statistical model being used for discovering incongruences or rare behaviours by examining network traffic.
  Additional algorithms: The machine learning entity is open for the inclusion of additional algorithms to enhance the overall detection capabilities of the platform as well as allow the correlation between the detected anomalies and specific threats by classifying the results of the anomaly detection algorithms.

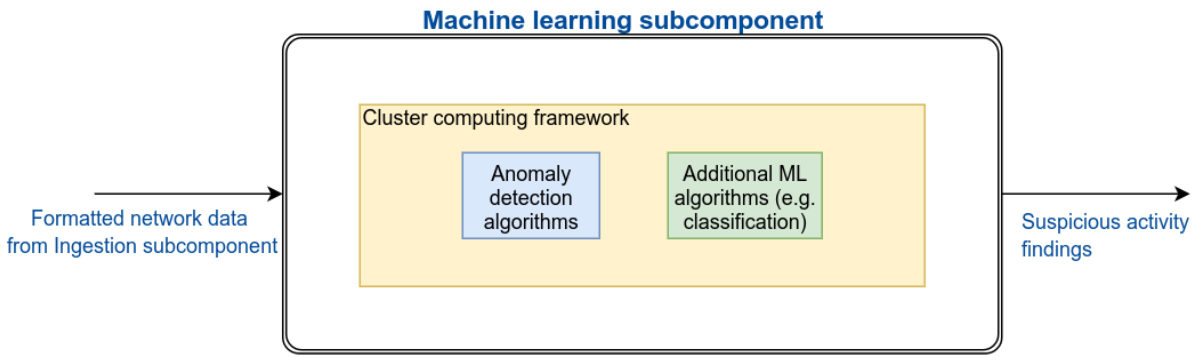
Machine learning subcomponent
Cluster computing framework
Anomaly detection algorithms
Additional ML algorithms (e.g. classification)
Formatted network data from Ingestion subcomponent
Suspicious activity findings

Figure 18: Machine learning entity overview.

- Operational Analytics (OA)
  The OA entity (Figure 19) consists of several context enrichments, noise filtering, whitelisting and heuristics processes to produce a list of the most likely patterns which may comprise security threats. It provides utilities to extract and transform data, by loading the results into output files. It supports basic data types such as flow, DNS and proxy logs that correspond to the most common types of network threats. The output of the OA entity can be used in the Remediation Engine, to provide recommendations to the users or to optionally activate task-specific countermeasures in the form of security functions from the vNSF Store. It also offers a combined view of the above information in the form of information to be pushed to the Dashboard for better visualisation purposes. A description of each element is given below.
  Threat interaction tools: An interactive tool that allows for a comprehensive interaction with the network anomalies detected by the machine-learning component.
  Ingest summary: It presents the amount of network data that has been ingested on the cluster.
  Filtering tools: A set of tools that provide the ability for customised results based on time, source/destination, severity, type etc.
  Whitelisting tools: A convenient set of tools to exclude some of the detected anomalies from the results, thus dealing with potential false-positives.

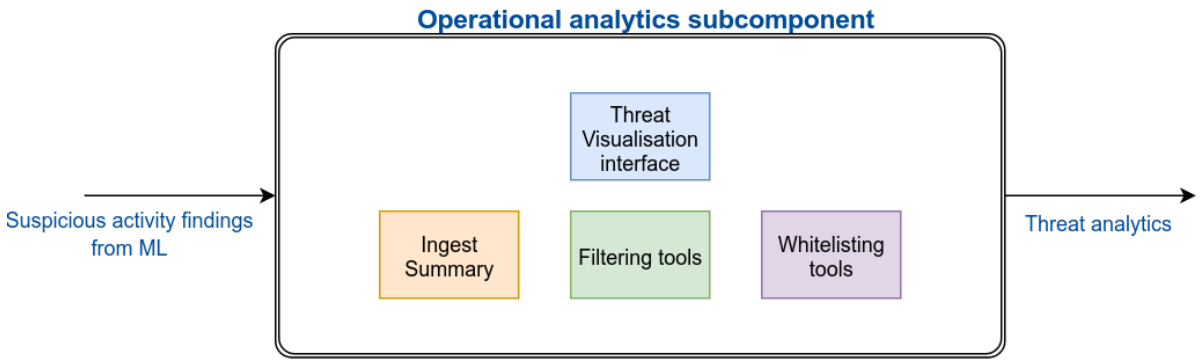
Operational analytics subcomponent
Threat Visualisation interface
Ingest Summary
Filtering tools
Whitelisting tools
Suspicious activity findings from ML
Threat analytics

Figure 19: Operational analytics subcomponent overview.

Security Data Analysis module

The Security Data Analysis module (Figure 20) is an entity based on a combination of Big Data analytics and machine learning techniques that can efficiently process and analyse a vast amount of network data online and automatically discover and classify cybersecurity threats. This engine follows the architecture designed by the commercial network anomaly detector commercialised by Talaia Networks, which has received very good feedback from customers around the world because of its comprehensive detection of network attacks and its low false positive rate.

The Security Data Analysis module has four subcomponents that are described in more detail later. Briefly, this module receives the network data from the distributed storage system /cache subcomponent. The main input used is flow aggregated data (e.g., NetFlow [9]), however, it also can utilise other sources of information (e.g., DNS, network logs). This data is provided to the Event Clustering and the Service Profiling subcomponents that compute several measurements and statistics that are used in the Alarm Correction subcomponent to detect anomalous behaviours related to security issues. Once an anomalous behaviour is detected the Anomaly Classifier subcomponent oversees classifying it among different network attacks (e.g., DDoS, network scan). All this information is then outputted from the Security Data Analysis to the Remediation module.

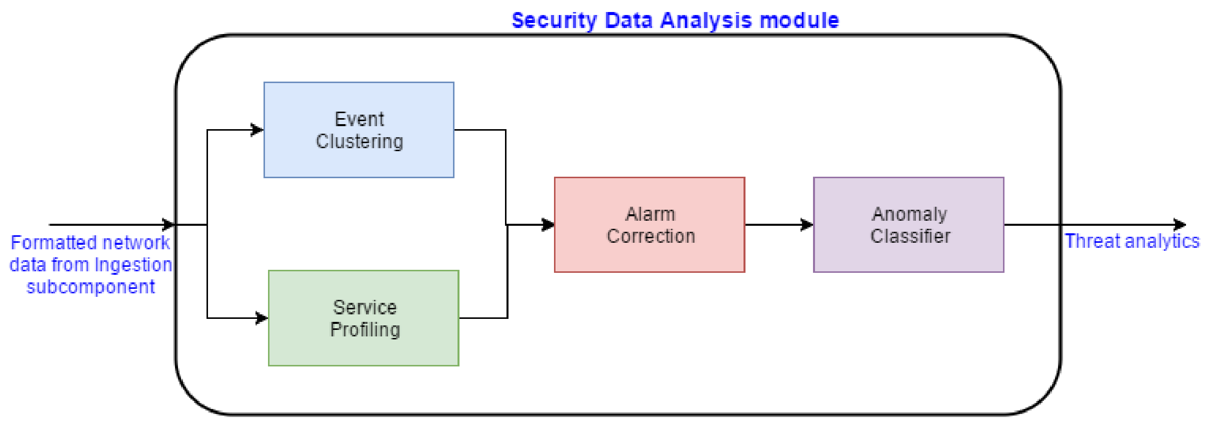

Figure 20: Security Data Analysis module overview.

- Event Clustering
  The Event Clustering entity consists of the adaptation of data mining techniques able to discover multi-dimensional patterns of network usage in the data provided. This entity detects clusters of events in the data which are frequent and share a specific behaviour. The detected clusters, although they can be related to benign traffic, are suspicious of being anomalous traffic that can be related to security issues. The data involved in the cluster and the specific statistics and measurements shared by the events of the cluster are sent to the Alarm Correction to decide if the detected cluster is related to a security issue.
- Service Profiling
  The Service Profiling entity performs a thorough and detailed analysis of the data provided to create profiles of all the services contained in such data. To perform this analysis in real-time, the engine is relying on extremely efficient data structures combined with cutting-edge data mining techniques. The objective of this entity is to identify and understand the behaviour of the different services in the data. In addition, this entity is also able to discover the services that are prone to be affected by networks attacks. Similarly, to the Event Clustering entity, the information related to the services identified is then forwarded to the Alarm Correction entity for further analysis.
- Alarm Correction

The Alarm Correction entity receives as input the output from the Event Clustering and the Service Profiling entities. By comparing and enriching this information with a proprietary methodology, this entity can identify anomalous behaviours in the traffic and discern with very high accuracy between benign and malign behaviours related to security issues. The malign behaviours detected, and the data related to them are forwarded to the Anomaly Classifier entity.

- Anomaly Classifier
  Once a malign anomalous behaviour is detected by the Alarm Correction entity, the Anomaly Classifier entity oversees identifying the type of the anomaly. To that end, the Anomaly Classifier entity uses the measurements and statistics as input for a machine learning technique that can classify the anomaly between different volumetric attacks (e.g., DDoS) and zero-day attacks. All the resulting information is then provided to the Remediation module, that based on the characteristics and the anomaly type can better provide accurate counter-measurements to mitigate the security issues.

2.6. Cybersecurity topologies phase

The DARE includes two subcomponents on top of the data analytics framework which oversee defining the remediation and recommendations actions to be presented to the final user via the Dashboard. A Remediation action consists in a cybersecurity topology responsible for addressing a specific network security threat. These actions will ultimately be translated into a set of vNSFs with proper configuration and deployment location, therefore allowing its instantiation in the secure network environment. Configurations are specified as a set of high-level, technology-independent policies with a uniform description regardless of the targeted vNSF type or implementation.

The recommendation and remediation subcomponent is aware of the current state of the network infrastructure to optimise the security impact of the vNSFs of the different Network Services. This awareness is built upon information regarding running instances for both vNSFs and NSs retrieved from the Orchestrator per SecaaS client. A cybersecurity topology will be generated by a detected threat, which is converted into a high-level abstraction of a remediation recipe. However, the actual remediation is not to be performed directly in this subcomponent, which is not oversee directly modifying the status of the infrastructure, but to be proposed to the SHIELD operator via the Security dashboard (using the Dashboard API), that oversees accepting or declining it. If accepted, the remediation action is applied in the network infrastructure of the SecaaS client through the Orchestrator, which also forwards the request for the translation of policies to low-level configurations, carried out within each vNSF.

2.6.1. The recommendation and remediation subcomponent

The high-level block diagram representing the architecture of the recommendation and remediation engine and its interaction with other SHIELD components is depicted in Figure 21. The engine’s internal subcomponents are described as follows.

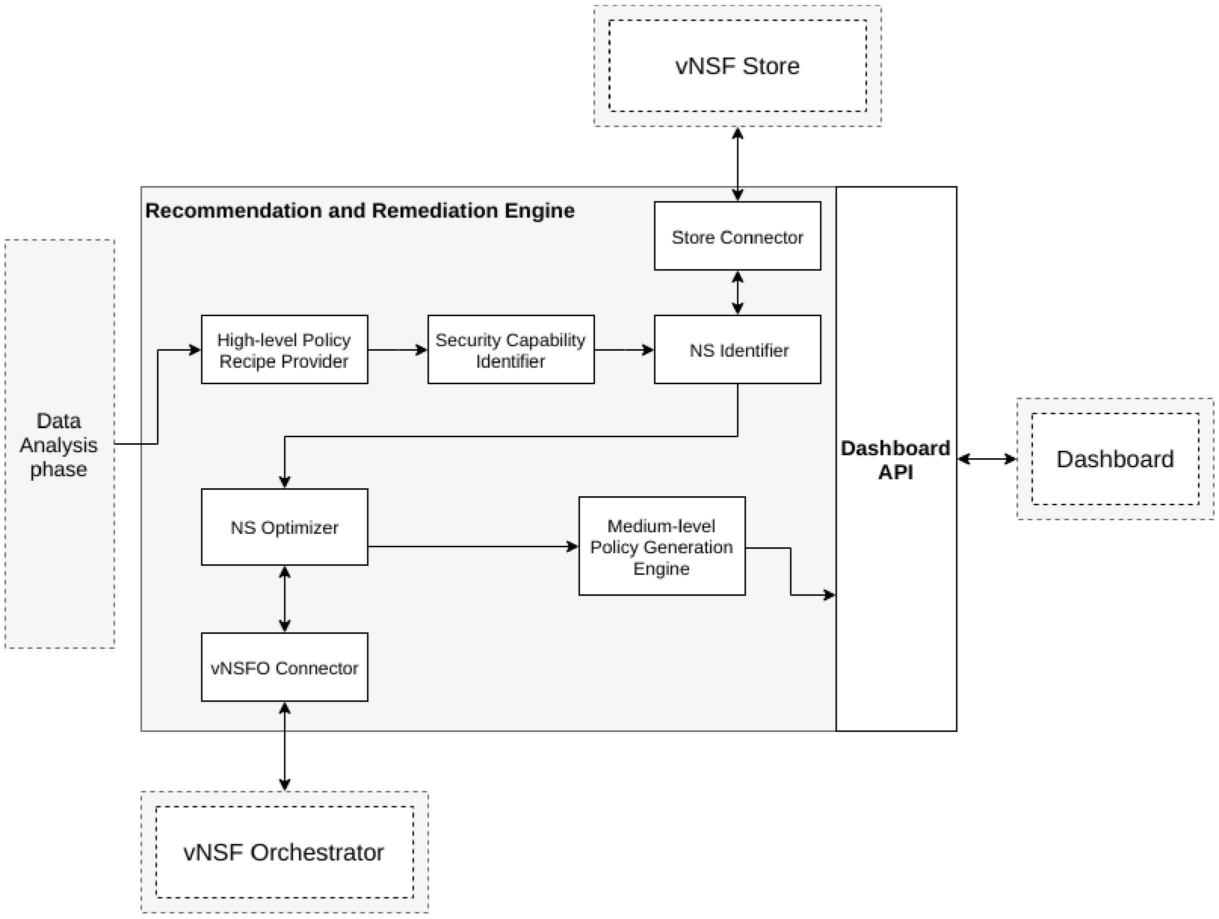
vNSF Store
Recommendation and Remediation Engine
Store Connector
Data Analysis phase
High-level Policy Recipe Provider
Security Capability Identifier
NS Identifier
Dashboard API
Dashboard
NS Optimizer
Medium-level Policy Generation Engine
vNSFO Connector
vNSF Orchestrator

Figure 21: The recommendation and remediation subcomponents.

High-level Policy Recipe Provider

This module receives an event from the data analytics framework of the DARE, containing alerts and contextual information related to an occurring threat. Starting from it, this module will define a "recipe", consisting of a set of security requirements specified in a high-level policy abstraction targeting the mitigation of the detected threat. The recipes will be stored in an internal repository, to allow their update/addition/removal by a security analyst.

Security Capability Identifier

Using the information created by the previous module, this module will be responsible for mapping each remediation high-level recipe to a set of security capabilities. A capability is defined as a basic feature that can be configured to enforce a security action (e.g. address translation, authentication, data protection, authorisation, routing, resource protection, resource analysis).

NS Identifier

This module oversees selecting the sets of vNSFs (the NSs) that match the required security capabilities. To do so, it requires the knowledge of the security capabilities offered by each of the NS in the NS Catalogue, hence it directly interacts with the vNSF Store.

Store Connector

This module oversees connecting the subcomponent with the vNSF Store, by consuming its API to retrieve the information about the available NSs in the catalogue.

NS Optimiser

This module will oversee selecting the best NS that matches the required capabilities, according to an optimisation criterion. In addition, this module will be able of verifying if a NS has been already deployed for the SecaaS client, to optimise the deployment of the remediation by providing only the updated configuration.

vNSFO Connector

This module oversees connecting the subcomponent with the vNSF Orchestrator, by consuming its API to retrieve information about the already deployed NSs for the SecaaS client.

Forwarding Graph Generator

This module oversees describing the list of vNSFs into a topological arrangement that would allow the vNSF Orchestrator to deploy them into the network infrastructure.

Medium-level Policy Generation Engine

This module generates the medium level policy abstraction starting from the Forwarding Graph and the list of capabilities identified to address the security threat. Each vNSF capability is associated to a policy, expressed in an application-independent syntax.

2.6.2. The Dashboard API subcomponent

This subcomponent consists of an interface in charge of presenting the result of the recommendation and remediation decisions to the Dashboard. The information to be provided will consist of an optimised cybersecurity topology and a set of application-independent rules to implement the mitigation.

2.7. Security Dashboard

The Security Dashboard (Dashboard from now on) is SHIELD’s topmost component enabling users and third-party applications to use SHIELD’s internal features. The Dashboard is therefore the entry point of SHIELD solution, seamlessly encapsulating the access and use of all its information and features in this component. Being the only point of access for the SHIELD users, it eases the integration and builds a more secure application, since the access control is more robust and protected. Besides integrating with all SHIELD’s components, the Dashboard is also responsible for the implementation of a set of support features. It will provide user and SecaaS client with management features, billing and monetisation capabilities as well as a remediation subcomponent responsible for persisting and dispatching (upon validation by authorised users) all SHIELD’s remediation suggestions.

Displayed in Figure 22, are the currently envisioned Dashboard's internal subcomponents, considering the requirements and responsibilities associated with it.

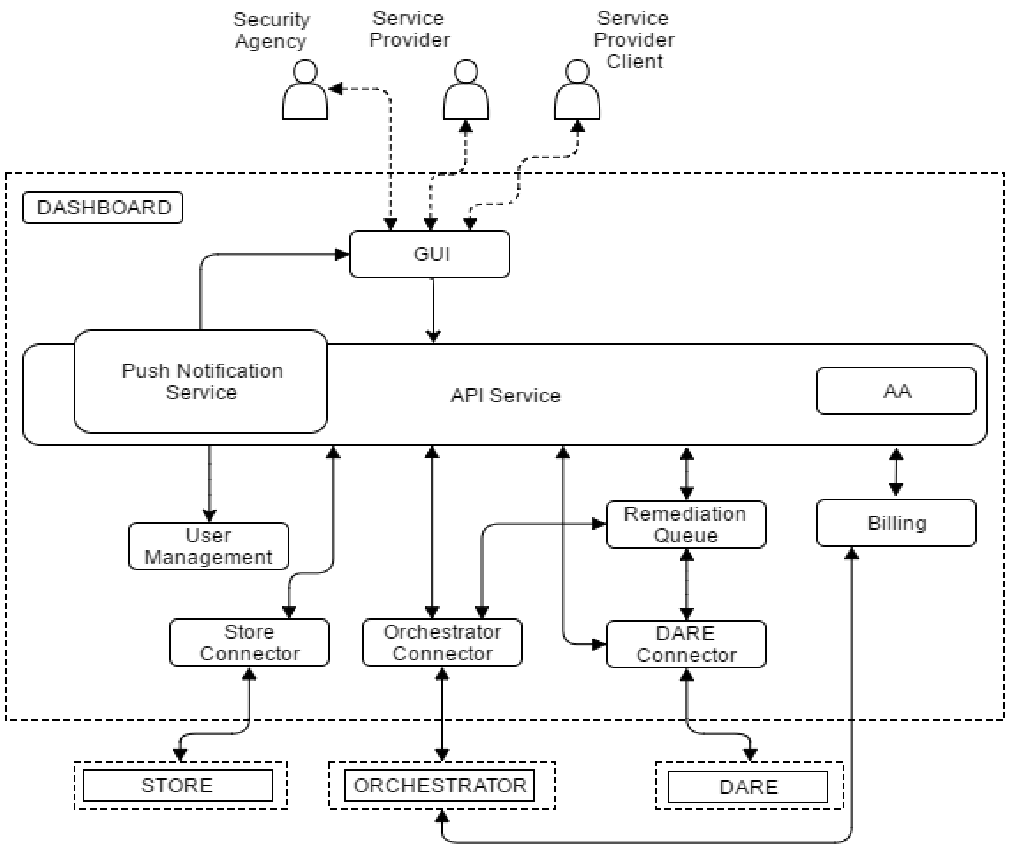

Figure 22: Dashboard internal subcomponents.

As depicted in the previous figure, the Dashboard architecture is divided into multiple subcomponents, each one with a well-defined scope implementing its set of features. Each one of these subcomponents is described below, allowing the understanding of both how this component interacts with other SHIELD components as well as the internal workflows of the Dashboard.

GUI

The Dashboard user interface will provide an appealing and intuitive web-based interface exposing the security-related features to end users. Different permission levels will be attributed to different users, thus allowing the graphical user interface to adapt the provided features to the ranking of the logged user. By using a web browser, end users will then be able to see SHIELD's monitoring system thus perceiving an overview of the security status of their services. Moreover, this interface will also display detected vulnerabilities as well as remediation suggestions that allow to mitigate each detected vulnerability. The end user will be able to analyse the proposed actions of a remediation and decide whether to apply or not the suggestion. SecaaS client, user and billing features will also be present in the security dashboard graphical user interface (taking into consideration each user's permission level) allowing to control the access to SHIELD's features as well as the monetisation of SHIELD's tools. This subcomponent, as illustrated in the previous figure will interact only with API Service as well as Push Notifications Service subcomponent providing an entry point for SHIELD's end users as depicted in the previous figure.

API Service

API Service provides an abstraction from the GUI as well as potential third party applications aimed to interact with SHIELD solution. It interacts with all Dashboard's internal subcomponents forwarding a given user request to an entity responsible either for implementing a feature or forwarding the request to another component. Following what was previously described in the current section, the authorisation of the type of request performed by the user will firstly be validated through authorisation and authentication features provided by AA subcomponent.

Push Notification Service

This subcomponent is responsible to push notifications/events from the bottom layers of SHIELD to either the graphical user interface or third-party applications. The current subcomponent will therefore enable information visible by end users to be updated without the need of the user to request its refresh, therefore allowing interested and authorised end users to always perceive the security status of their services.

AA

AA subcomponent stands for Authentication and Authorisation providing a set of features regarding these two scopes. This subcomponent therefore ensures that SHIELD's available resources are only accessible to users that have the needed rights to access them. To do so, this subcomponent will first validate the authentication and then the authorisation for each request. Authorisation encompasses the steps needed to identify the user responsible for a given request and to check if the user-associated token can access a given resource, therefore enabling different sets of operations to be accessible to different user roles. Only if both, authentication and authorisation are assured, a specific request will be forwarded and access to the Dashboard's internal components will be provided.

User Management

As previously mentioned, the access to SHIELD's internal features will be controlled in each request made to the API Service, taking into consideration the authorisation and authentication level access of the responsible user. Hence, there is a need of a subcomponent that is responsible for the management (creation, edition, deletion) of users in the scope of SHIELD. User Management subcomponent will be responsible to implement this management features ultimately providing and defining the access level of each user in the solution.

Remediation Queue

This subcomponent will be responsible for persisting and providing the suggested actions originated in the DARE's remediation subcomponent. As mentioned before, end users will be responsible for the validation/rejections of these recommendations (unless automatic remediation is selected). Moreover, if a validation is received regarding a given recommendation, this will then be forwarded to the Orchestrator for application through the Orchestrator Connector, therefore allowing SHIELD to act upon and solve a specific security thread.

Billing

Billing subcomponent is responsible for providing a set of features that enable all the SHIELD's monetisation features. Following what will be defined in SHIELD's business model, this subcomponent will allow SHIELD operators to charge for instance a "pay per use", price per vNSF, or "flat rate" whenever a new NS, vNSF or remediation action is applied.

Store Connector, Orchestrator Connector, DARE Connector

Since Dashboard component is envisioned to interact with all SHIELD's components, an abstraction layer of these connections was added in the Dashboard's architecture to minimise the implementation's dependencies across Dashboard's internal subcomponents. These three subcomponents (Store, Orchestrator and DARE connectors) will allow different SHIELD's Dashboards to interact with these three mentioned components.

3. Specifications and Implementation

To drive and properly guide the development of the platform, apart from the overall architecture and design presented in the previous section, there is a need of transforming the requirements elicited in D2.1 [4] and D2.2 [2] into technical specifications for each one of the phases. As stated in D2.1 and D2.2, there are three types of requirements: i) platform requirements (PF), ii) non-functional requirements (NF), and iii) service requirements (SF). As i) and ii) have different implications in each phase in the data value chain (ingestion, analytics, decision and visualisation), they will be transformed into specifications per phase. For iii) we analyse them from a general point of view since they have implications in a general way into the components of the WP4. This transformation is shown in Table 1.

Table 1: General service specifications and fulfillment of service requirements.

Numbering	Title	Description
SF01	Content filtering	A security service COULD provide URL filtering based on different configurable categories (e.g. political, violence, sex, social networks, etc.) in the internet web browsing.
S_SPEC_1	The platform may be enriched with topic modelling algorithms that will provide the necessary insight for the Remediation Engine to proactively filter specific content, depending on the desired configuration provided by the user through the dashboard.	
SF02	Detect/Block access to malicious websites	A security service SHALL control access to malicious websites, such as phishing servers, malware spreading, C&C servers, etc. The user must be alerted and the access to the site could be blocked/allowed depending on the configured policy rule.
S_SPEC_2	The platform will incorporate algorithms that will detect malicious connections and will provide relevant information to the Remediation Engine and the Dashboard. The Remediation Engine will recommend specific medium level policies (MSPL – Subsection 3.3) and/or vNSFs to stop these connections.	
SF03	Security assessments	A security service COULD provide continuous vulnerability assessment on the network, hosts or applications.
S_SPEC_3	The platform may include automated security check routines that will scan the network for vulnerabilities (e.g. open ports) and remediate the threats by recommending the deployment of medium level policies and/or new vNSFs to remediate the vulnerabilities.	

SF04	L4 traffic filtering	A security service SHALL monitor traffic based on configuration rules. Traffic packets are filtered, and specific traffic is either allowed, rejected or blocked based on a predefined set of rules (usually based on source IP, destination IP, destination port, etc.). Commonly called firewall.
S_SPEC_4	The platform will monitor network connections by deploying cognitive analytics and will provide feedback about the level of suspicion of each connection. The Remediation Engine will recommend medium level policies to apply to the vNSFs or directly the deployment of new vNSFs to stop these connections.	
SF05	Central log processing/SIEM	A security service COULD collect and correlate security logs from different legacy user sources and generate alerts. This service is intended to provide the user with a way to process its security logs that are not generated by a vNSF in SHIELD.
S_SPEC_5	All the logs of the system must be centralised ingested into the system and tagged as log information. This log information can be analysed in batch at best effort. Using probability of an anomaly and anomaly classification new alarms can be generated.	
SF06	Malware detection	A security service COULD detect (and optionally clean) files with malware downloaded from Internet.
S_SPEC_6	The platform will incorporate a threat classification algorithm that will correlate network anomalies to specific threats, including malware detection. Remediation Engine can recommend new vNSFs or new policies to existing vNSFs to stop connections from the infected machine and isolate it from the network. Later, the malware may be cleaned, or user will be requested to manually clean it.	
SF07	Spam protection	A security service SHALL protect against unwanted emails, based on source reputation lists and content analysis.
S_SPEC_7	The platform will incorporate a threat classification algorithm that will correlate network anomalies to specific threats, including spam activities. Network connections can be stopped by using middle policies and/or vNSFs recommended by the Remediation engine.	
SF08	DoS Protection	A security service SHALL protect against volumetric Denial of Service attacks. Detect the

		DoS attack and divert the traffic for filtering. Forwarding the good traffic flows to the destination.
S_SPEC_8	The platform module will incorporate a threat classification algorithm that will correlate network anomalies to specific threats, possibly including volumetric DoS attacks. Remediation Engine can recommend the deployment of new middle policies (e.g. specific IPSs to be isolated) and or new vNSFs (e.g. firewalls).	
SF09	Intrusion Detection/Prevention System	A security service SHALL detect attacks with a wide range of techniques such as network flow or behaviour analysis and deep packet inspection. Allow traffic flows according to IPS rules. Monitor traffic network traffic at OSI layer 7 and generate alerts for security policy violations, infections, information leakage, configuration errors and unauthorised clients.
S_SPEC_9	The platform will analyse the main types of ingested network traffic (NetFlow, DNS, proxy) and will combine the detected results with those provided by network monitoring vNSFs. The Remediation Engine will be able recommend to the user either to deploy new IDPS vNSFs or middle policies to the already deployed ones.	
SF10	Honeypots	A security service COULD provide a Honeypot service that simulates or impersonates specific services (e.g., Windows computer, Web server, IoT or SCADA device, etc.) to detect malicious behaviours in the network.
S_SPEC_10	The platform will incorporate a threat classification algorithm that will correlate network anomalies to specific threats, possibly being able to distinguish incidents that would require the deployment of a Honeypot service. The Remediation Engine will be able to recommend the deployment of a Honeypot vNSF in a specific PoP and redirect the traffic there through the deployment of L4 or L7 filtering vNSFs or the configuration of already existing ones using medium level policies.	
SF11	Sandboxing	A security service COULD provide a sandbox service for executing and analysing programs. Must provide the possibility to install different OSs.
S_SPEC_11	The platform will incorporate a threat classification algorithm that will correlate network anomalies to specific threats, possibly being able to distinguish incidents that would require the deployment of a sandbox service.	

SF12	VPN	A security service COULD provide a secure tunnel service to connect the branch of a client with users in Internet or other branches.
S_SPEC_12	The platform will incorporate a threat classification algorithm that will correlate network anomalies to specific threats, possibly being able to distinguish incidents that would require the deployment of a secure tunnelling service.	

Apart from the transformation of the requirements into specifications, this section also defines the specific technologies that will be used to develop every subcomponent. Although each phase exposes several technologies that can be used to fulfil the specifications (and hence the requirements), Apache Spot [1] has been selected as the main technology for the entire DARE. Apache Spot is an in-development, open-source platform for network telemetry and anomaly detection. Apache Spot provides tools to accelerate the ability to expose suspicious connections and previously unseen attacks using flow and packet analysis technologies. It also features a built-in ingestion subcomponent that is responsible for handling and transferring the raw network data into the data analytics engine. Spot is built over very mature technologies like:

- Cloudera CDH [10] for data ingestion and storage (which uses Hadoop HDFS [11] and Apache Hive [12]). CDH is an Open Source platform distribution that helps to perform end-to-end Big Data workflows.
- Spark [13] for machine learning and streaming. Apache Spark is a fast and general engine for large-scale data processing.
- ReactJS [14] and Flux [15] for the web components. ReactJS is a JavaScript framework for building user interfaces. Flux is the application architecture that Facebook uses for building client-side web applications.
- IPython [16] for the Spot virtualisation server. IPython is a command shell for interactive computing in multiple programming languages.
- GraphQL [17] for query data from HDFS Parquet [18] files. GraphQL is a query language for custom API's, and a server-side runtime for executing queries by using a user-defined type system.
- Hadoop [19] for distributed file system. Apache Hadoop allows for the distributed processing of large data sets across clusters of computers using simple programming models.
- Hive for data storage. Apache Hive facilitates reading, writing, and managing large datasets residing in distributed storage using SQL.

Using Spot, SHIELD will benefit in several aspects:

- Leveraging the on-going efforts of the community,
- Taking advantage of the dissemination potential offered by Spot,
- Increasing the exploitation possibilities of SHIELD,
- Collaborating with experts on cybersecurity to provide better solutions,
- Following a methodology and good practices in code development.

On the other side, Spot will also benefit by the SHIELD developments since they will help the project to quickly achieve a more mature state. The choice of an integral, open-source and active cybersecurity solution like Spot goes perfectly in line with the concept an Innovation Action. It must be stressed out, at this point, that Spot is still far from suitable to be adopted "as-is" in SHIELD; in the context of SHIELD, Spot will be just used as the starting point for the DARE developments. The SHIELD team has already identified several extensions, which will be needed to the core Spot platform to fulfil SHIELD requirements, mostly related to functionalities such as: mitigation capabilities; near-real-time operation; classification of threats; optimised operation in an NFV environment; and enhancement of the data model to support for more types of information. These extensions are described in detail in the following sections.

In Figure 23, the subcomponents envisioned in the architecture are shown together with the technology that will be used to develop them.

It is worth to mention that Spot is not "yet another IDPS" but a platform that offers to the clients the possibility to develop their own machine learning engines. This approach allows SHIELD to focus the effort on the innovative algorithms that will be used for threat detection and mitigation.

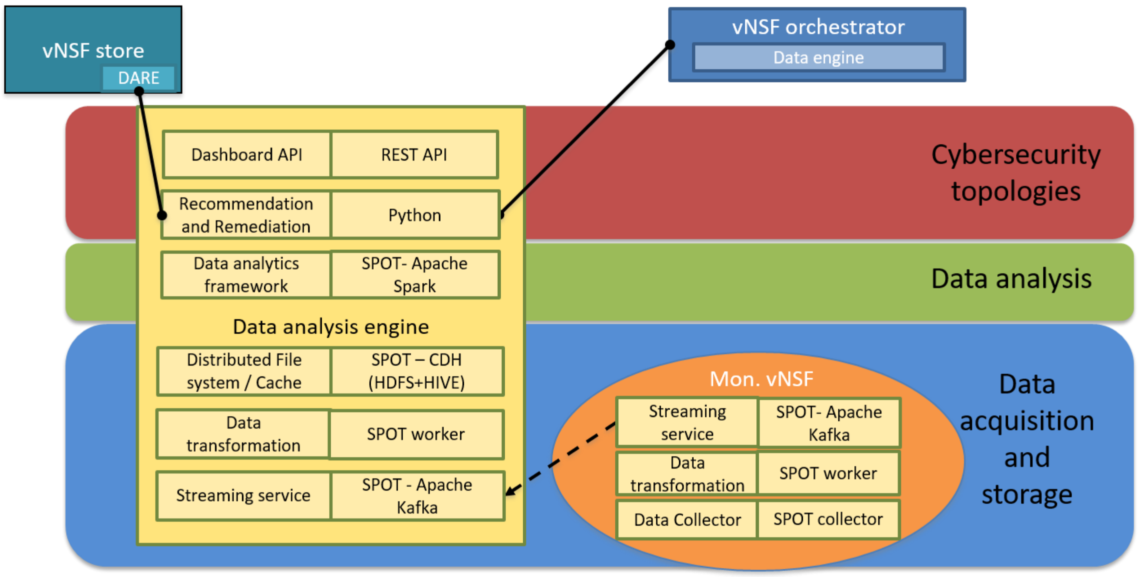

Figure 23: List of subcomponents and used technologies

3.1. Data acquisition and storage

The Data Acquisition and Storage phase involves mechanisms and methods to capture and transfer heterogeneous network information, from the monitoring vNSFs, into the central data analytics engine. This phase is described by two different low-level architectures, the centralised and the distributed (Section 3.2).

Specifications

The requirements elicited in D2.2 are here transformed into specifications from the point of view of the Data acquisition and storage phase (Table 2).

Table 2: Specifications of the Data Acquisition phase and fulfillment of requirements.

Req. number	Requirement Name	Requirement description
PF04	Security data monitoring and analytics	The platform SHALL be able to collect and analyse metrics and logs from the vNSFs in real time in order to detect security incidents
I_SPEC_01	The Data Acquisition and Storage framework utilises collector daemons to detect, capture and transfer heterogeneous network data from vNSFs into the central data analytics engine through a streaming service in real time. Worker daemons are also used, to transform the collected data into human readable format before storing it to a distributed file system. After being stored in the distributed file system, the data is available for further processing analysis.	
PF16	Historic reports	The platform SHALL generate reports of past incidents based on historic data.
I_SPEC_02	Report generation is supported by using the abstraction layer of the Data Acquisition and Storage framework, to run queries on tables, where the historic data is stored.	
PF22	Management communication security	The platform SHALL encrypt all the management communications.
I_SPEC_03	Dispatch of data to the DARE will be authenticated and encrypted.	
NF01	Response time	The platform SHALL report the incident within a relatively short time (in the order of seconds).
I_SPEC_04	The Data Acquisition and Storage framework has a distributed architecture, which allows fast data access and processing and returns results in a relatively short time (in the order of seconds).	
NF04	Data Volume	The platform SHALL be able to handle data in the order of Terabytes.
I_SPEC_05	The Data Acquisition and Storage framework is fully scalable and can handle increasing data volume by adding more data nodes to the infrastructure.	
NF06	Performance factors	The platform SHALL offer an availability-related performance similar to carrier grade system. It includes recovery time and redundancy capability.

I_SPEC_06	The technologies used to implement the acquisition and storage phase will allow redundancy (e.g. N+1) and resilience against failures.	
NF07	Compliance to standards	The platform SHALL conform to well-established standards, in particular with respect to data export (e.g. STIX) and input (e.g. NetFlow).
I_SPEC_07	The ingestion chain will support NetFlow format as input, as well as various log formats.	
ERC01	Access to and portability of personal data	All components that process and/or store personal, identifiable information SHALL provide data subjects with a way to access and review their personal data.
I_SPEC_08	The storage component will expose methods for retrieving data by IP address and/or by URL parts (which can be possibly associated to persons)	
ERC02	Data rectification and erasure	All components that process and/or store personal, identifiable information SHALL provide data subjects with a way to request that their data be rectified or erased.
I_SPEC_09	The storage component will expose methods for deleting data by IP address and/or by URL parts (which can be possibly associated to persons)	

Implementation details

The centralised architecture of the Data Acquisition and Storage phase (refer to Figure 15) will feature a built-in ingestion framework of the Apache Spot platform, which will be responsible for handling the raw network data that will be transferred directly to a specific path of the data analytics engine. Ingestion framework consists of several edge nodes, running on Linux OS and handling the incoming network traffic. It also supports Apache Kafka [20] as a streaming platform, for handling all the real-time network data feeds.

Inside each vNSF there are daemons, called data collectors, which monitor the vNSF for new files with network data generated by it. In the case of centralised architecture, once new data files are detected, collectors capture them from local file system and publish them to Apache Kafka. Apache Kafka splits the raw network data into specific topics and smaller partitions, while creating a data pipeline for each type of data. Each pipeline sends the network data, stored by data collectors, to specific daemons, called workers. Workers are running in the background inside the data analytics engine, as part of the data transformation subcomponent. Each worker is subscribed to a specific topic and partition of Apache Kafka. It reads raw network data from the partition, decodes and translates it into comma-separated files (CSV), by using dissection tools. Once the data has been transformed, worker stores the input in HDFS with both the original and human-readable format, making it available to Hive tables, so it can be accessible by SQL queries.

The Data Acquisition and Storage phase also supports a distributed architecture (refer to Figure 16). The main difference between the centralised and the distributed architecture is that in the centralised one, collectors are running inside the data analytics engine, while in the distributed one, there is one collector per vNSF. When a vNSF generates a new data file, the collector of the vNSF will detect it and will fork a new child process. The child process will decode and translate the raw network file into a comma-separated output with specific structure. Then, using the Apache Avro serialization framework, it will convert the processed output to an avro-encoded format and will send it to the Apache Kafka streaming platform. The collector does not publish the entire processed output to Kafka, but divides it into smaller chunks of bytes, providing a steady flow of data through Kafka. In addition, using parallel processes, delays are significantly reduced when there is a huge load of incoming network files, optimizing the performance of the collector. Published messages in the Kafka cluster are consumed by worker daemons in the analytics engine. A worker process listens to a specific partition of the Kafka topic, consumes messages, deserializes and stores them in HDFS, to be imported directly into Hive tables. Unlike the centralised architecture, which stores both the original and the human-readable formats inside the data analytics engine, only the transformed version of the files is stored in the distributed architecture.

As the Data Acquisition and Storage phase will be based on the Apache Spot platform, the heterogeneous network information that will be captured via specialised vNSFs shall be compatible with the following structure and format:

- NetFlow [9]: data files which contain network traffic as it enters or exits from a monitoring interface. By analysing NetFlow data, a network administrator can determine things such as the source and destination of traffic, class of service and the causes of congestion.
- DNS: PCAP files (packet capture) [21] out of DNS servers. PCAP files contain network packet data, created during a live network capture. By analysing PCAP files, vital information can be retrieved regarding the monitored network and its characteristics.
- Proxy: popular proxy format logs will be supported like the bluecoat format.

By default, Apache Spot uses specific open-source decoders to transform raw network data into comma-separated files. According to the data type, a different dissection tool is being used. For flow traffic, a modified version of nfdump [22] is used to dissect the flow packets into comma-separated files. For PCAP files (packet captures) the TShark [23] tool is used with a combination of options to generate the comma-separated files. The proxy log files are parsed using Apache Streaming [24] before being inserted into Hive tables.

Apart from the above, it is possible to take advantage of other data types produced by the vNSFs, such as possible alerts, to collect extra information which could be helpful in further processing by the machine learning algorithms (e.g. data obtained from the hardware attestation). Apache Spot's Ingestion Framework does not support those type of data and for this reason a new submodule has been implemented under SHIELD's scope. New collectors capture alerts produced by the vNSFs and new workers store these alerts to HDFS. In addition, Apache Kafka's configuration needed to be extended to support these types of data. This submodule can be used in both centralized and distributed architecture, without any further modification of the original implementation. Moreover, along with alerts, measurements could also be sent on the load and usage of each vNSF, to prevent machine failures or data loss.

By gathering data from different vNSFs in the central infrastructure, it is important to know which vNSF has sent each data record. For this reason, a new feature has been added in the current implementation: the Universally Unique IDentifier (UUID) of the current vNSF will be sent among the data to know from which machine these data originate. This information is also stored in Hive tables, to be accessible in the future if needed.

Note also that a collector will be implemented inside the Trust monitor consuming the API provided by it.

Comparison with similar technologies

Both centralised and distributed architectures use Apache Kafka as streaming service. Kafka is a unified platform for handling all the real-time data feeds. It supports low latency message delivery and guarantees fault tolerance in the presence of machine failures. Kafka is distributed, partitioned, replicated and can scale easily without any down time.

Another streaming service that could be used is Apache Storm [25], a free and open-source, distributed, real-time computation system. Storm is designed to process vast amount of data in a fault-tolerant and horizontal scalable method. It is a streaming data framework that has the capability of high ingestion rates. However, Storm is mostly a computation unit, meaning it can execute all kind of manipulations on real time data in parallel. In our implementation, all the computations are taking place inside the data analytics engine and not on streaming process. In addition, Storm does not have the ability to share data with multiple systems, something that may be useful in the future, since Kafka can provide the same data to multiple storage systems (e.g. a new data analytics engine that does not use Hive but uses Pig).

Storage is done in Hive tables over HDFS. Using Hive, it is possible to launch SQL queries over a Distributed File System like HDFS which facilitate the work of accessing the data. A possibility is to use Spark directly, which works over HDFS without the need of any abstraction layer like Hive. However, this will imply that access to data will be more difficult since SQL like language cannot be used. Apache Pig [26] is another abstraction layer on top of HDFS that facilitates the task of accessing the data. Pig is flow-based, instead of table-based like Hive. Although Pig is a possibility, Spot uses Hive because of the convenience of developers that are used to SQL.

3.2. Data analysis phase

The DARE will leverage two different data analytics modules that will export their findings to a shared Remediation engine, to produce optimal results. The Cognitive Data Analytics module will be based on the Apache Spot platform, an in-development, and open-source project for network telemetry and anomaly detection. The Security Data Analysis module will implement a version of Talaia’s proprietary network visibility solution based on the SecaaS architecture. The aforementioned modules will be modified and functionally enriched, with respect to the fulfilment of the SHIELD’s requirements. The Remediation Engine will exploit the Data Analytics Engine’s output to detect security incidents and will recommend the triggering of actions to mitigate the threats, utilizing open-source technologies and implementing the security policies defined in the next section. Next follows a specifications and implementation subsection for each module involved in the DARE.

Specifications

The requirements elicited in D2.2 are here transformed into specifications from the point of view of the Data analysis phase (Table 3).

Table 3: Specifications of the Data Analysis phase and fulfillment of requirements.

Req. number	Requirement Name	Requirement description
PF04	Security data monitoring and analytics	The platform SHALL be able to collect and analyse metrics and logs from the vNSFs in real time in order to detect security incidents
A_SPEC_01	The platform will include worker nodes that will utilise the ingested network traffic to implement batch (historic) and streaming (real-time) machine-learning algorithms to detect security threats.	
PF08	PF08. Platform expandability	The SHIELD platform offers well-documented APIs and interfaces as well as SDKs and guidelines so that third parties can easily develop new security functions and services.
A_SPEC_02	The developed platform will be developed using open-source and each module will be isolated through APIs. Several analytics engines can be used. Specifically, two analytic engines will be developed. One open source and based on cognitive principles and another one privative and based on pattern discovery techniques.	
PF13	Mitigation	The platform SHALL be able to trigger, in the case of an event, proper actions in order to mitigate the threat.
A_SPEC_03	The platform will provide information of the detected threats to the Remediation engine, to initiate the mitigation procedure.	
PF16	Historic reports	The platform SHALL generate reports of past incidents based on historic data.
A_SPEC_04	The detected threats from this phase will be stored in the distributed file system and will be available for processing and reporting.	
PF17	Interoperability	All interfaces of the vNSFO, the vNSFs and the DARE are publicly documented and compliant to open standards to the maximum possible extent.
A_SPEC_05	The Cognitive Data Analytics module is based on open-source state-of-the-art technologies that comply to open standards and industry's best practices. The proprietary engine is used as a “black box” that will implement the given APIs to connect it to the platform.	

PF18	Service composition	The platform SHALL be able to compose security services by combining one or more of the available vNSFs.
A_SPEC_06	The platform will communicate with several vNSF collectors to ingest network traffic required for performing security analytics	
NF01	Response time	The platform SHALL report the incident within a relatively short time (in the order of seconds).
A_SPEC_07	The platform will incorporate state-of-the-art streaming processing technologies to process the ingested data.	
NF03	Scalability	The storage and processing capabilities of the platform SHALL be able to increase merely by adding resources to the system.
A_SPEC_08	The data analytics modules are based on state-of-the-art distributed storage and computing technologies (e.g. HDFS, Spark, Hive, etc.) which are scalable by design and associated with big data processing. The security DA module leverages a distributed architecture where the different components used for storage and processing can dynamically and independently allocate new resources and can scale to accommodate large volumes of data.	
NF05	Impact on perceived performance	When network traffic is proxied or analysed, the user experience SHALL not be degraded.
A_SPEC_09	The platform will be based on a distributed computing framework that will feature scalable storage and processing, load-balancing and resource management functionalities.	
NF06	Performance factors	The platform SHALL offer an availability-related performance similar to carrier grade system. It includes recovery time and redundancy capability.
A_SPEC_10	The technologies used to implement the data analysis modules phase are by design resilient against failures and ensure that all processing activities remain consistent even in the presence of network and node failures.	
NF07	Compliance to standards	The platform SHALL conform to well-established standards, in particular with respect to data export (e.g. STIX) and input (e.g. NetFlow).
A_SPEC_11	The two data analysis modules of the DARE will conform to well-established data input formats, in order to provide information to the Cybersecurity topologies phase.	

ERC01	Access to and portability of personal data	All components that process and/or store personal, identifiable information SHALL provide data subjects with a way to access and review their personal data. If the data processing does not require identification, the component is not required to provide access, unless the user can provide additional information enabling their identification (according to Article 11 of the GDPR).
A_SPEC_12	In terms of analytics, since the DARE offers multi-user support, each user will gain access only to threat results that are relevant to his organisation. This could be achieved by correlating the IP addresses of each SHIELD-monitored organisation to the overall analysis results, to provide tailored information, in compliance with the GDPR.	
ERC02	Data rectification and erasure	All components that process and/or store personal, identifiable information SHALL provide data subjects with a way to request that their data be rectified or erased. If the data processing does not require identification, the component is not required to provide this functionality, unless the user can provide additional information enabling their identification (according to Article 11 of the GDPR).
A_SPEC_13	All network traffic collected by the DARE could be erased upon request (platform website) and/or after a set period of time. The threat detection functionalities of the data analytics modules (e.g. training of the ML-based models, threat detection, results visualisation etc.) are not affected by the deletion of past data.	
ERC04	Transparency in data processing	The platform SHALL present visibly and transparently the technical information pertaining to the components' data processing. Data processing activities should be logged.
A_SPEC_14	Information regarding the data processing procedures followed by the DARE's two analytics modules will be logged and available to the users upon request.	
ERC05	Data retention	The components storing and processing personal identifiable data SHALL define a specific data retention period.
A_SPEC_15	The data analytics modules will retain data for a user-defined time period.	
ERC06	Transparency in traffic classification	Components with the ability to classify traffic and apply throttling/limiting measures SHALL provide detailed information.
A_SPEC_16	The Cognitive DA module is based on open-source technologies. Thus, all traffic classification information is available to the public. The Security DA	

	module is a commercial product and as such some specific details related to the exact algorithmic implementations may not be disclosed.

Cognitive Data Analytics module

The Cognitive Data Analytics module infrastructure consists of a cluster of nodes running on open-source OS and virtual machines. Each node will perform several operations and will be orchestrated by cluster and resource management technologies, specifically designed for big data applications. The specifications of the Cognitive Data Analytics module include:

- A cache system for real time analytics.
- Unified services for orchestration, operations and resource management.
- Machine Learning modules that provide scalable ML algorithms for network traffic filtering, as part of a cluster-computing framework.
- An Operational Analytics module that allows for the implementation of whitelisting and filtering techniques that will help reduce false positives and offer remediation recommendations to the users.

The module will consist of several nodes (physical or virtual machines), each of them performing a designated task. The worker nodes receive the ingested network traffic from the Data Acquisition Phase and are responsible for the operation of the machine learning entity. The Operational Analytics nodes execute the filtering and whitelisting functionalities and are intended to operate as the final editing step, before pushing the detection results to the Remediation Engine.

Implementation details

The cognitive Data Analytics module performs anomaly detection analytics, following the Spot architecture, by deploying a collection of state-of-the-art technologies (Hadoop, Spark, Kafka, etc.) in the form of an integrated ecosystem, the Cloudera Distribution for Hadoop. CDH is an Apache-licensed open-source framework that delivers the core elements of scalable storage and distributed computing, along with a Web-based UI and enterprise capabilities and is considered the most popular distribution for Apache Hadoop and related projects. Spot utilises CDH as a general dependency package for the development of its three main parts: ingestion (Subsection 3.1), machine-learning and operational analytics.

- Machine Learning: The machine learning entity is responsible for the detection of anomalies in network traffic and the prevention or mitigation of potential threats. Apache Spot already contains routines for performing anomaly detection on NetFlow, DNS and proxy logs and there is the intention to develop additional algorithms that will handle security event data and metrics gathered from the vNSFs. These routines consume a collection of network events and produce a list of the events that are considered to be the least probable, these being considered the most suspicious. The statistical model that is currently used for discovering abstract topics of these events and ultimately discovering normal and abnormal behaviour is a topic modelling algorithm called Latent Dirichlet Allocation [27]. LDA is a generative probabilistic model used for discrete data that is applied to network traffic by converting network log entries into words through aggregation and discretisation, to discover hidden semantic structures. Spot executes LDA routines using a Scala Spark implementation from MLlib, Apache Spark's scalable machine learning library. It should be noted that Spot's current capabilities do not include any anomaly classification algorithms that would interpret the detected outliers as specific threats/attacks, thus such an algorithm will be originally developed to meet this requirement. The module will exploit Spot's existing batch processing capabilities, coupled with the development of streaming analytics functionalities currently missing from Spot, to achieve real-time (or near real-time) visibility for threat detection.

- Operational Analytics: The operational analytics entity will exploit Spot's built-in results editing capabilities, as well any additional features that will result from the developed machine-learning features. Spot uses the Jupyter/IPython [16] notebook, a server-client application that allows editing and running notebook documents via a web browser, to apply filtering and whitelisting services thus providing a more accurate view of the overall anomaly detection procedure, by reducing false-positives.

Comparison with similar technologies

The Cognitive Data Analytics module was initially planned to be developed by leveraging machine learning techniques (e.g. Naive Bayes classification [28], Support Vector Machine [29]) to analyse events and network data. This would include the selection, configuration and deployment of several state-of-the-art frameworks for Big Data analysis (e.g. Apache Mahout [30], Scala [31] frameworks). The Apache Spot framework can be considered a superset of the above frameworks, as it features a combination of distributed computing (Hadoop), data lake management (Hive) and machine learning (Spark) frameworks along with auxiliary services to ingest, analyse and present the detected anomalies in network traffic. It being maintained by a large community is a strong indication that Spot will continue to evolve its cybersecurity capabilities, while the open-source nature of the project will allow for contributions regarding all its main components.

A similar framework that could be deployed for network analytics within the scope of SHIELD is Apache Metron/OpenSOC [32]. Metron is a cybersecurity application framework that provides the ability to ingest, process and store diverse security data feeds at scale to detect cyber anomalies. Metron and Spot have similar objectives and approaches from an ingestion, storage and user interface perspective. The key differences relate to the Open Data Model that is utilised only by Spot and the fact that Spot leans towards a machine-learning approach to provide results, while Metron is currently more focused on traditional deterministic (rules/signatures/patterns) analytics.

Other open source/commercial products not very close to the network traffic analysis that try to simplify the cognitive analysis closing the algorithms available or lock to their own sources (Splunk, Machine Learnings, ToolKit app) or on the contrary just offer wide open Machine Learning tools (Google TensorFlow, Amazon ML) where the client must setup the dataset injection model and create their algorithm from scratch. Apart from the fact that created trained algorithms code is not exportable, these products are not specialised in network security, such as Apache Spot.

Finally, Spot uses Spark instead of Hadoop. Spark is an extension of Hadoop in the sense that for batch processing it uses Hadoop, but it adds the possibility to work with cache and streaming. This will be mandatory for processing real-time traffic.

Security Analysis Module

The Security Analysis module is derived from the security module of Talaia's commercial product and will be integrated into the SHIELD platform as a stand-alone component. That module will be extended and integrated within the SHIELD framework. To this end, and similarly to the Cognitive Data Analysis module, the specification of the Security Analysis module includes functionalities for real-time detection and classifications of network performance anomalies with a low false positive ratio.

The different entities inside this module can run in a distributed infrastructure or in a stand-alone node (physical or virtual machine) depending on the scenario requirements. The operation will start with the reception of the data by the Data Ingestion framework that will transform and adapt the data, so it can be understood by the Security Analysis module. Then, the data is stored in a distributed storage system to assure no data is lost while the Security Analysis module is processing them.

Implementation details

From the implementation point of view, the Security Analysis module is a stand-alone black box integrated into the SHIELD framework. As previously mentioned, the Security Analysis module is derived from Talaia's commercial product. Because of this, some specific details related to the exact implementation cannot be published.

The original Security Analysis module only uses NetFlow-derived data to perform the detection and classification of network performance anomalies (e.g., DDoS, network scans). In this project, this module will be extended to use the data provided by the Apache Spot framework (e.g., DNS, proxy logs) and by the monitoring vNSFs developed within the SHIELD project. All the data available will then be transformed so the different entities in the Security Analysis module can take advantage of the new information. As a result, the data mining and machine learning techniques used in the four entities of the Security Analysis module will enrich and improve the detection and classification of network anomalies. In addition, the new information provided will also allow the detection and classification of new anomalies not detected in the original version.

Once new anomalies are detected and classified, the Security Analysis module will adapt its output (e.g., anomaly detected, traffic involved in the anomaly), so the Dashboard can represent it and the Remediation Engine can actuate on it.

Comparison with similar technologies

The technologies used in the Security Analysis module consist of a set of cutting-edge data mining and machine learning techniques from the literature combined with proprietary methods.

On the one hand, as part of a commercial product, similar solutions can be found in Talaia's competitors. However, similarly to Talaia, competitors are always very reticent to share the details of part of its core business. Among the different competitors, we can highlight Kentik [33] and DeepField/Nokia [34] because both solutions are software-based solutions commercialised from the cloud and that are currently being adapted to the NFV paradigm. In addition, similarly to the Security Analysis module, these solutions base their classification and detection of anomalies in data mining and machine learning techniques enriched with proprietary methods. There are also hardware-based solutions as the one provided by Arbor Networks [35] that can detect network anomalies by using deep packet inspection techniques.

On the other hand, the open-source Apache Spot framework used in this project and the Metron framework are also similar solutions.

3.3. Cybersecurity topologies phase

The policy engine leverages the policy specification and optimisation techniques developed in the scope of the EC-funded project SECURED [36]. The SECURED project aimed at the definition of an abstract configuration for each security capability, which could be consistently transformed into specific settings by the actual implementation of the security control.

Specifications

The requirements elicited in D2.2 are here transformed into specifications from the point of view of the Cybersecurity topologies phase (Table 4).

Table 4: Specifications of the Cybersecurity topologies phase and fulfillment of requirements.

Req. number	Requirement Name	Requirement description
PF02	vNSF lifecycle management	The platform SHALL be able to manage the full lifecycle of vNSFs (on boarding, instantiation, chaining, configuration, monitoring and termination).
T_SPEC_01	The Recommendation and Remediation subcomponent will oversee defining configurations, using a high-level, application-independent syntax, for each vNSF. To do so, it will provide pre-defined recipes to address each supported threat. These recipes will define the protection requirements to be implemented by the vNSFs to address the network security threat. The recipes will be stored in a database and an API will be provided to interact with it. Starting from the protection requirements, a set of security capabilities will be derived. Each vNSF will have to support one or more capabilities, allowing this subcomponent to select the optimal set of vNSFs (the Forwarding Graph). The Remediation and Recommendation subcomponent will then translate each capability into a high-level policy for configuring the vNSF and will provide the suggested mitigation to the final user via Dashboard, by interacting with the Dashboard API subcomponent.	
PF06	Ability to offer different management roles to several users	The platform SHALL provide domain management with accessibility to the resources of a domain by different users. The admin of a domain has to be able to create management users with different roles.

T_SPEC_02	The Recommendation and Remediation subcomponent will provide mitigation actions to be applied in the user’s domain. No information will be gathered regarding other SecaaS clients during this operation.	
PF08	Platform expandability	The platform SHALL be easily extended to support new security services.
T_SPEC_03	The Recommendation and Remediation subcomponent will be able to interact with the vNSF Store to select the best set of vNSFs to address a security threat with a generic pull interface. In this way, the subcomponent will be able to choose among all the different and evolving implementations for the NSs.	
PF12	Threat data sharing	The platform SHALL allow to share threat data with a third entity. The granularity of such data depends on the severity and type of each attack.
T_SPEC_04	The Cybersecurity Topologies phase of the DARE will for forward recommendations to the Dashboard along contextual information about the detected threat. This information could be used by a third entity, such as a cybersecurity agency, to identify the attacks in the ISP network.	
PF13	Mitigation	The platform SHALL be able to trigger, in the case of an event, proper actions in order to mitigate the threat.
T_SPEC_05	The Cybersecurity Topologies phase of the DARE will implement the complete workflow to derive a mitigation action, consisting in a topology of vNSFs and their configuration (expressed in an application-independent syntax) starting from an occurring threat.	
PF21	Operation Traceability	The platform SHALL provide profile-related event generation for each of the user actions. E.g.: platform administrator, domain administrator, management user, etc.
T_SPEC_06	The Cybersecurity Topologies phase will not require user driven actions for the mitigation of incoming threats, as the recipe will be automatically selected according to the attack and severity. User actions may occur only at the generation/update of recipes hence they will be tracked via application logs.	
PF22	Management communication security	The platform SHALL encrypt all the management communications.

T_SPEC_07	The Cybersecurity Topologies phase of the DARE will result in a recommendation message that will be forwarded to the Dashboard in a secure way, i.e. by means of encryption of the TLS protocol.	
NF01	Response time	The platform SHALL report incidents within a relatively short time (in the order of seconds).
T_SPEC_08	The Cybersecurity Topologies phase of the DARE will provide mechanisms to aggregate mitigation rules, if possible, to reduce the response time in reporting incidents and mitigating them.	
NF02	Availability	The storage and processing capabilities of the platform SHALL be able to increase merely by adding resources to the system.
T_SPEC_09	The Cybersecurity Topologies processing components, as well as the recipes database, can be deployed on several nodes in a load-balanced scenario to increase their availability.	
NF03	Scalability	The storage and processing capabilities of the platform SHALL be able to increase merely by adding resources to the system.
T_SPEC_10	The Cybersecurity Topologies processing components, as well as the recipes database, can be deployed on several nodes in a load-balanced scenario to increase their scalability.	
NF06	Performance Factors	The platform SHALL offer an availability-related performance similar to carrier grade system. It includes recovery time and redundancy capability.
T_SPEC_11	The Cybersecurity Topologies processing components, as well as the recipes database, can be deployed on several nodes in a redundant scenario, as it performs as a stateless component of the platform.	
NF07	Compliance to standards	The platform SHALL conform to well-established standards, in particular with respect to data export (e.g. STIX) and input (e.g. NetFlow).
T_SPEC_12	The Cybersecurity Topologies phase of the DARE will process data provided by the analytics engine, which conform to well-established data input formats, and forward mitigations in lightweight messages to the Dashboard.	

NF08	Deployment and support simplicity	The platform SHALL be easily installed and maintained, without the need of specific expertise.
T_SPEC_13	The Cybersecurity Topologies components are implemented as a standalone application that can be deployed separately from the other DARE sub-systems.	
SF01	Content filtering	A security service COULD provide URL filtering based on different configurable categories (e.g. political, violence, sex, social networks, etc.) for internet web browsing.
T_SPEC_14	The Cybersecurity Topologies phase will provide a recipe to configure vNSFs that address this security issue.	
SF02	Detect/Block access to malicious network locations	A security service SHALL control access to malicious network locations, such as phishing servers, malware spreading websites, Command & Control (C&C) servers, etc. The user must be alerted and the access to the site could be blocked/allowed depending on the configured policy rule.
T_SPEC_15	The Cybersecurity Topologies phase will provide a recipe to configure vNSFs that address this security issue.	
SF03	Security assessments	A security service COULD provide continuous vulnerability assessment on the network, hosts or applications.
T_SPEC_16	The Cybersecurity Topologies phase could provide a recipe to configure vNSFs that address this security issue.	
SF05	Central log processing/SIEM	A security service COULD collect and correlate security logs from different legacy user sources and generate alerts. This service is intended to provide the user with a way to process its security logs that are not generated by a vNSF in SHIELD.
T_SPEC_17	The Cybersecurity Topologies phase could provide a recipe to configure vNSFs that address this security issue.	
SF08	Denial of Service (DoS) Protection	A security service SHALL protect against volumetric Denial of Service attacks.

T_SPEC_18	The Cybersecurity Topologies phase will provide a recipe to configure vNSFs that address this security issue.	
SF09	Intrusion Detection/Prevention System	A security service SHALL detect attacks with a wide range of techniques such as network flow or behaviour analysis and deep packet inspection.
T_SPEC_19	The Cybersecurity Topologies phase will provide a recipe to configure vNSFs that address this security issue.	
SF10	Honeypots	A security service COULD provide a Honeypot service that simulates or impersonates specific services (e.g., Windows computer, Web server, IoT or SCADA device, etc.) in order to detect malicious behaviours in the network.
T_SPEC_20	The Cybersecurity Topologies phase could provide a recipe to configure vNSFs that address this security issue.	
SF11	Sandboxing	A security service COULD provide a sandbox service for executing and analysing programs
T_SPEC_21	The Cybersecurity Topologies phase could provide a recipe to configure vNSFs that address this security issue.	
SF12	Virtual Private Network (VPN)	A security service COULD provide a secure tunnel service in order to connect the branch of a client with users on the Internet or other branches.
T_SPEC_22	The Cybersecurity Topologies phase could provide a recipe to configure vNSFs that provide a secure tunnel service.	
ERC09	Lawful Interception	The vNSFs SHALL support LI capacities, or integrate a LI system, if the vNSF changes the public IP address (for Internet connection) or encrypts the internet traffic. LI capacities are defined by ETSI. Law enforcement agencies may require access to a number of transmitted telecommunications regarding a particular subject, target, date, etc. If network operators/service providers initiate encoding, compression or encryption of telecommunications traffic, law enforcement agencies require the network operators/service

		providers to provide intercepted communications in clear.
T_SPEC_23	The Cybersecurity Topologies phase of the DARE will process anomaly detection results but will not change public IP addresses neither encrypt any traffic.	

Implementation details

The policy engine to be developed in the Cybersecurity Topologies phase will leverage the functionalities provided by several modules developed within the SECURED project, but it will aim to extend their capabilities and adapt the workflow to what is expected in the DARE.

The different layers of policy abstraction, as expressed in the specifications for this subcomponent, will be mapped to different languages, namely:

- High-level Security Policy Language (HSPL), suitable for expressing the general protection requirements without specifying configuration rules;
- Medium-level Security Policy Language (MSPL), suitable for expressing configuration rules in an application-independent syntax.

The policy abstraction layers will be relevant for the SHIELD recommendation and remediation engine, as they will be used to describe the mitigation action starting from the general protection requirements to the configuration rules to be applied by each vNSF. The first will be utilised by the High-level Policy Recipe Provider to describe one or more rules that compose the recipe. The latter will be used to describe configuration rules for each vNSF, whose functionalities will be mapped on the security capabilities. It is to be noted that each vNSF in the catalogue will have to provide this information embedded in its metadata, to allow the recommendation and remediation engine to choose the best set of vNSFs for a specific threat.

The SECURED project also designed and developed different transformation tools to ease the configuration of security functions, namely:

- H2M Service for HSPL to MSPL transformation;
- M2L Service for MSPL to low-level configuration transformation.

Hence the transformation from MSPL to low-level configuration will not be part of the recommendation and remediation engine, no integration of the M2L Service is needed in this component.
The recommendation and remediation engine will leverage the functionalities offered by the H2M Service to provide an optimal set of vNSFs with proper MSPL configuration starting from the set of HSPL rules in input.

The two modules that will be implemented from scratch, namely the High-level Policy Recipe Provider and the Dashboard API subcomponent, will follow the best practices in development, testing and documentation and will leverage the functionalities offered by several frameworks, such as Swagger [37] for the API provisioning and different technologies for No-SQL database (e.g. MongoDB [38], Redis [39], Apache Cassandra [40]), required to store the HSPL-based recipes in a key-value structure (where the key would be the threat “signature”). A REST API with Create, Read, Update, and Delete CRUD capabilities will be provided to let a system administrator interact with the recipes' database.

Dashboard API operations

The following table (Table 5) describes the envisioned operations of the Dashboard API.

Table 5: Operations of the Dashboard API.

Operation	Arguments	Description
post_mitigation	threat context information, network service, recommendation	Send the list of MSPL policies to be applied, along with contextual information of the attack and the identifier of the Network Service to be deployed to address the incoming threat.

Comparison with similar technologies

The configuration of vNSFs in an orchestrated, distributed environment has received great attention in literature, but there are not currently open-source, widespread technologies that provide this capability because of the lack of formal representations for the type of data needed. The policy specification engine, as originally designed and developed in SECURED, has been considered as a fitting solution for the DARE's recommendation and remediation subcomponent, as it provides both the two-level abstraction for policies and the functionalities to identify, optimise and configure vNSFs running in a SDN/NFV environment. In addition, the MSPL language is defined as a meta-model, which may be extended to support different security functions, other than the ones defined in SECURED. Other solutions, which will be briefly introduced in this section, are either too specific for a security function (e.g. packet filter) or don't provide a convenient way of describing configurations in an application-independent syntax. Proprietary solutions, if any available, are not described in this section as they would provide translation mechanisms for vendor-specific network security functions. Firmato [41] is a proposal for translating high-level security requirements into packet filter configurations, based on an entity-relationship model to represent the knowledge base (e.g. the protection requirements and the topology of the network. Its applicability to large and heterogeneous networks, which may be typical for NFV environments, has not been proven, as the technology has been applied only to a network with a single border firewall. FACE [42] is another model for configuring a firewall in an NFV environment. It takes as input the topology and a global high-level policy, and outputs the packet filter rules. MIRAGE [43] is a tool for analysis and deployment of security policies, which is not limited to supporting packet filter's rules, but also configurations for VPN gateways and IDSs. The SECURED's solution is considered to be more generic, as it adopts an extensible capability-based language, and it is also supported by an open-source implementation that has been already tested in a NFV scenario, close to the one envisioned in SHIELD.

3.4. Dashboard

Based on the requirements elicitation available in D2.2 as well as on the specification of the Dashboard component previously provided in the current document, this section will explain the requirements currently addressed by the Dashboard component as well as the implementation details currently envisioned for this component (Table 6).

Specifications

The requirements elicited in D2.2 are here transformed into specifications from the point of view of the Dashboard (Table 6).

Table 6: Specifications of the Dashboard component and fulfillment of requirements.

Req. number	Requirement Name	Requirement description
PF03	vNSF status management	The operator SHALL be able to control the lifecycle via a graphical user interface. The vNSF lifecycle should support events like DEPLOY, START, STOP, MODIFY, DELETE.
D_SPEC_01	Dashboard has connectivity with Orchestrator component (Orchestrator connector subcomponent) allowing lifecycle features to be provided to the end user through the Graphical User Interface.	
PF05	Analytics visualisation	The operator SHALL be able to see the analytics visualised in e.g. a dashboard.
D_SPEC_02	DARE connector subcomponent will allow GUI to expose monitoring information persisted in this component. Furthermore, Remediation Queue subcomponent will stage recommendation sent by the DARE waiting for acceptance/rejection from the end user.	
PF06	Ability to offer different management roles to several users	The platform SHALL provide domain management with accessibility to the resources of a domain by different users. The admin of a domain has to be able to create management users with different roles.
D_SPEC_03	User management and AA subcomponents will provide Authentication, Authorisation and user management features allowing creating and using different roles with different sets of permissions. SHIELD's features and information will be filtered based on the role and permissions of each authenticated user.	

PF07	Service elasticity	The platform COULD provide the mechanism to allow scalability of the vNSFs.
D_SPEC_04	Dashboard will allow triggering this mechanism through its interface, using the Orchestrator connector to ensure this feature.	
PF09	Access control	The platform SHALL provide a secure environment. Authentication mechanisms that should control the access and restrict access only to authenticated users.
D_SPEC_05	User management and AA subcomponents will provide authentication, authorisation and user management features. Only authenticated users will have access to SHIELD's internal features and information.	
PF12	Log Sharing	Sharing logs with a third entity SHALL be allowed. The granularity of the data provided by the logs depends on the severity and type of each attack.
D_SPEC_06	The Dashboard will provide access to APIs that will allow the sharing of logs between the DARE and other entities.	
PF13	Mitigation	The platform SHALL be able to trigger, in the case of an event, proper actions in order to mitigate the threat.
D_SPEC_07	The Dashboard will allow the deployment of vNSFs and policies via the GUI.	
PF14	Multi-tenancy	The platform SHALL accommodate multiple users, with isolated services and secured access to analytics
D_SPEC_08	User management and AA subcomponents will provide authentication, authorisation and user management in a multi-user environment.	
PF15	Service store	The store SHALL allow selecting security services from the catalogue.
D_SPEC_09	Store connector subcomponent will allow GUI to expose Store's security service catalogue features.	
PF16	Historic reports	The platform SHALL generate reports of past incidents based on historic data.
D_SPEC_10	Historic reports will be available in the dashboard using DARE connector to fetch this information.	

PF17	Interoperability	The platform SHALL expose openly-defined APIs for information exchange with third parties.
D_SPEC_11	Dashboard will provide an open API allowing third party applications to use it.	
PF20	Billing framework	The platform SHALL implement a billing framework for the use of the security services. The clients should be able to access to the functionalities defined by their payment modality.
D_SPEC_12	Billing features will be assured by Dashboard's Billing subcomponent enabling SHIELD's monetisation features.	
PF21	Operation Traceability	The platform SHALL provide profile-related event generation for each of the user actions. E.g.: platform administrator, domain administrator, management user, etc.
D_SPEC_13	All User actions logged and the records available to the appropriate profiles. The log shall store at the very least the username, the profile, the action and a timestamp for the occurrence.	
PF22	Management communication security	The platform SHALL encrypt all the management communications.
D_SPEC_14	The communication with the Dashboard is done using HTTPS, where the messages are encrypted using the Transport Layer Security protocol.	
NF01	Response time	The platform SHALL report incidents within a relatively short time (in the order of seconds).
D_SPEC_15	The Dashboard will provide a message queue for incidents notification. As soon as an incident is placed in the queue the user is notified through a visual alert.	
NF02	Availability	The core platform SHALL be able to recover in case of hardware failures.
D_SPEC_16	The Dashboard will employ technologies resilient to failure and support failover when in the event of a node failure.	
NF03	Scalability	The storage and processing capabilities of the platform SHALL be able to increase merely by adding resources to the system.

D_SPEC_17	The Dashboard will be embrace the scalability by design principle, be it scale in/out, concurrency, rate limiting or data storage. These requirements must be present in the technology stack selected for the implementation.	
NF07	Compliance to standards	The platform SHALL conform to well-established standards, in particular with respect to data export (e.g. STIX) and input (e.g. NetFlow).
D_SPEC_18	The Dashboard will provide a mechanism for data exchange. It will only as a vessel whereby the data contents or format are agnostic to the Dashboard.	
NF08	Deployment and support simplicity	The platform SHALL be easily installed and maintained, without the need of specific expertise.
D_SPEC_19	The Dashboard will provide installation scripts to ease and automate all the steps required for the deployment process.	
NF09	vNSF hardening	The vNSFs SHALL be hardened.
D_SPEC_20	The Dashboard will provide an option for an authorised User to mark a vNSF as ready for deployment. It is expected the User only performs this action once the sandboxed vNSFs goes through some verification procedure.	
SF05	Central log processing/SIEM	A security service COULD collect and correlate security logs from different legacy user sources and generate alerts. This service is intended to provide the user with a way to process its security logs that are not generated by a vNSF in SHIELD.
D_SPEC_21	The Dashboard will provide a way for the authorised user to deploy the appropriate NS to handle this scenario.	
SF11	Sandboxing	A security service COULD provide a sandbox service for executing and analysing programs.
D_SPEC_22	The Dashboard will provide a way for the authorised user to deploy the appropriate NS to handle this scenario.	
SF12	Virtual Private Network (VPN)	A security service COULD provide a secure tunnel service in order to connect the branch of a client with users on the Internet or other branches.
D_SPEC_23	The Dashboard will provide a way for the authorised user to deploy the appropriate NS to handle this scenario.	

ERC01	Access to and portability of personal data	All components that process and/or store personal, identifiable information SHALL provide data subjects with a way to access and review their personal data. If the data processing does not require identification, the component is not required to provide access, unless the user can provide additional information enabling their identification (according to Article 11 of the GDPR).
D_SPEC_24	The Dashboard will only require minimal data from any user. For the platform operation users, it will collect no personal data aside from an e-mail for communication purposes. As for billing the SecaaS clients are only identified by an ID, any personal data will live in an external invoicing system.	
ERC03	Access to related Data Protection information	The platform SHALL provide the data subject with easy access to the following information: - The identity and contact details of the data controller(s) - The identity and contact details of the Data Protection Officer - The purpose of processing and categories of data concerned - The recipients of the collected data - A statement on transfer of data to third parties (including cross-border) - An interface that allows the user to lodge a complaint to the Data Protection Officer
D_SPEC_25	The Dashboard will provide a user interface to enter Data Protection information, to display said information and to allow lodging a complaint.	
ERC04	Transparency in data processing	The platform SHALL present visibly and transparently the technical information pertaining to the components' data processing. Data processing activities should be logged.
D_SPEC_26	When onboarding a vNSF the developer will provide information on the data protection regulatory compliance. When browsing for vNSFs to use in a NS, the user will be presented with such information in simple and clear way, so an informed selection can be made.	

ERC05	Data retention	The components storing and processing personal identifiable data SHALL define a specific data retention period.
D_SPEC_27	When browsing for vNSFs to use in a NS, the Dashboard will present to the user the data retention period defined by the developer.	
ERC06	Transparency in traffic classification	Components with the ability to classify traffic and apply throttling/limiting measures SHALL provide detailed information.
D_SPEC_28	When browsing for vNSFs to use in a NS, the Dashboard will present to the user information on the data protection regulatory compliance. It will also present a log of any action to throttle or block traffic (provided such information is conveyed to the Dashboard).	
ERC07	Notification obligation	In the case of a breach in a component that processes personal data, the platform SHALL produce a breach notification. Data rectification or erasure should be accompanied with a notification to the data subject unless it is difficult or involves disproportionate effort, as per article 19 of the GDPR.
D_SPEC_29	The Dashboard will convey the breach notification to the contact associated with the tenant. Such notification is provided by the component which detects the breach.	
ERC08	Net Neutrality	The platform SHALL not recommend actions that lead to user traffic penalization, unless explicitly required for threat mitigation. The net neutrality rules adopted by the European Parliament on 30 April 2016 aimed to strengthen net neutrality by requiring internet service providers (ISPs) to treat all traffic equally, without favouring some services over others. For this reason, no service could be used by an ISP to punish or to favour the traffic of a user respect the rest of the users.
D_SPEC_30	The Dashboard will convey to the user a visual warning associated with any threat mitigation which may lead to traffic penalization.	
ERC09	Lawful Interception	The vNSFs SHALL support LI capacities, or integrate a LI system, if the vNSF changes the public IP address (for

		Internet connection) or encrypts the internet traffic. LI capacities are defined by ETSI.
D_SPEC_31	The Dashboard will include a role for users associated with LI. Such role will allow the user to add LI-capable NSs to an existing chain.	

Implementation details

The Dashboard contains a set of subcomponents that together allow providing all the features envisioned for the component. Regarding its Graphical User Interface subcomponent, it will be based on web technologies allowing end users to access it using a common web browser. Hence, standard technologies will be used such as Hyper Text Markup Language (HTML), JavaScript and Cascading Style Sheets (CSS). The use of web frameworks based on these technologies such as AngularJS [44] is also envisioned allowing to boost development efficiency, productivity, data handling and maintenance. Regarding visual styling, technologies such as Sass will be incorporated as an extension to CSS enabling new manipulation methods as well as an efficient code structuring. The GUI subcomponent will leverage results achieved in previous and running EC-funded projects namely T-Nova [45]and SELFNET [46]. The features provided by the Graphical User Interface of each one of these projects will be merged and extended taking advantage of the result of all discussions and conclusions that took place in each project. More precisely, the web interfaces associated with the vNSF catalogue, store, permission management and vNSF orchestration will be based on the conjunct result provided by these two projects. Regarding network topology visualisation, SHIELD's GUI will leverage the work done in both SELFNET and SONATA [47] in an attempt of providing an appealing and intuitive interface for this feature. While SELFNET focused on the physical and virtual connectivity of the virtual machines instantiated across the Network Infrastructure, SONATA implemented a hierarchical visualisation on the resources associated with each Network Service. For the implementation of this visualisation, the use of Scalable Vector Graphics (SVG) [48] is envisioned with the support of D3js framework [49] providing a set of relevant tools allowing its manipulation. The use of both D3js and SVG is enforced by the fact that both SELFNET and SONATA use this combination to implement its topology visualisation features.

The interaction between GUI and API Service is envisioned to be implemented using REST technology. Consequently, a REST client will be developed in the GUI subcomponent allowing the connection to a REST Service to be developed and provided by API Service. Regarding the real time push of information coming from API Service to the GUI, websockets technology is envisioned allowing the seamless implementation of the envisioned workflow.

As previously mentioned, API Service will be based on REST technology providing a set of endpoints allowing the use of features provided by SHIELD's internal components. These endpoints will interact with a specific Dashboard's internal subcomponent (User Management, Remediation Queue, Billing, Store Connector, Orchestrator Connector, and DARE Connector) responsible for sanitizing data as well as for interacting with the correct connector. Regarding the connector subcomponents, their goal is to abstract the interaction with a given SHIELD component so the use of technologies like real time messaging, advanced message queuing, REST or SOAP technologies will depend on the interfaces and workflows provided by it.

AA - Role-Based Access Control (RBAC)

The authorization and authentication for the Dashboard is based on the definition of users and roles to associate with a user. The user is authenticated against a set of credentials and authorised to use resources provided he/she holds the proper role. This is the basis for Role-based Access Control and the available roles for SHIELD are described next. A diagram framing the roles within SecaaS client is presented in Figure 24.

<u>Roles:</u>

Developer. He/she can submit vNSFs for onboarding into the platform. It is also the role with the least privileges granted in the entire platform.

Lawful Interception. He/she can list all the vNSFs and NSs, as well as the recommendations history, and be notified of any events targeted for vNSFs instances, for all the SecaaS clients in the platform.

Platform Administrator. He/she performs all the Operations & Maintenance (O&M) tasks available for a SHIELD instance, having the highest level of privileges granted to a user.

Service Management. He/she performs all the O&M tasks associated with Network Services and vNSFs from creation, to deployment, to configuration, and to decommissioning. This role has the absolute control over which NSs are available for a SecaaS client and thus deployable by a less privileged user associated with the SecaaS client.

SecaaS Administration. He/she performs all the O&M tasks associated with a specific SecaaS client, having the highest level of privileges granted to a user tasked with managing a SecaaS client.

SecaaS Auditing. He/she can explore data related to auditing purposes such as statistics, access and high-level operational logs, billing information and the likes, to determine the resources used and the overall service provided to a SecaaS client.

SecaaS Maintenance. He/she performs all the maintenance tasks associated with a SecaaS client, be it backups, periodic cleanup tasks, scheduling of any kind, etc. It is bared from performing any tasks concerning NSs or vNSFs.

SecaaS Monitoring. He/she gets notified of any events targeted for vNSFs instances associated with a SecaaS client. This role can list recommendations and inspect its contents and status, but it is bared from performing any action on said recommendations.

Cyber Agency. He/she has the same role as SecaaS Monitoring with the significant difference that he/she can access information for all the SecaaS clients in the platform.

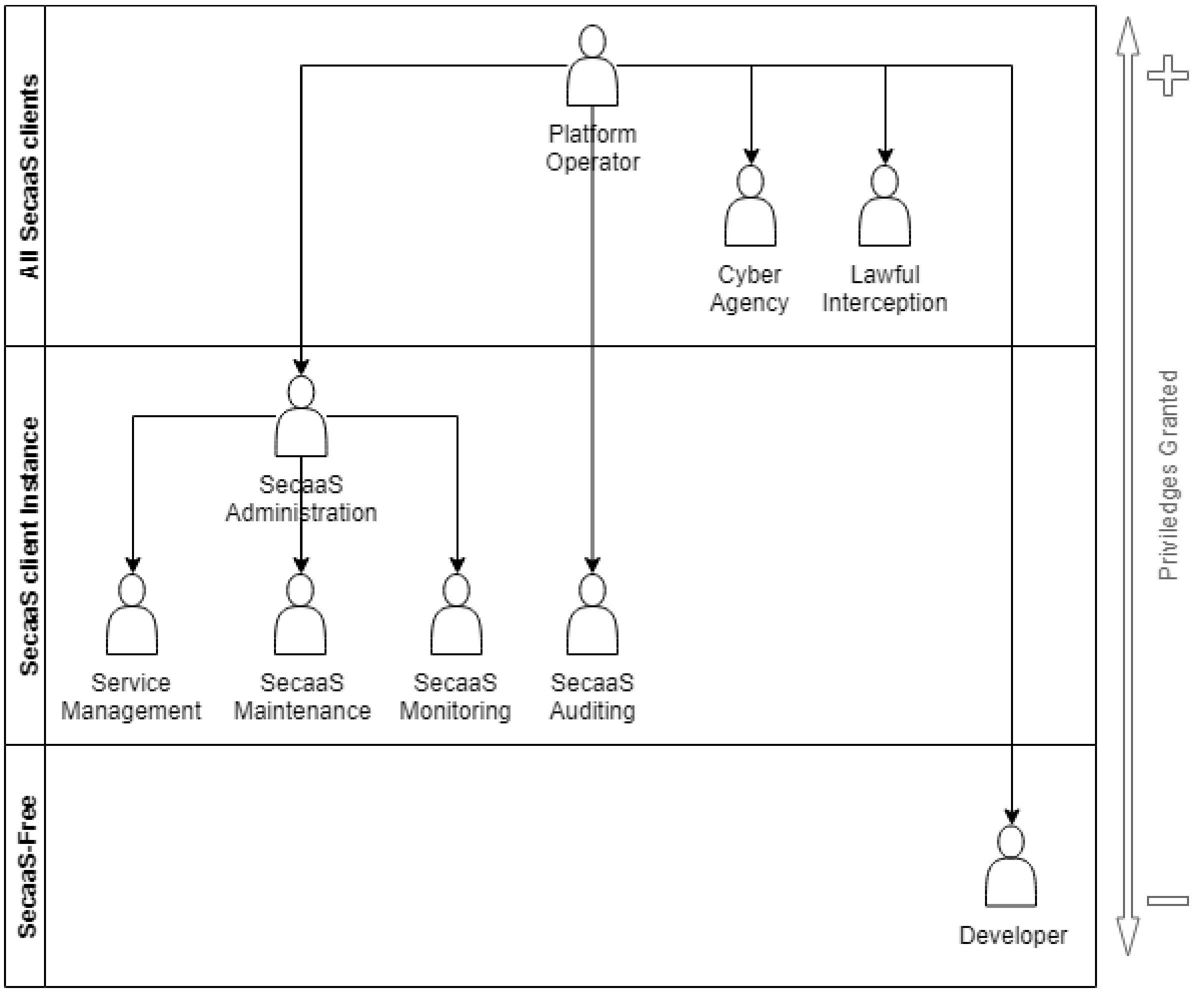

Figure 24: Role-based access control

<u>Operations</u>

The operations the several roles can perform on the Dashboard are described next. A mapping showing which roles are allowed for each operation is presented in Table 7.

Apply recommendations. Apply a security recommendation for a SecaaS client.

CRUD on NSs. Create, list, update and delete NSs.

CRUD on Users/Roles. Can create, list, update and delete Users and assign Roles to Users.

CRUD on SecaaS clients. Create, list, update and delete SecaaS clients in the platform.

Deploy NSs on a SecaaS client. Deploy a NS for a SecaaS client it can manage.

Events notification. Receive notifications for events produced by vNSFs assigned to a SecaaS client.

List NSs for a SecaaS client. Enumerate all NSs deployed (either past or present) for a SecaaS client.

List recommendations. Enumerate all recommendations applied to a SecaaS client.

List vNSFs for a SecaaS client. Enumerate all vNSFs deployed (either past or present) for a SecaaS client.

Manage NS status. Start, stop, modify or delete a NS for a SecaaS client.

Manage vNSFs status. Start, stop, modify or delete a vNSF for a SecaaS client.

Onboard NSs. Onboard NSs, so they are available for all SecaaS client to use.

Onboard vNSFs. Onboard vNSFs, so they are available for all SecaaS clients to use.

Query NS status. Request the status for a NS instance associated with a SecaaS client.

Query vNSF status. Request the status for a vNSF instance associated with a SecaaS client.

Submit vNSFs. Submit (SecaaS client-agnostic) vNSFs to the Store.

Table 7: RBAC Operations and associated Roles

	SecaaS								
Operation	Auditing	Monitor.	Mainte.	Mngmt	Admin.	Platform Admin.	Develop er	Cyber Agency	Lawful Intercep.
General									
CRUD on Users/Roles						x			
CRUD on SecaaS clients						x			
vNSFs									
Submit vNSFs						x	x		
Onboard vNSFs						x			
Manage vNSFs state				x	x				
Query vNSFs status		x							
List vNSFs for a SecaaS client	x		x		x	x			
Network Services									
CRUD on NSs			x	x	x	x			
Onboard NSs				x	x	x			
Deploy NSs on a SecaaS client				x	x	x			
Manage NS state				x	x				
Query NSs status		x			x	x			
List NSs for a SecaaS client	x		x		x	x			x
Recommendations									

Events notification		x		x	x	x		x	x
Apply recommendations				x	x	x			
List recommendations	x			x	x	x			x

Billing

The billing implementation shall focus on gathering data on vNSF and NS usage for a SecaaS client and compute the associated monthly amount of money to pay. The same holds for developers' revenue. The actual bill and payment system to handle monetary transactions and payment records is considered outside the scope of the project, as well as discounts, promotions, sales, offers, etc. SHIELD produces all the fundamental data for bill documents that another system (either part of the SHIELD ecosystem or an external one) can use as input for invoicing and payments management.

The billing model to implement shall provide for:

<u>Developers</u>

When onboarding a vNSF, the developer defines the price charged to the platform operator for using it. This price shall be based on usage and can be defined as:

- *One-time payment*, where a fee is due just for having the vNSF available in the catalogue.
- *Monthly payment*, where a fee is due for every month the vNSF is used, regardless of how many times it is instantiated during the span of the month.
- *Instantiation-based*, in which a fee is due for every time the vNSF is instantiated, even if during the same day.

<u>Platform Operator</u>

When validating a sandboxed vNSF, the platform operator may define a price to be charged for the vNSF. This price shall be based on instantiation and uses the same model defined for a developer.

The final price to charge for the instantiation of a vNSF is the fee defined by the developer, plus the one set by the operator. To be noted that the price set by the operator may be none, a fixed value, or a percentage based on the developer's fee, either positive or negative, to accommodate potential price increase or reduction.

When a NS is onboarded, the price is set by the platform operator. This price shall be based on instantiation and resorts to the same model defined for a developer. To be noted that the price set by the operator may be none, a fixed value, or a percentage based on the overall amount computed for the vNSFs instantiated in the service.

The total price to charge for the instantiation of a NS is the sum of the fees defined for every vNSF instantiated in the scope of the service, plus the fee defined for the actual service.

<u>SecaaS clients</u>

The price for a security service is set by the platform operator. This price shall be based on instantiation and resorts to the same model defined for a developer. To be noted that the price set by the operator may be none, a fixed value, or a percentage based on the overall amount computed for the NSs instantiated in the service.

The total price to charge for the instantiation of a security service is the sum of the fees defined for every NS instantiated in the scope of the service, plus the fee defined for the actual service bundle.

Comparison with similar technologies

Security Dashboards are currently available both in private and open source initiatives. These dashboards are often based in web browser technologies however desktop or mobile applications are also used. Looking at the features commonly provided by both private and open source initiatives the main goal seems to be allowing end users to quickly understand the status of the controlled environment by flagging the last incidents reported as well as the last remediations suggested/implemented. SHIELD's dashboard aims to align with this approach providing an interface that allows authorised users to quickly check the last events detected by the platform as well as remediation action to mitigate each one of them. Regarding open source initiatives, the most prominent solution is Apache Spot, which is providing a security dashboard that enables the quick visualisation of the last threads detected by its framework. Looking into the private sector, one of the most relevant solutions is the Security dashboard of IBM providing a view on the security health, by showing the top 10 attacks/intruders/victims as well as the blocked actions performed by the security tool. SHIELD aims to extend the features provided by these dashboards, allowing not only the end users to have an intuitive and appealing interface for visualizing the last events detected in the network but also to orchestrate its environment as well as control the available mitigation actions through a NS and vNSF catalogue.

4. Regulatory Compliance Specifications

Ensuring the regulatory compliance of the DARE is a key activity that removes barriers towards the adoption of the SHIELD platform. This section discusses the regulatory framework that applies across EU member states that is relevant to the operational aspects of the DARE. Based on the analysis first provided in D3.2 [3] (some findings are also contained in this deliverable to make it self-contained), SHIELD extracts the regulations-based specifications for the key components of the DARE and illustrates how to start implementing compliance mechanisms. The key focus of this work is on:

- Privacy and Data Protection,
- Obligations of the service provider to Law Enforcement/CERTs,
- Non-discrimination and protection of the individual's rights against behavioural profiling, and
- (to a smaller extent) to EU's Net Neutrality laws.

4.1. EU regulatory framework

General Data Protection Regulation: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance) [50]

The EU General Data Protection Regulation is in place to safeguard the rights of the data subjects and enable them to better control their personal data. The Regulation aims to alleviate the fragmentation in data protection law across EU member states and replace the previous Directive with a unified set of rules. A detailed presentation of the GDPR is included in D3.2.

The network data that are being processed by SHIELD's DARE components may include personal data in the form of IP addresses, emails, requested URIs, etc. SHIELD's DARE, however, does not profile a natural person's behaviour but rather provides a user-agnostic profile of the network. The DARE does not asses a natural person's behaviour (e.g. buying patterns, religious beliefs, health or insecure practices in their employer's network etc.). The DARE monitors the network activity and finds anomalies in atypical traffic patterns. The DARE can then take remedial action against the anomaly (if identified as an attack) and can provide indication of endpoints that are compromised e.g. by malware.

As is also the case with the vNSF ecosystem, processing for the explicit purpose of security can considered lawful. Cybersecurity and protection of network infrastructures against intrusions and breaches can be considered as vital interests of the data subject and as tasks carried out in the public interest. The ISP that uses SHIELD's DARE to secure their network need to inform their clients accordingly and ask for their consent within their contract. In the case of ISP offering services to organisations as SecaaS clients, the processing of network data is required to fulfil a contract. The SecaaS client needs to ensure that the persons utilising their network are informed of the specific network monitoring activities. In this case, the service provider and the SecaaS client act as joint controllers. **Article 7** of the GDPR also states that when consent is given in the context of a written declaration which also concerns other matters (e.g. a contract) it must be presented in a form that is easily distinguishable and comprehensible, otherwise the declaration will not be considered binding. A full analysis of the roles of each actor within the SHIELD use cases is provided in D3.2 (Section 4).

Open Internet Regulation: Regulation (EU) 2015/2120 of the European Parliament and of the Council of 25 November 2015 laying down measures concerning open internet access and amending Directive 2002/22/EC on universal service and users' rights relating to electronic communications networks and services and Regulation (EU) No 531/2012 on roaming on public mobile communications networks within the Union (Text with EEA relevance). [51]

The Open Internet Regulation establishes the circumstances where traffic classification and management are legitimate. It lays down specific net neutrality rules and governs the way ISPs may choose to manage the traffic that passes through their networks, while ensuring equal and non-discriminatory treatment of traffic. The SHIELD DARE does not apply rate limiting rules. However, the remediation engine may recommend such an action. The action, and the related security event are logged in the Dashboard, where the user may select to apply it (or roll back a previous selection). If the rate limiting action is applied, it is enforced by a SHIELD vNSF.

ePrivacy Directive: Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications) [52]

The ePrivacy Directive, also known as the EU Cookie Law, sets the rules for the collection of cookies and ensures confidentiality of electronic communications. Currently there is a proposal [53] for a revision of the ePrivacy Directive, to better align it with the GDPR, consider continuing technical innovation, and to transform it into a Regulation. This would mean that the EU Member States would implement the Regulation as-is, as opposed to a Directive which can be implemented in any way considered suitable by the Member States. The proposal for the Regulation was released on January 2017. The SHIELD Dashboard shall request the user consent on the use of cookies. Such cookies shall only be used in the scope of a web session management and will not be persisted once the session is over. Furthermore, no user or browsing data is collected for tracking user preferences or behaviour patterns.

Data protection in criminal investigations: Directive (EU) 2016/680 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data by competent authorities for the purposes of the prevention, investigation, detection or prosecution of criminal offences or the execution of criminal penalties, and on the free movement of such data, and repealing Council Framework Decision 2008/977/JHA. [54]

Network Information Security Directive: Directive (EU) 2016/1148 of the European Parliament and of the Council of 6 July 2016 concerning measures for a high common level of security of network and information systems across the Union [55]

Although these directives do not apply directly to SHIELD, they are relevant as service providers may be required to cooperate with law enforcement in a criminal investigation or with appropriate cybersecurity agencies in case of a cyberattack. The DARE may expose APIs for exchange of information with relevant third parties under these directives, although the exact use of such APIs would be defined by the ISP or SecaaS client and their internal policies regarding statutory process (unless access is court-mandated).

Non-discrimination:

Council Directive 2000/78/EC of 27 November 2000 establishing a general framework for equal treatment in employment and occupation. [56]

European Charter of Fundamental Human Rights [57], esp. Article 8(1) on the protection of personal data

Treaty of Amsterdam [58] (1997/1999 establishing the protected grounds against discrimination) & Treaty of Lisbon [59] (2007/2009 making the ECHR Bill of Rights legally binding)

Council of Europe recommendations on profiling: Recommendation CM/Rec(2010)13 of the Committee of Ministers to member states on the protection of individuals with regard to automatic processing of personal data in the context of profiling. [60]

Non-discrimination is particularly relevant in the case of SHIELD since the DARE components utilise advanced machine learning algorithms to identify anomalies in data traffic. Although the non-discrimination body of law in the EU regards access to employment, education etc. which are out of the scope of SHIELD, we can consider some basic principles and definitions to be free-standing. Access to the Internet can be regarded as a basic service that should be available to all citizens and any discriminatory practices should be abolished. Although SHIELD's DARE does not profile the user's behaviour for cybersecurity, some definitions should be in place, for future reference:

- The entry into force of the Treaty of Amsterdam in 1997, enabled the European Commission to legislate on non-discrimination based on **defined protected grounds** which **include gender, age, race, ethnicity, religion, belief, age, disability and sexual orientation**. The GDPR considers data that may expose these aspects of the data subject as "special category" data.
- Protection against discrimination is not only present in EU Law but also within the **European Charter of Human Rights (ECHR)** that was proclaimed by the European Union and the Member States in 2000. The ECHR declared the fundamental human rights to be protected and became legally binding after the 2009 Treaty of Lisbon.
- Most definitions in EU law and ECHR regard cases of direct discrimination. The EU Agency of Fundamental Rights (FRA) [61], however, further defines **indirect discrimination**, when a rule that appears to be neutral affects a specific group of citizens in a significantly more negative way, by comparison to others in a similar situation. It also defines **harassment and instruction to discriminate** as violating the dignity of a person.

Hence, any data processing component that profiles aspects of the data subject with respect to these protected grounds, should have safeguards in place to ensure that processing is lawful and that such information cannot be misused and lead to discriminatory practices. The Council of Europe has published a recommendation on safeguards for processing that leads to profiling, although this predates the GDPR and there was no legal definition of profiling at the time.

4.2. Best practices

D3.2 initiated an analysis of best practices with respect to privacy and data protection. Although the work was performed for the vNSF ecosystem, there are a lot of parallels with the DARE. In summary, the principle of Privacy-by-design focuses on maintaining a user's privacy by introducing appropriate safeguards as a software product is being developed. This is based on design improvements such as data anonymization, data minimization, etc. and is distilled in the following foundational principles [62]:

1. **Proactive not reactive; preventative not remedial:** the approach should be characterised by proactive measures that come before-the-fact.
2. **Privacy as the default setting:** Even if a user does not set specific policies, their privacy is still, automatically protected.
3. **Privacy embedded into design:** Privacy is integral to the system without diminishing functionality and not "bolted on as an add-on".
4. **Full functionality – positive-sum, not zero-sum:** False dichotomies ("privacy vs security") should be avoided and no unnecessary trade-offs should be made.
5. **End-to-end security – full lifecycle protection:** Strong security measures are essential and should apply to the entire data lifecycle. This extends to the introduction of Security-by-design.
6. **Visibility and transparency – keep it open:** Trust is easier to build when there is transparency and the stated promises can be verified across all stakeholders.
7. **Respect for user privacy – keep it user-centric:** Keep in mind the interests of the individual and provide privacy defaults, notices and empowering user-friendly options.

The development of a system according to Privacy-by-Design principles, should be complemented by a Data Protection Impact Assessment that clearly maps the possible risks to possible mitigation measures, and provides detailed information on how data are being stored, handled etc. within the DARE. OWASP maintains a list of top 10 Privacy risks and related countermeasures [63]. In the following table we analyse these risks, adapt them to the SHIELD DARE (like the work performed in D3.2 for the vNSF ecosystem) and propose countermeasures to account for the platform's specificities, which can be adopted in an operational deployment of SHIELD in a production network.

Table 8 Privacy risks and countermeasures.

Privacy Risk	Application to SHIELD	Countermeasures
P1 Web Application Vulnerabilities	Attention should be payed to the virtual machine's hypervisor technology since the DARE modules are hosted in VMs. This applies to the Security Analytics Module and the Data Analysis module that store data in HDFS.	Perform penetration tests, monitor vulnerabilities (including those related to the VM hypervisors), train developers in secure development, install updates, fixes etc.
P2 Operator-sided Data Leakage	Failure to prevent a data leak can result in loss of confidentiality. This applies especially to the HDFS that stores information.	Access control and Identity management following the principle of least privilege, strong encryption for personal data, awareness training, data classification

		and handling policies, data leak prevention/early warning, privacy-by-design, data anonymization/pseudonymization.
P3 Insufficient Data Breach Response	The persons affected by a data leak should be informed. Immediate action should be taken to limit a data breach, which should be followed by remediation measures. This applies to the modules that utilise the HDFS for storage. Breach notification should exist as a feature.	Develop/Test/Maintain an incident response plan, a data breach notification system, determine the scope/scale of the breach, notify the Data Protection Officer, investigate the data breach and provide documentation and reports.
P4 Insufficient Deletion of Personal Data	Appropriate data retention periods should be defined. After the retention period is over the data should be deleted, (or upon request by the data subject). If retention is not necessary, the data should be deleted after processing.	Follow the data minimization principle and adhere to GDPR data subject rights (e.g. right of deletion, restriction of processing, right to be forgotten etc.), document data retention policies, deletion should be verifiable.
P5 Non-transparent policies, terms and conditions	This relates to not providing sufficient information to describe how data is collected, processed, stored, managed etc. This information should always be easily accessible and understandable. SHIELD provides this information in the specifications for each data processing component.	Develop terms and conditions for the SHIELD services, make information available and comprehensible, separate terms and conditions for GDPR in a contract, use visual materials (icons, pictograms etc.), document changes to terms and conditions, keep track of user consent, provide opt-out policies (when feasible). Data Protection information should be available to the client in the dashboard.
P6 Collection of data not required for the primary purpose	The collection of user-related data that are not necessary for the purposes of the system is a major privacy risk. This applies to data that were collected without the data subject's knowledge or consent. Consent procedures are not in place.	The purpose of data collection/processing should be transparent. Data should only be collected for the specified purpose (data reduction/minimization), opt-out policies should be set when feasible, apply conditioned collection (only under specific circumstances). Consent procedures should be available (From the data controller to the data subject – see D3.2 for a detailed explanation of roles and obligations, Section 4)
P7 Sharing of Data with third party	Provision of a user's data to a third party without the user's knowledge and consent. The existence of APIs for third party data exchange should be clear. If data are being monetized the user should be aware.	Proxy the content on self-hosted servers and not directly with a third party, apply tokenization or anonymization, develop a monitoring framework that can whitelist/blacklist third parties, develop appropriate contractual arrangements, monitor user complaints, special

		provisions should be made for cross-border sharing.
P8 Outdated personal data	The use of outdated, incorrect or bogus user data, failure to update or correct the data. In SHIELD this applies particularly to the IPs involved in remediation actions and to the login credentials of the data processors.	Implement a procedure to obtain input from users and update their data, ability to roll back a remediation action in case of false positive attack detection, in case of updates all related subsystems should be aware. ML algorithms should be re-trained with accurate data.
P9 Missing or insufficient Session Expiration	Failure to effectively enforce a session termination. May result in additional data collection without the user's consent or awareness or even to theft of credentials. In SHIELD this applies particularly in the user interfaces requiring login credentials (e.g. administrators, dashboard users etc.)	Automatic session expiration should be set with appropriate expiration times based on the criticality of the application and the data. Session timeout could be configurable, reminder messages to log out can be implemented.
P10 Insecure Data Transfer	Failure to provide data transfers over encrypted and secured channels, may lead to data leaks, failure to limit the leak surface. In SHIELD this applies to the data transfer between vNSF-DARE, DARE-vNSFO.	Send personal data through secure protocols, apply secure configurations, allow connections over secure protocols and disallow unsafe connections, avoid inclusion of personal information in session ID/URL, activate privacy extensions (e.g. in IPv6)
P11 Privacy rights compromise	The use of accumulated network traffic for profiling purposes, and exposure of personally identifiable information by combining such information with other metadata such as IP addresses, URI queries etc.	The DARE deploys transparent procedures that require network data collection for specific cybersecurity-related tasks only. User behavioural profiling is not part of any DARE functionalities.
P12 Multi-user support data leakage	Failure to provide an isolated environment to each user, so as to gain access only to threat results that are relevant to his organization, thus exposing personal information of other users.	The DARE will correlate the IP addresses of each SHIELD-monitored organisation to the overall analysis results, to provide tailored information to the Dashboard, in compliance with the GDPR.

4.3. Regulatory compliance specifications

Based on the analysis performed in D3.2, SHIELD developed a template for the definition of the regulatory compliance specifications for each vNSF. This template is herein adapted to the key DARE components and the Dashboard that store or process personal data.

4.3.1. DARE Compliance Specifications

According to the template that was presented, SHIELD provides the compliance specifications of the SHIELD DARE components Table 9, Table 10 and Table 11.

Table 9 Compliance specifications for the DARE Security Analytics module.

1	General Information	Component Name	Security Analytics module	
		Version	6	
		Developer	Talaia	
		Description	Analyses network traffic and detects anomalies	
		Certification & Standardisation	None	
2	Interfaces and Formats	Data Inputs	NetFlow	
		Data Outputs	Identified Anomalies	
		Data Formats	Nfcapd (input), csv (output)	
3	GDPR applicability	Personal Data	Y	IP addresses
		Special Categories	N	
		Identifiability	N	
4	Data Storage	Data Storage	N	
		Data Retention	N/A	
		Data Encryption	N/A	
		Pseudonymisation	N/A	
		Anonymisation	N/A	
5	Data Processing	Purpose	Cybersecurity, Network Anomaly Detection	
		Profiling	N	Relies on network pattern profiling, not user profiling
		Monetisation	N	
		Data Processing	Proprietary, closed source	
		Data Processor	Proprietary, closed source	
		Data Protection Officer	Appointed by the Service Provider	
		Data Controller	Service Provider/network operator	
		Consent processes	Agreement between user and service provider	
		Lawfulness	The identification and analysis of network anomalies is a lawful activity	
6	Data sharing	Other SHIELD components	Y	vNSFs
		Third parties	N	
		Law enforcement	N	
		Cross-border data sharing	N	
		CERT/CSIRT	N	
7	Data Subject Rights	Right of access	N/A	
		Right of rectification		
		Right to be forgotten		
		Restriction		
		Notification		
		Data portability		
8	Open Internet	Traffic Classification	N	
		Rate Limiting	Y	

9	Non-Discrimination	Potential for misuse	N/A	
10	ePrivacy	Protection of the contents of a communication	N/A	Not applicable, the module does not parse communication contents
		Use of cookies to provide a user experience and track user preferences	N/A	Not applicable, the module does not utilise cookies

Table 10 Compliance specifications for the DARE Data Analysis module.

1	General Information	Component Name	Data Analysis module (Apache Spot)	
		Version	**V1.0**	
		Developer	Infili, Space Hellas, Telefonica	
		Description	Ingests network data and performs anomaly detection	
		Certification & Standardisation	None	
2	Interfaces and Formats	Data Inputs	NetFlow, DNS, Proxy data	
		Data Outputs	Identified Anomalies	
		Data Formats	Nfcapd, pcap (inputs), csv (output), bluecoat logs	
3	GDPR applicability	Personal Data	Y	IP addresses
		Special Categories	N	N
		Identifiability	N	
4	Data Storage	Data Storage	N	
		Data Retention	N/A	
		Data Encryption	N/A	
		Pseudonymisation	N/A	
		Anonymisation	N/A	
5	Data Processing	Purpose	Cybersecurity, Network Anomaly Detection	
		Profiling	N	Does not profile the user, but the whole network
		Monetisation	N	
		Data Processing	Ingestion by Spot Workers, Latent Dirichlet Allocation (LDA) for anomaly detection	
		Data Processor	Spot admin, Appointed by the Service Provider (SP)	
		Data Protection Officer	Appointed by the Service Provider (SP)	
		Data Controller	Service Provider (SP)	
		Consent processes	Within SP contract	
		Lawfulness	Cybersecurity anomaly detection is a lawful use of Apache Spot	
6	Data sharing	Other SHIELD components	Y	vNSFs forward
		Third parties	N	
		Law enforcement	N	
		Cross-border data sharing	N	
		CERT/CSIRT	N	
7		Right of access	N/A	

	Data Subject Rights	Right of rectification		
		Right to be forgotten		
		Restriction		
		Notification		
		Data portability		
8	Open Internet	Traffic Classification	N	
		Rate Limiting	Y	
9	Non-Discrimination	Potential for misuse	N/A	
10	ePrivacy	Protection of the contents of a communication	N/A	
		Use of cookies to provide a user experience and track user preferences	N/A	

Table 11 Compliance specifications for the Recommendation and Remediation Engine.

1	General Information	Component Name	Recommendation and Remediation Engine	
		Version	**V1.0**	
		Developer	POLITO	
		Description	Parses the analytics results and creates security configurations to be applied by the NSs in a medium-level policy abstraction.	
		Certification & Standardisation	None	
2	Interfaces and Formats	Data Inputs	Anomalies reports	
		Data Outputs	Medium-level Security Policy Configuration (MSPL) rules	
		Data Formats	CSV, XML	
3	GDPR applicability	Personal Data	Y	IP address of machines that are part of the anomaly report
		Special Categories	N	
		Identifiability	N	
4	Data Storage	Data Storage	N	
		Data Retention	N/A	
		Data Encryption	N/A	
		Pseudonymisation	N/A	
		Anonymisation	N/A	
5	Data Processing	Purpose	Cybersecurity, Mitigation	
		Profiling	N	
		Monetisation	N	
		Data Processing	Parsing of the anomalies reports, mapping to pre-defined “recipes” of configuration rules targeted for the detected anomalies.	
		Data Processor	Appointed by the Service Provider	
		Data Protection Officer	Appointed by the Service Provider	

		Data Controller	Service Provider	
		Consent processes	Within the Service Provider contract	
		Lawfulness	Recommendation generation is a lawful use of the Recommendation and Remediation engine.	
6	Data sharing	Other SHIELD components	Y	Dashboard receives the mitigation rules
		Third parties	N	
		Law enforcement	N	
		Cross-border data sharing	N	
		CERT/CSIRT	N	
7	Data Subject Rights	Right of access	N/A	
		Right of rectification		
		Right to be forgotten		
		Restriction		
		Notification		
		Data portability		
8	Open Internet	Traffic Classification	N	
		Rate Limiting	Y	
9	Non-Discrimination	Potential for misuse	N/A	
10	ePrivacy	Protection of the contents of a communication	N/A	
		Use of cookies to provide a user experience and track user preferences	N/A	

4.3.2. Dashboard Compliance Specifications

According to the template that was presented, SHIELD provides the compliance specifications of the SHIELD Dashboard components in Table 12.

Table 12 Compliance specifications for the Dashboard.

1	General Information	Component Name	Dashboard
		Version	v1.0
		Developer	ubiwhere
		Description	Provides a web-based Graphical User Interface
		Certification & Standardisation	None
2	Interfaces and Formats	Data Inputs	User login credentials, identified anomalies
		Data Outputs	None

		Data Formats		
3	GDPR applicability	Personal Data	Y	login credentials, IP addresses
		Special Categories	N	
		Identifiability	N	
4	Data Storage	Data Storage	N	
		Data Retention	N/A	
		Data Encryption	N/A	
		Pseudonymisation	N/A	
		Anonymisation	N/A	
5	Data Processing	Purpose	Cybersecurity	
		Profiling	N	
		Monetisation	N	
		Data Processing	Parsing of anomalies detection data to present to the user	
		Data Processor	SecaaS Client admin, Appointed by the Service Provider (SP)	
		Data Protection Officer	Appointed by the Service Provider (SP)	
		Data Controller	Service Provider (SP)	
		Consent processes	Within SP contract	
		Lawfulness	Providing remediation data to the user, which falls into the Recommendation and Remediation engine lawfulness	
6	Data sharing	Other SHIELD components	Y	Send remediation rules to the vNSFO
		Third parties	Y	On-demand to Cybersecurity Agencies, and once vetted by the Service Provider
		Law enforcement	Y	On demand, and once vetted by the Service Provider
		Cross-border data sharing	N	
		CERT/CSIRT	N	
7	Data Subject Rights	Right of access	Y	Only applicable to user credentials as no other data is stored
		Right of rectification		
		Right to be forgotten		
		Restriction		
		Notification		
		Data portability		
8	Open Internet	Traffic Classification	N	
		Rate Limiting	Y	
9	Non-Discrimination	Potential for misuse	N/A	
10	ePrivacy	Protection of the contents of a communication	Y	The communication with the Dashboard is done using HTTPS, where the messages are encrypted using the Transport Layer Security protocol
		Use of cookies to provide a user experience and track user preferences	N/A	Despite the use of cookies, these are strictly limited to session management and have absolutely nothing to do with tracking any kind of user preferences or behaviour

4.4. Compliance and Certification

D3.2 has addressed certification with respect to the vNSF ecosystem. The same basic assumptions are also herein presented for the sake of delivering a coherent document. GDPR Certification of DARE components can be crucial towards their adoption. In SHIELD's case, certification requires the existence of a trusted third party that inspects each of the components and can verify that it is compliant with GDPR and that the information provided in its specifications are accurate.

In checking the compliance of DARE application with the General Data Protection Regulation-GDPR- and its relevant provisions, it is necessary to bear in mind its scope and configuration. DARE function, is the storage and elaboration of data through the deployment of algorithms, so that to derive conclusions concerning any spotted suspicious messaging and circulation of data, which aim for instance to the illegal extraction of data, the defrauding of other users and the conduct of cyber-attacks. Again, the compliance test will make sure to comfort the cornerstone principles of Data and Privacy Protection, the fulfilment of any laid obligations toward the Law Enforcement, the net neutrality in accordance with the laid legislation demands and the non-discrimination and protection of the individual's rights against behavioural profiling.

Specifically, and with respect to the relevant provisions of the GDPR, the following remarks must be made:

Article 2 of the GDPR, which provides for the Material Scope of the Regulation. It is evident that DARE falls within its scope given that apart from the procession of personal data, it uses, DARE, automated means (which are being referred in the first paragraph of the article) and algorithms in implementing the aforesaid procession. Furthermore it is important to be noted that the very scope of DARE' application which is the detection of illegal practices, which pose threats to the interests of third parties and the public interest by and large, is being identified as one of particular importance, which could also entail its exemption from the Regulation's scope, as long as it is being contacted by the competent authorities (article 2, para 2 (d) of the Regulation). This admission sets an important parameter concerning the compliance test of DARE with the GDPR and its scope of application.

Article 5, concerning the Principles relating to processing of personal data. DARE by its scope and design is fully compliant with the principles enumerated in the Article concerning the processing of personal data. The data are being collected and processed lawfully, fairly and in a transparent manner and in order exclusively to detect illegal and detrimental behaviour. To this end the collected data and the way they are being processed will be adequate, relevant and limited to what is necessary for the attainment of the prescribed aforesaid goals. Furthermore, and to avoid false conclusions the collected data will be accurate and updated and every reasonable measure, especially through the deployment of the most suited algorithms, will be taken to this direction. Accordingly, their storage will be last for as long as it is necessary for the scope of their collection and procession. In addition, they will be processed in a manner that ensures their security and their protection from unlawful and unauthorized access and intrusion. All appropriate checks and safeguards, through the sophisticated design of DARE, are being applied to this end. The aforesaid qualities of DARE can be verified by the competent authorities in every given case and request.

Article 6, concerning the lawfulness of processing. DARE's design and application serve the scopes and interests as they are prescribed by the GDPR and especially the protection of the interests of the data subject and any other natural person, while it might also serve the fulfilment of a task assigned in the public interest, along with the instructions of the data subject and/ or in accordance which has given thereto. In any event DARE will be applied in full conformity with the considerations mentioned in Article 6, para.4 and especially the possible consequences of the intended further processing of the data along with the existence of appropriate safeguards, which may include encryption and pseudonymisation.

Article 9, concerning the processing of special categories of personal data. DARE will not be applied, but for the enlisted exceptions and especially the substantial public interest and public health, in the processing of special categories of personal data.

Article 13, concerning information to be provided where personal data are collected from the data subject. DARE enables-by its design and application- its controller to provide the data subject with all the information referred in the Article, in accordance with the laid rules and legislation.

Article 15, concerning the right of access by the data subject. DARE by its design and application, enables the controller to provide the data subject with the information referred in this Article, in accordance with the law and the goals of the processing to the extent they are respectively protected by the provisions of the GDPR.

Article 17, concerning the right to erasure. DARE by its design and application enables the controller to meet all the requests put forward by the data subject, in accordance with the Article and the exceptions thereto.

Article 20, concerning the right (of the data subject) to data portability. DARE by its design and application, enables the controller to meet the requests prescribed in the article, but for the exceptions thereto.

Article 23, concerning the restrictions. DARE by its design and application, can easily adapt to the requests mentioned in the Article and enables its controller or processor to correspond accordingly to the measures adopted by the Union or Member State law. Besides the design and application of DARE serve the very same goals, like the ones endeavoured by the restrictions. DARE operates as a safeguard of legitimacy and facilitates the detection of unlawful and detrimental practices to public order and individuals' legitimate interests.

Article 25, concerning the Data protection by design and by default. DARE by design and to serve its scope deploys efficient methods in the processing of the data, including where appropriate pseudonymisation, as an additional safeguard. Besides the pursuit of its scope presupposes discretion in the collection and processing of data. Hence the principles of data minimization, in terms of their amount, period of storage and range of accessibility, duly apply.

Article 30, concerning the records of processing activities. DARE by design and operation, enables the controller and where applicable his representative, to maintain a record of the processing activities referred in the Article.

Article 32, concerning the Security of processing. DARE by design meets all the requirements of Security mentioned to this Article. To this end, besides the system' s availability to adhere to an approved code of conduct or an approved certification mechanism, once they are established, it possesses the capabilities of pseudonymisation and encryption of personal data along with the specifications to ensure its ongoing confidentiality, integrity, availability and resilience. Accordingly, it has the capabilities to restore the availability and access to personal data in a timely manner, in the event of a physical or technical incident, along with the means for regularly testing, assessing and evaluating the effectiveness of its security of the processing.

Article 35, concerning data protection impact assessment. Although DARE is designed to operate in full compliance with the laid legislation and the safeguards provided by the GDPR, especially with respect to the data subject rights, it has the readiness to proceed with the assessment mentioned in the Article. To this end the assessment will include, once it is so required by the authorities to be established, a systematic description of the envisaged processing operations and their purposed, which fundamentally are the detection of illegal practices and behaviours, the proportionality and necessity of the processing operations in relation to the purposes, the risks to the rights and freedoms of data subjects and the measures envisaged to address these risks, including the safeguards and security measures to be deployed to this end, taking into account the legitimate interests of the affected persons.

Article 37, concerning the designation of the data protection officer. DARE by design and scope, can be easily checked and controlled by the Data protection officer, provided by the Article.

Article 40, concerning the Codes of Conduct. DARE by design and operation, possess all the specifications and safeguards to fulfil the checks and requirements included in the Codes of Conduct of this Article.

According to the GDPR Article 42, "the Member States, the supervisory authorities, the Board and the Commission shall encourage the establishment of data protection certification mechanisms and of data protection seals and marks". The certification should be voluntary and transparent, and the certification body should be granted cooperation and access to the processing.

Article 43 of the GDPR states that certification bodies should be accredited (ISO 17065). DARE by design and operation, has all necessary qualities and specifications to withhold the certification test, provided therein and to fulfil all respective requirements. Accordingly, it has the readiness to provide the competent Bodies and Authorities, once they are established, with all requested information, as it is prescribed by the Article.

Articles 44 and Article 49, concerning the transfers of personal data to third countries or international organizations. DARE, as it is legally operated and supervised will not proceed in any transfer of personal data to third countries or international organizations, unless a decision on the adequacy of the third country or international organization has been issued by the Commission, or appropriate safeguards, based on laid legislation or prior authorization by the competent supervising authorities, have been given. In any event enforceable data subject rights and effective legal remedies for data subjects, must be adequately ensured, before any transfer. Taking into account the primary goal of DARE, which is the protection of legitimate interests, through the detection of illegal practices and behaviours, it is perfectly suited to serve the exception of Article 49,para1 ,d ,concerning the transfer of data to third countries and international organizations, in cases of important reasons of public interests, even in the absence of the preconditions envisaged in Article 44 through 46.

Article 82, concerning the right to compensation and liability. DARE by design and operation fully respects the provisions of the GDPR. Its application does not leave room for infringements which could expose the controller or processor to liabilities. Thus, the consequences of any damages caused by the illegal processing of data, are fully acknowledged.

Article 85, concerning the processing and freedom of expression and information. DARE by design and scope of action, is not processing data for journalistic, academic, artistic or literary purposes and is not oriented to take advantage of the relevant derogations, mentioned in the Article.

Article 88, concerning the processing in the context of employment. DARE, is being designed and will be operated in full respect of the specificities of employees' rights and the respective limits in the processing of their data in the employment context. It pursues its goals, striking a delicate balance with the sensitivities concerning the employees' status and the employers' and customers' rights as well.

Article 89, concerning the safeguards and derogations relating to processing for archiving purposes in the public interest, scientific or historical research purposes or statistical purposes. DARE notwithstanding does not pursue the purposes identified in the Article, possess the capabilities and technical safeguards to fulfil the relevant requirements and to enable the pursuit of the aforesaid purposes in the public interest.

DARE by design and operational capabilities, fulfil in their entirety the provisions of GDPR, to the extent they apply and regulate its operation and safeguards both the fundamental rights of the data subject and the legitimate interests of the affected, by the processed data, persons. It is a tool for the efficient operation of the cyber transactions and –while fully respects the fundamental rights of the data subject, in accordance with the laid legislation, checks and balances -contributes in the boosting and safeguarding of the digital economy.

As GDPR is being implemented in each Member State, it is expected that multiple data protection certification providers will be accredited with the relevant national authorities. Thus, it will be possible in the future for DARE developers to get their products certified for GDPR compliance. At consortium level, SHIELD partners intend to reach out to certification bodies that are accredited for GDPR compliance certifications, but it is uncertain if certifications will be available across all member states by the date of the project's completion (February 2018).

Apart from the GDPR, compliance with well-known standards and privacy reference frameworks can be considered. **ISO/IEC 27001 Information Security Management Systems** is a well-known international standard for information security that provides a set of standardised requirements for an information security management system (ISMS). **ISO/IEC 27018**[1] similarly defines guidelines for implementing personal data protections and specifies controls within ISO/IEC 27001. ISO certification in this context can be considered suitable for SHIELD's case. **ISO/IEC 29100**[2] provides a privacy framework. The **OASIS Privacy Management Reference Model**[3] can also be considered, as an open standard for privacy and data protection.

[1] Information technology -- Security techniques -- Code of practice for protection of personally identifiable information (PII) in public clouds acting as PII processors https://www.iso.org/standard/61498.html
[2] ISO/IEC 29100:2011 https://www.iso.org/standard/45123.html
[3] OASIS PMRM https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=pmrm

5. Conclusions

5.1. Status of the usable information-driven engine

Using the requirements and the high-level design of D2.2, T4.1 has performed a detailed analysis and has produced a detailed design and architecture, where all the WP4 subcomponents are explained, their workflows are shown, the communication between them is defined and a study about the implementation technologies is exposed. The results are the present report on the low-level architecture and design of the DARE and the Dashboard (the WP4 components), as well as a transformation from the requirements to the technical specifications, and a choice of the technologies used.

We have realised that due to the work that has been done during the first year of the project, several subcomponents have been redefined, mainly because of changes in the state-of-the-art from the moment when the proposal was submitted and because of the more extensive knowledge gained within the consortium. However, none of these changes introduce a major shift from the overall technical approach of the project, as laid out in the DoA.

Although we expect to acquire new knowledge and get more insights during the development phase, the consortium does not envision major adjustments during the updates of the design deliverables (D2.2 - M17, D3.2-M19 and D4.2-M19). Note that the work exposed in this deliverable perfectly separates the Architecture and Design (blocks and workflows), the specifications (requirements from the technical point of view) and the implementation (the technologies used). This separation isolates the subcomponents in a way that the implications of a change in any of these aspects (architecture, design, specifications and implementation) will be minimised.

As an Innovation Action, SHIELD's vision is to leverage state-of-the-art techniques and try not to reinvent the wheel. To this end, SHIELD has studied the most mature open source technologies and has concluded that Apache Spot will be the main solution to be reused and improved to build the DARE. Apache Spot has some of the most important functionalities needed by the DARE (ingestion, data treatment, extensible analytic framework and a dashboard) however, it is still missing some relevant aspects needed by SHIELD. Firstly, Apache Spot has been built to be a batch solution and although streaming technologies have been considered, the collection of data is completely centralised (workers read a folder for new files). This is not enough for SHIELD since one of the envisioned functionalities is the capacity to process vNSF logs and alerts in real-time. Secondly, the platform offers an anomaly detection algorithm based on probabilities of events however, neither classification of threats is being done nor real-time processing. Moreover, since Spot is completely lacking threat mitigation and recommendation functionalities, these will be originally developed, so that the capabilities of the DARE are in accordance with what was initially envisioned in the DoA. Finally, as SHIELD must integrate information from multiple sources in the Dashboard (Store, vNSF Orchestrator, recommendations, and results from Security engine and results from Cognitive engine), it will not use the dashboard provided by Spot, but will directly use the API provided by the analytics framework.

With all these aspects in mind, we conclude the second iteration of the project design and we enter the second iteration of the development phase, having drafted a clear technical roadmap till the end of the project.

5.2. Future work

The work for T4.1 “Engine infrastructure and software: specifications, design and architecture” concludes with this document, which provides the results for the second design cycle in SHIELD. Tasks 4.2-4.5 continue the work for the development of the subcomponents for the DARE and the Dashboard. A rigorous testing plan is in place to validate the new components, based on the work in D5.1. Results of WP4 activities will be presented in SHIELD’s upcoming demonstrations as presented in the D2.2 demonstration roadmap. D3.3 “Integrated secure framework ready for experiments” will report all the results stemming from WP4 developments and accompany the delivery of the WP4 prototypes.

6. References

[1] "Apache Spot," [Online]. Available: http://spot.incubator.apache.org/. [Accessed March 2018].

[2] SHIELD consortium, "D2.2 Updated Requirements, KPIs, design and architecture," 2017.

[3] SHIELD consortium, "D3.2 Updated specifications, design and architecture of the vNSF ecosystem," 2018.

[4] SHIELD consortium, "D2.1 Requirements, KPIs, design and architecture," 2017.

[5] SHIELD consortium, "D4.1 Specifications, design, and architecture for the usable information driven engine," 2017.

[6] SHIELD consortium, "D3.1 Specifications, design and architecture for the vNSF ecosystem," 2017.

[7] SHIELD consortium, "D5.1 Integration results of SHIELD HW/SW modules," 2017.

[8] SHIELD consortium, "D2.3 Business models evaluation," 2018.

[9] "NetFlow," [Online]. Available: https://netflow.us/. [Accessed March 2018].

[10] "Cloudera CDH," [Online]. Available: https://www.cloudera.com/products/open-source/apache-hadoop/key-cdh-components.html. [Accessed March 2018].

[11] "HDFS Architecture Guide," [Online]. Available: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html. [Accessed March 2018].

[12] "Apache Hive," [Online]. Available: https://hive.apache.org. [Accessed March 2018].

[13] "Apache Spark," [Online]. Available: https://spark.apache.org/. [Accessed March 2018].

[14] "React," [Online]. Available: https://reactjs.org/. [Accessed March 2018].

[15] "Flux," [Online]. Available: https://facebook.github.io/flux/. [Accessed March 2018].

[16] "IPython," [Online]. Available: https://ipython.org/. [Accessed March 2018].

[17] "GraphQL," [Online]. Available: http://graphql.org/. [Accessed March 2018].

[18] "Apache Parquet," [Online]. Available: https://parquet.apache.org/. [Accessed March 2018].

[19] "Apache Hadoop," [Online]. Available: https://hadoop.apache.org/. [Accessed March 2018].

[20] "Apache Kafka," [Online]. Available: https://kafka.apache.org/. [Accessed March 2018].

[21] "pcap files," [Online]. Available: https://wiki.wireshark.org/Development/LibpcapFileFormat. [Accessed March 2018].

[22] "nfdump," [Online]. Available: https://github.com/phaag/nfdump. [Accessed March 2018].

[23] "TShark," [Online]. Available: https://www.wireshark.org/docs/man-pages/tshark.html. [Accessed March 2018].

[24] “Apache Spack Streaming,” [Online]. Available: https://spark.apache.org/streaming/. [Accessed March 2018].

[25] “Apache Storm,” [Online]. Available: https://storm.apache.org/. [Accessed March 2018].

[26] “Apache Pig,” [Online]. Available: https://pig.apache.org/. [Accessed March 2018].

[27] B. e. al, “Latent Dirichlet Allocation,” *Journal of Machine Learning Research,* vol. 3, pp. 993-1022 , 2003.

[28] S. Mukherjee and N. Sharma, “Intrusion Detection using Naive Bayes Classifier with Feature Reduction,” *Procedia Technology,* vol. 4, pp. 119-128, 2012.

[29] M. N. Mohammed and N. Sulaiman, “Intrusion Detection System Based on SVM for WLAN,” *Procedia Technology,* vol. 1, pp. 313-317, 2012.

[30] “Apache Mahout,” [Online]. Available: https://mahout.apache.org/. [Accessed March 2018].

[31] “Scala,” [Online]. Available: http://www.scala-lang.org/. [Accessed March 2018].

[32] “Apache Metron,” [Online]. Available: https://metron.apache.org/. [Accessed March 2018].

[33] “Kentik,” [Online]. Available: https://www.kentik.com/. [Accessed March 2018].

[34] “Deepfield,” [Online]. Available: https://deepfield.com/. [Accessed March 2018].

[35] “Arbor,” [Online]. Available: http://es.arbornetworks.com/. [Accessed March 2018].

[36] “The SECURED project,” [Online]. Available: http://www.secured-fp7.eu/. [Accessed March 2018].

[37] “Swagger Framework,” [Online]. Available: http://swagger.io/. [Accessed March 2018].

[38] “MongoDB,” [Online]. Available: https://www.mongodb.com/. [Accessed March 2018].

[39] “Redis,” [Online]. Available: https://redis.io/. [Accessed March 2018].

[40] “Apache Cassandra,” [Online]. Available: https://cassandra.apache.org/. [Accessed March 2018].

[41] A. M. K. N. a. A. W. Y. Bartal, “Firmato: A novel firewall management toolkit,” *ACM Transactions on Computer Systems,* vol. 22, no. 4, p. 381–420, November 2004.

[42] P. V. a. A. Prakash, “FACE: A Firewall Analysis and Configuration Engine,” *SAINT05: Symposium on Applications and the Internet,* p. 74–81, February 2005.

[43] F. C. N. C.-B. a. S. P. J. Garcia-Alfaro, “Mirage: A management tool for the analysis and deployment of network security policies,” *Data Privacy Management and Autonomous Spontaneous Security,* vol. 6514, p. 203–215, 2011.

[44] “AngularJs,” [Online]. Available: https://angularjs.org/. [Accessed March 2018].

[45] “T-Nova,” [Online]. Available: http://www.t-nova.eu/. [Accessed March 2018].

[46] “SELFNET,” [Online]. Available: https://selfnet-5g.eu/. [Accessed March 2018].

[47] “SONATA,” [Online]. Available: http://sonata-nfv.eu/. [Accessed March 2018].

[48] “SVG,” [Online]. Available: https://www.w3.org/Graphics/SVG/. [Accessed March 2018].

[49] “D3js,” [Online]. Available: https://d3js.org/. [Accessed March 2018].

[50] "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Da," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32016R0679. [Accessed March 2018].

[51] "Regulation (EU) 2015/2120 of the European Parliament and of the Council of 25 November 2015 laying down measures concerning open internet access and amending Directive 2002/22/EC on universal service and users' rights relating to electronic communications," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1520936363145&uri=CELEX:32015R2120. [Accessed March 2018].

[52] "Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications)," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32002L0058. [Accessed March 2018].

[53] "Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL concerning the respect for private life and the protection of personal data in electronic communications and repealing Directive 2002/58/EC (Regulation on Privacy and Electronic Commu," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1520936405198&uri=CELEX:52017PC0010. [Accessed March 2018].

[54] "Directive (EU) 2016/680 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data by competent authorities for the purposes of the prevention, investigation, detectio," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1520936541517&uri=CELEX:32016L0680. [Accessed March 2018].

[55] " Directive (EU) 2016/1148 of the European Parliament and of the Council of 6 July 2016 concerning measures for a high common level of security of network and information systems across the Union," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1520936601568&uri=CELEX:32016L1148. [Accessed March 2018].

[56] "Council Directive 2000/78/EC of 27 November 2000 establishing a general framework for equal treatment in employment and occupation," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?qid=1520936644204&uri=CELEX:32000L0078. [Accessed March 2018].

[57] "Charter of Fundamental Rights of the European Union (2000/C 364/01)," [Online]. Available: http://www.europarl.europa.eu/charter/pdf/text_en.pdf. [Accessed March 2018].

[58] "Treaty of Amsterdam amending the Treaty on European Union, the Treaties establishing the European Communities and certain related acts," [Online]. Available: http://www.europarl.europa.eu/topics/treaty/pdf/amst-en.pdf. [Accessed March 2018].

[59] "Treaty of Lisbon amending the Treaty on European Union and the Treaty establishing the European Community, signed at Lisbon, 13 December 2007," [Online]. Available: http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A12007L%2FTXT.

[60] "Council of Europe Recommendation CM/Rec(2010)13 and explanatory memorandum," [Online]. Available: https://rm.coe.int/16807096c3. [Accessed March 2018].

[61] “European Agency of Fundamental Rights (FRA) Handbook on non-discrimination case law,” [Online]. Available: http://fra.europa.eu/en/publication/2011/handbook-european-non-discrimination-law-2011-edition. [Accessed March 2018].

[62] A. Cavoukian, “Privacy by Design - the 7 foundational principles,” [Online]. Available: www.privacybydesign.ca. [Accessed March 2018].

[63] “OWASP Top 10 Privacy Risks and Countermeasures,” [Online]. Available: https://www.owasp.org/images/0/0a/OWASP_Top_10_Privacy_Countermeasures_v1.0.pdf. [Accessed March 2018].

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

Acronym	Meaning
API	Application Programming Interface
BSS	Business-Support System
CRUD	Create, Read, Update, and Delete
CSS	Cascading Style Sheets
DARE	Data Analysis and Remediation Engine
DDoS	Distributed Denial of Service
DNS	Domain Name System
DoS	Denial of Service
ERC	Ethical and Regulatory Compliance (requirements)
ESB	Enterprise Service Bus
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
HDFS	Hadoop Distributed File System
HTML	Hyper Text Markup Language
IDPS	Intrusion Detection and Prevention System
IP	Internet Protocol
ISP	Internet Service Provider
LDA	Latent Dirichlet Allocation
NF	Non-Functional (requirement)
NFV	Network Function Virtualisation
NS	Network Service
O&M	Operations & Maintenance
OA	Operational Analytics
ODM	Open Data Models

PF	Platform Functional (requirement)
PoP	Point of Presence
REST	REpresentational State Transfer
RSA	Rivest–Shamir–Adleman
SecaaS	Security as a Service
SF	Service Functional (requirement)
SP	Service Provider
SQL	Structured Query Language
SVG	Scalable Vector Graphics
TM	Trust Monitor
UC	Use Case
URI	Uniform Resource Identifier
UUID	Universally unique identifier
vNSF	virtual Network Security Function
vNSFO	vNSF Orchestrator

Annex A. Technical Updates and Revisions

Technical advances and updates in D2.2/D4.2

A.1. The DARE

Hybrid architecture

In D2.1 the DARE functionalities were completely centralised with the envisioned Enterprise Service Bus (ESB) managing the communication between the central DARE and the vNSFs. According to the research done during T4.1, it has been considered that some subcomponents were missing from the architecture. The data collector module is a good example of this situation. This module works with specific formats in the sense that it accepts data from different formats (for example data from DNS and data from a proxy). Hence, one collector must be developed for each format accepted in SHIELD. Moreover, to avoid flooding the network, we consider the possibility that some uncompressed format specific files can be transformed into generic formats which can be compressed before being sent to the central engine. Hence, the Data transformation subcomponent (which replaces the Data Services Centre, a name that could lead to confusion) can either be distributed in the case of specific not compressed formats or centralised for specific formats that already have minimum size.

Moreover, to avoid confusion with commonly defined terms and because the data has been already transformed, the storage module is no longer called “Staging” (which commonly refers to raw and heterogeneous data storage module) but is now called “Distributed File System”.

A streaming service

The envisioned ESB has been replaced by a simpler streaming service. The reason is that SHIELD does not need some of the advanced services provided by ESBs like multi-tenant communications (more than one endpoint for each channel), synchronisation services or bidirectional flows. Instead of this complex system, a streaming service which only deals with reliable message delivery is considered to be enough.

A more complete data analytics framework

The Complex Event Processor (CEP) as well as the processing area have been incorporated in the data analytics framework since it is precisely the machine learning algorithms provided by the data analytics framework the ones that will classify the traffic and manage the events.

A recommendation and remediation engine with a global view

The SHIELD consortium considers that this subcomponent must use not only the information given by the data analytics framework, but also the global view of the system. Hence, the recommendation and remediation engine will consider all the variables needed to recommend or remediate a cybersecurity vulnerability:

- The threats and attacks detected by the data analytics framework.
- The different types of vNSFs provided by the Store and the policies that can be applied to them.
- The vNSFs already deployed in the network.

Using this central position, it will recommend network services (sets of vNSFs) together with the policies and the specific deployment locations with the objective to remediate the detected attack.

A.2. Data acquisition and storage

A new module has been developed to meet the requirements of the distributed architecture. Network files are processed on vNSF-side, and only the text-formatted output is sent through the network, reducing the network throughput. In addition, using the Apache Avro serialization framework, the processed output is converted to an Avro-encoded format before it is published in the Kafka cluster, eliminating corruption issues in the data.

After performing a set of tests between the distributed collectors and workers, we noticed that it is preferable to send small pieces and not the entire processed output to Kafka cluster, providing a steady flow of data between the collectors and the workers. Furthermore, using parallel processes for both collector and worker processes, delays are significantly reduced, especially when there is a huge load of incoming network files, optimizing the performance of the ingestion module.

To take advantage of other data types produced by the vNSFs, like alerts and metrics, new collectors and workers have been implemented. These can be used in both centralized and distributed architecture, without any changes to the original implementation. Lastly, the UUID of the current vNSF is sent among the data to know from which machine these data originate. This information is stored in Hive tables, to be accessible in the future.

A.3. Data analysis

The Cognitive and the Security data analysis modules that comprise the Data Analysis phase have been subjected to extensive testing since D4.1 with various attack simulations, to evaluate and improve the platform's detection efficiency. The outcome of these tests was used as a reference point for future development planning, with a view to fulfilling the project's critical requirements. More specifically:

For the Cognitive Data Analysis module, the lack of a threat classification capability was found to reduce the overall impact of the module, since the detected anomalies were not being categorized as specific threats. This issue is currently being addressed with the development and implementation of a Random Forest classifier whose purpose will be to analyse the detected anomalies provided by Apache Spot's LDA algorithm and assign them to specific threat classes. The upcoming module is expected to exploit the existing state-of-the-art Python and Spark machine learning libraries (e.g. scikit-learn, MLlib etc.) and will support HDFS connectivity. Moreover, several partners are currently experimenting with the development of anomaly detection algorithms that could be used as an alternative solution to the existing one (LDA), resulting in the enrichment of this open-source module. Finally, further modifications to the module that will allow for near-real time analysis have been scheduled for the immediate future.

For the Security Data Analysis module, the output format of the provided results has been altered to be consistent with the one provided by the Cognitive module, as a first step for implementing a unified threat response. In terms of anomaly detection capabilities, an additional functionality that will enable data exfiltration detection is also being studied. In more detail, the study involves the detection of DNS tunnelling-based exfiltration attacks by means of statistical analysis of NetFlow traffic, which will allow the detection of such anomalies without accessing the packet payload. Moreover, the module was modified to be capable of ingesting traffic captures in CSV format. As a result, there is consistency between the data formats that are supported by the tow modules and furthermore, the resources that are required by the collector are reduced as it no longer needs to generate traffic in multiple formats.

A.4. Cybersecurity topologies

The recommendation and remediation engine has been based on the functionalities of the modules for policy generation offered by the SECURED project, but it has been re-written and re-engineered to support the SHIELD workflows and interactions with other infrastructure components. More specifically, the H2MService based on Java has been dropped and its basic capabilities have been developed in a Python project. The SECURED policy abstractions, i.e. the HSPL and MSPL, have been reused for the policy generation process and extended to support the vNSFs security capabilities.

Moreover, the module has been modified to support the selection of pre-defined Network Services from the vNSF Store, hence its Forwarding Graph Generator sub-component has been removed from the internal architecture. According to D2.2, the Service Composition Platform Requirement is no longer part of the Cybersecurity Topologies phase of the DARE.

A.5. The Security Dashboard

The Dashboard was promoted to the only component to interact with third-party entities within the SHIELD framework. Any external interaction with SHIELD, regardless of which internal component should handle it, must be done through the Dashboard. The same applies for user interactions.

The GUI for the Store has been moved to the Dashboard since this is the only component handling user interaction.

Monetization definition was rescoped in which the platform only provides billing data on vNSF and NS usage and does not handle any invoicing or payment related operations. vNSF and NS pricing definition was moved from the Store to the Billing subcomponent in the Dashboard. These changes allow the platform to integrate with external billing systems.

A.6. Listing of changes from D4.1

The following table tracks the changes to this document, as compared to its preliminary (base) version, D4.1.

#	Revision action	Revision description	Tracking notes
1	Updated	Executive summary	Executive summary reflects the new requirements in D2.2 and the scope of the deliverable w.r.t. D4.1
2	Added	Subsections 1.2, 1.3	Added to provide the scope of the document, links to other deliverables and the organisation of its contents
3	Added	Section 2.1 and sub-sections	New section introduced to describe the WP4 guiding principles.
4	Updated	Section 2.1.3, 2.1.4	Reworded billing model and monetization introducing D2.3 considerations.
5	Updated	Section 2.6.1	Updated the internal architecture of the Recommendation and Remediation Engine.
6	Updated	Section 2	Updated billing model and changed the use-case diagram according to the agreements on the SecaaS use-case
7	Updated	Section 3.1, 3.2, 3.3, 3.4	Updated to reflect the finalised requirements in D2.2, and resource requirements for the DARE components and the Security Dashboard.
8	Added	Section 3.1, 3.2, 3.3, 3.4	Added ethical and regulatory compliance (ERC) requirements for the DARE components and the Security Dashboard.
9	Updated	Section 3.2: Specifications	Updated the specifications table with information about the scalability of the Security Analytics module.
10	Updated	Section 3.3 Specifications	Updated specifications table to align with the finalized requirements in D2.2.
11	Added	Section 3.4	Introduced Role-Based Access Control (RBAC), describing the envisaged roles for the SHIELD platform.
12	Added	Section 4	Added to ensure GDPR compliance for the DARE and the Security Dashboard.
13	Added	Section 4	Added compliance specifications for the Security Analytics module.
14	Added	Section 4	Added compliance specifications of the Recommendation and Remediation Engine.
15	Added	Section 4	Added compliance specifications for the Dashboard.
16	Updated	Subsection 5.1	Moved content from old section 5
17	Added	Subsection 5.2	Added future work for WP4.
18	Updated	Section 6	Updated text to include future work for WP4 and conclusions/lessons learned after Y1.
19	Added	Annex A	Tracks changes in technical content and in the text since D4.1.