A coordinated framework for cyber resilient supply chain systems over complex ICT infrastructures

D2.2 IT-1 architectural requirements and design

Document Identification			
Status	Final	**Due Date**	31/08/2021
Version	1.0	**Submission Date**	21/09/2021

Related WP	WP2	**Document Reference**	D2.2
Related Deliverable(s)	D2.1, D3.1, D4,1	**Dissemination Level (*)**	PU
Lead Participant	TUBS	**Lead Author**	Admela Jukan, Jasenka Dizdarević
Contributors	ATOS, SYN, XLAB, TID, UPC, POLITO, OPT, Sonae, STS, Uminho, Capgemini	**Reviewers**	Aleš Černivec, XLAB Nelly Leligou, SYN

Keywords:
Architectural design, cyber resilience constraints and requirements, communication and interfaces

This document is issued within the frame and for the purpose of the FHISY project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 952644. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.

Document Information

List of Contributors	
Name	Partner
Admela Jukan, Jasenka Dizdarević, Marc Michalke, Francisco Carpio, Mounir Bensalem	TUBS
Panos Trakadas, Panagiotis Karkazis	SYN
Joao Costa, Ales Černivec	XLAB
Ignazio Pedone, Cesare Cameroni, Daniele Canavese	POLITO
Eva Marín Tordera, Xavier Masip	UPC
Ana Machado Silva, José Duarte	SONAE
Henrique Santos	UMINHO
José Javier de Vicente	ATOS
Guillermo Jiménez Prieto	CAPGEMINI
Jose Manuel Manjón Cáliz	TID
Grigorios Kalogiannis	STS
Antonis Gonos	OPT

Document History			
Version	Date	Change editors	Changes
0.1	30/4/2021	Admela Jukan, Jasenka Dizdarević, Marc Michalke, Francisco Carpio (TUBS)	Table of content
0.2	28/5/2021	Jasenka Dizdarević, Francisco Carpio (TUBS)	First round of contributions (All partners)
0.3	15/6/2021	Jasenka Dizdarević, Francisco Carpio (TUBS)	Contributions to sections 2,4 and 5 (All partners)
0.4	12/7/2021	Jasenka Dizdarević (TUBS)	Added sections 4.1.6, 5.1, 5.2
0.5	31/7/2021	Jasenka Dizdarević (TUBS)	Harmonization
0.6	4/8/2021	Admela Jukan, Jasenka Dizdarević, Marc Michalke, Francisco Carpio (TUBS)	Ready for project internal review
0.7	27/8/2021	Admela Jukan, Jasenka Dizdarević, Francisco Carpio (TUBS)	Addressed comments received from internal reviewers (XLAB, SYN)
1.0	31/8/2021	Jose Francisco Ruiz (Atos), Juan Alonso (Atos)	Quality assessment and final version to be submitted.

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	Admela Jukan, Jasenka Dizdarević, Marc Michalke, Francisco Carpio (TUBS)	27/08/2021
Quality manager	Alonso, Juan Andres (ATOS)	31/08/2021
Project Coordinator	Ruiz, Jose Francisco (ATOS)	31/08/2021

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
AC	Access Control
AI	Artificial Intelligence
AMQP	Advanced Message Queuing Protocol i
API	Application Programming Interface
BP	Business Platform
CON	Constraint
D2.2	Deliverable number 2 belonging to WP2
DC	Data Consumption
DP	Data Provider
EC	European Commission
EDC	Enforcement and Dynamic Configuration
F2F	Farm to Fork use case
GUI	Graphical User Interface
IdM	Identity Management
IRO	Intent-based Resilience Orchestrator and Dashboard
IT	Information Technology
MEC	Multi-Access EDGE Computing
ML	Machine Learning
NED	The Network Edge Device
NFV	Network Functions Virtualisation
NSF	Network Security Function
OT	Operational Technology
PoC	Proof of Concept
RAE	Risk Assessment Engine
REQ	Requirement
SADE	Secure Autonomous Driving function at the Edge use case
SCM	Security Assurance & Certification Manager
SEM	Secure Edge Node
SIA	Secure Infrastructure Abstraction

Abbreviation / acronym	Description
SIEM	Security Information and Event Management
SPI	Security & Privacy Data Space Infrastructure
TIM	Trust & Incident Manager
VNF	Virtual Network Functions
WAF	Web Application Firewall
WBP	Wood-based Panels use case
WP	Work Package
XACML	EXtensible Access Control Markup Language
XL-SIEM	Cross-Layer Security Information and Event Management

Executive Summary

This document represents an initial version of the FISHY architectural solution for cyber resilience provisioning in ICT-based supply chains, which will serve as the foundation of the FISHY platform functional specification and is the main outcome of the work done in WP2 in the task "T2.2 – Cyber resilience related constraints and requirements" and in the task "T2.3 - Architectural design".

Starting with the main high-level concepts of the FISHY platform, it introduces the so-called "action areas of concern", which define the mapping of different components to different areas of ICT supply chain where FISHY will be deployed. This refers to a logical division of FISHY platform structure into organizations, realms and domains. Then, the focus moves on describing the requirements and constraints which need to be satisfied and met by the proposed architectural solution, which have been studied in task "T2.2 - Cyber resilience related constraints and requirements". This document also includes an overview of FISHY use cases, indicating their mapping to the proposed reference architecture. Following the initial reference architecture positioning and identification of imposed requirements/constraints, the document describes the individual building components of FISHY platform. Here, the structuring concept relies on a design of security, trustworthiness and certification layer with the main objective of ensuring system resilience of the entire ICT supply chain, as well as correctly auditing and assessing ICT systems to ensure a correct implementation of security policies.

The document also includes initial specification of the interfaces between the architectural building blocks, which will drive the cross-functional integration of the FISHY platform in WP5. Finally, the reference architecture described in this document will be used to consolidate convergence between the activities in technical work packages WP3, WP4 and WP5, in charge of implementing different functional components envisioned in the FISHY reference architecture.

1 Introduction

1.1 Purpose of the document

Deliverable D2.2 "IT-1 architectural requirements and design" is the first iteration of the two deliverables, which will be produced by Tasks T2.2 and T2.3 within WP2, whose main objective is providing an architectural design of the FISHY novel solution for cyber resilience provisioning in ICT based supply chains satisfying their requirements. The second version of architectural design will be included in D2.4 and will take into consideration the results of the piloting activities that take place in WP6.

This deliverable describes the initial FISHY architecture, setting up a reference framework for the activities of the technical WPs. Section 2 starts with a short overview of FISHY platform and its proposed structure (a division based on a concept of so called "action areas of concern" - organizations, realms and domains), defined to maximize the flexibility of the FISHY deployment potential. The identification of requirements and constraints is described in Section 3, as the deployment options defined affect the operations and functionalities that the FISHY platform and its components should support. This section also includes a description of different instantiation of the action areas of concern for each use case. With the requirements defined, Section 4 goes into a detailed description of FISHY components, their functionalities and interfaces. The deliverable also provides a defined FISHY operational data flow, which will be particularly useful to partners as a reference, when developing their individual components, in order to identify interactions and interfaces with other components.

The FISHY architecture described in this document will serve as a foundation for all technical activities as they progress in parallel, thus ensuring the successful execution of the development tasks in WP3, WP4 and WP5.

1.2 Relation to other project work

This deliverable builds on the work done in other tasks of WP2, tasks T2.1 and T2.2 where the set of technological imperatives and cyber resilience related requirements and constraints was identified and analysed, motivating the design of the FISHY architecture. It also takes into consideration initial designs of WP3 system component Trust Manager (TM) described in D3.1 [1] and WP4 system component Security and Certification Manager described in D4.1 [2]. The outcomes of the architectural design will contribute to the FISHY platform component development under WP3, WP4 and WP5, as well as the platform integration.

1.3 Structure of the document

This document is structured in the following way:

- **Section 2** gives the high-level overview of the FISHY platform structure and defines its action areas of concern
- **Section 3** describes functional and non-functional requirements and constraints imposed on the FISHY framework by use cases, as well as their mapping to FISHY platform structure
- **Section 4** describes the FISHY platform building components
- **Section 5** describes the communication and interfaces among FISHY components
- **Section 6** determines the FISHY operational data flow (hello-world).

2 FISHY system description

This section describes a high-level conceptual specification of the FISHY architecture, which is to be used as a starting point in developing a coordinated framework for cyber resilient ICT supply chain systems, built upon tools and methodologies which will enable strong security assurance and certification management, trust management, data and privacy management. In this section we also introduce action areas of concern of FISHY platform, which help us define deployment options and alternatives which render our solution flexible.

2.1 Reference architecture

This preliminary version of the FISHY architecture is based on the architecture presented in the FISHY proposal and is here shown in Figure 1. It consists of a set of building modules which include: Security Assurance & Certification Manager (SCM), Enforcement and Dynamic Configuration (EDC), Trust & Incident Manager (TIM), Security & Privacy Data Space Infrastructure (SPI), Intent-based Resilience Orchestrator & Dashboard (IRO) and Secure Infrastructure Abstraction (SIA). Next, we give an overview of the different modules that have been proposed.

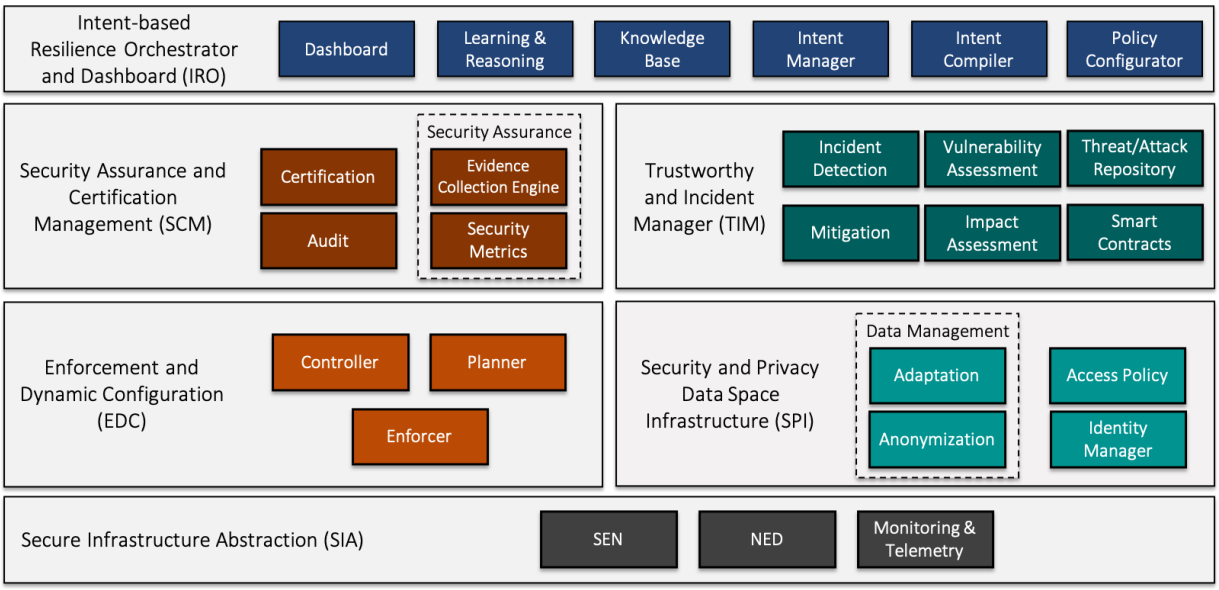

Figure 1: FISHY architecture

SCM module combines methods, procedures, and tools, which enable continuous assessment of the security posture of the complex ICT systems. It also coordinates the monitoring process, the automated evidence-based security reporting and the certification towards ensuring that the required security policies are correctly implemented. EDC module is in charge of security policies enforcement and configuring the specific infrastructure and network security functions (NSF) to ensure resilience. Specification and internal design of SCM and EDC modules, along with the details on the functionalities and interactions of their corresponding subcomponents, have been described in the deliverable D4.1 [2]. SPI module is in charge of gathering and storing data produced by low-level components of the ICT supply chains and interfacing them with the higher-level modules of ICT infrastructure. It also defines the data management procedures, which would incorporate corresponding encryption and anonymization, the tools for assessing the security of the stakeholder's device, component or/and system. TIM module encompasses a set of tools, such as vulnerability and risk estimation, along with

incident detection and management, with a goal of developing mechanisms, which would ensure security assessment of the stakeholder's supply chains. Specification and design of TIM and SPI modules, as in case of SCM and EDC modules, along with the details on the functionalities and interactions of their corresponding sub-components have been described in deliverable D3.1 [1].

IRO module is in charge of translating security requirements within the FISHY platform into intents and in turn corresponding security workflows and policies, combining known intent-based techniques and AI to automate the processing and intent management. In addition, this module integrates FISHY dashboard interface for system security, control and performance monitoring facilitation. SIA module facilitates connectivity among different infrastructures (IoT, edge, cloud) and the FISHY platform, controlling connectivity and providing telemetry of the network, in order to adapt it to other modules. The sub-components of the main FISHY modules and their specific functionalities are detailed in Section 4.

It is important to note that the initial FISHY architecture (as described in the proposal), has been slightly modified in some aspects. First, the "Monitoring & Telemetry" module has been moved outside the IRO to SIA, as it is related to the telemetry data coming from network, connectivity, and infrastructure (cloud, disk, network topology, etc.). Furthermore, the EDC's module Resilience Manager has been removed (compared to the proposal architecture), as its features have been absorbed by the EDC's Controller module (as will be detailed in section 4).

The FISHY platform will consider multiple types of users with different privileges. For instance, it has already been agreed that the FISHY platform administrator will have full privileges. Other potential roles include the supply chain administrator (in charge of system setup and definition of certain metrics and rules), as well as administrators of infrastructure participating in a supply chain with rights to set rules and metrics only for the area-system they administer.

2.2 Platform structure and action areas

It was important for us to investigate possible deployment alternatives before proceeding to the identification of requirements as the deployment options affect a) the operations and functionality that the FISHY platform and its components should support and b) the way this is marketed and pushed into the market. To further clarify the situation, in our attempt to define where each component can be deployed, we came up with very different opinions, even within the consortium, which were filed by the different ICT-based supply chain that each partner had in mind. To maximise the flexibility of the FISHY platform deployment potential, we came up with the definition of the "action areas of concern".

To this end Figure 2 shows a high-level overview of the FISHY platform, divided into three different "action areas of concern": organization, realm and domain. ***Organizations*** can be either companies, consortiums or law enforcement entities that cooperate with the FISHY supply chain platform. Every organization can be divided into different ***realms*** according to the cybersecurity constraints, policies or rules. Following the same approach, within every realm one or more ***domains*** can be established between groups of assets with certain relationships, for instance, the same network, location or infrastructure.

Some of the FISHY components are intended to run within the organization (i.e. EDC, SPI and SIA) and some others externally and will most likely be managed by a third party service provider (i.e. IRO, SCM and TIM) either in the cloud or on-premises. EDC, SPI and SIA are deployed per domain basis, so even within the same organization, those components are independently deployed over different domains regardless to which realm they belong to. IRO, SCM and TIM, on the other hand, are logically centralized components running outside any organization, composing the FISHY Control Service. Different instantiation of this concept for each use case will be described in the following section.

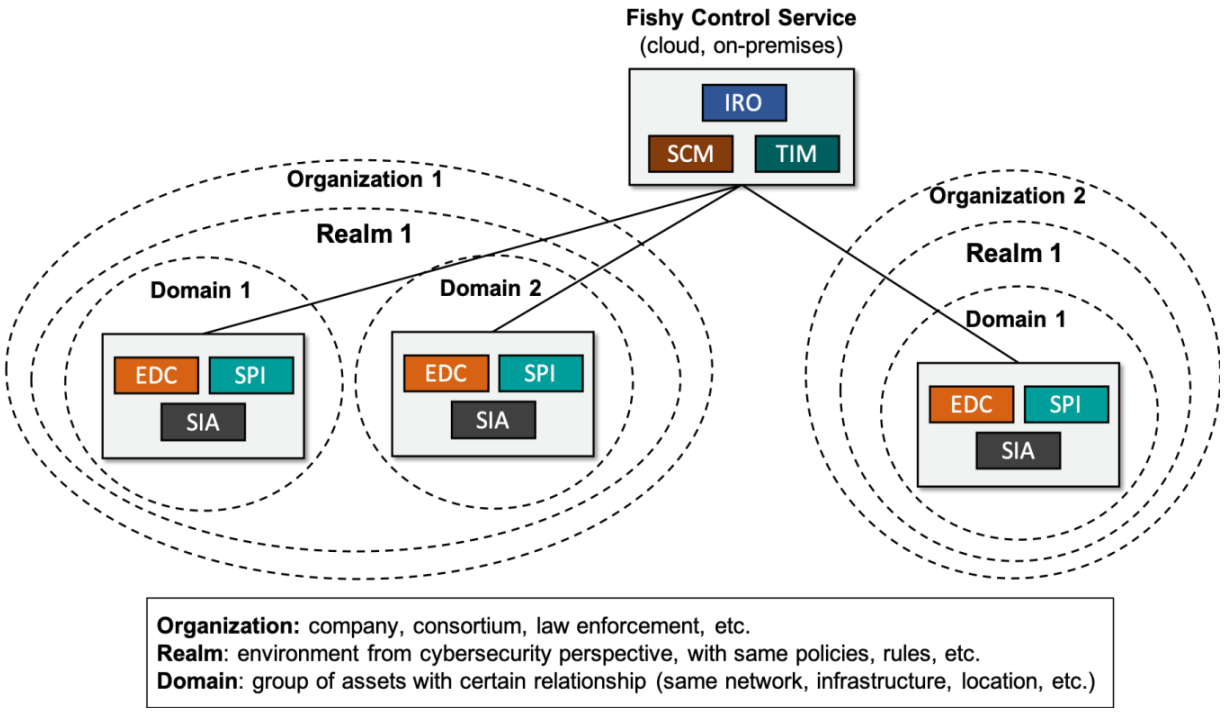

Figure 2: FISHY platform structure

3 Cyber resilience related constraints and requirements

To identify a clear list of concrete requirements and constraints that the FISHY platform should meet/respect, FISHY partners have studied carefully the exemplar use cases brought by the pilot partners. It is worth pointing out that pilot-specific requirements are not included in this deliverable as these are relevant only to the implementation and piloting activities (WP6) and do not affect the high-level architectural design. Thus, in this deliverable, we study the three use cases as an inspiration for our architectural work and any pilot specific requirements are described in D6.1, which is also due M12.

The rest of the chapter is organised in three subsections: in the first subsection we include a narrative of each of the three pilot use case indicating their mapping to the FISHY reference architecture and platform structure (described in Section 2), as well as the points which are sources of requirements and constraints that feed into the architectural design, which will be later described in Section 4. Then, in the next subsections, we tabulate the requirements and constraints to be taken into consideration during the design of the architecture.

3.1 Use cases description and mapping to FISHY architecture

3.1.1 Use case 1: Farm to Fork (F2F) supply chain

The Farm to Fork (F2F) pilot considers the route of a food product (i.e. table grapes) over a number of heterogeneous business segments and corresponding IT infrastructures, which are deployed along different networks comprising a supply chain. The high-level overview of the pilot architecture and accompanied/hosted services are shown in the Figure 3.

This pilot aims to establish and demonstrate a provenance chain Business Platform (BP) that covers the farming, storage, distribution (logistics), and retail sub-domains of the F2F route. The setup comprises a decentralized, geographically dispersed system consisting of heterogeneous sensors, IoT platforms and networks. FISHY platform is gathering data from all the above sources of data/information (as depicted in Figure 3) and processes it in a way that: 1) data becomes anonymized and adapted in the Security & Privacy Data Space Infrastructure, 2) processed in terms of security and trust in the Trust & Incident Manager and 3) are used to trigger reconfiguration suggestions or alerts through the Enforcement and Dynamic Configuration module.

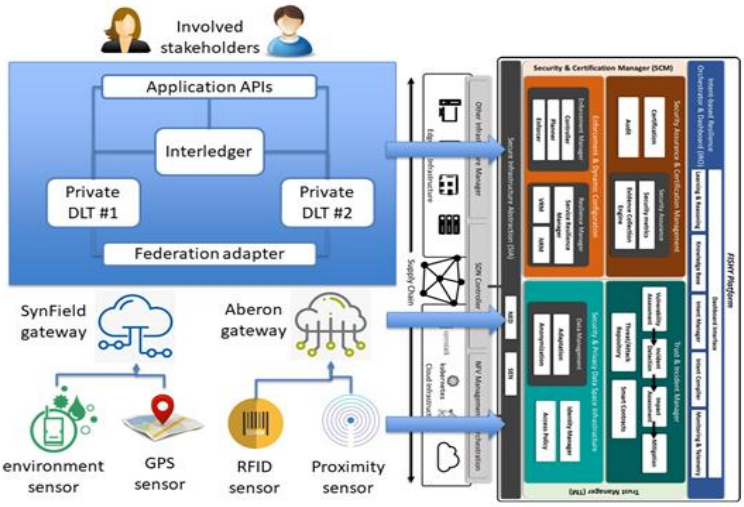

Figure 3: F2F platform and its connection with the FISHY platform

3.1.1.1 Mapping of F2F use case to the FISHY reference architecture and platform structure

F2F supply chain consists of three organizations, namely, the company that provides the system for the collection and processing of data from the farm field (organization 1), the network and system of the transportation organization (organization 2) and the warehouse organization where the appropriate software has been installed (organization 3).

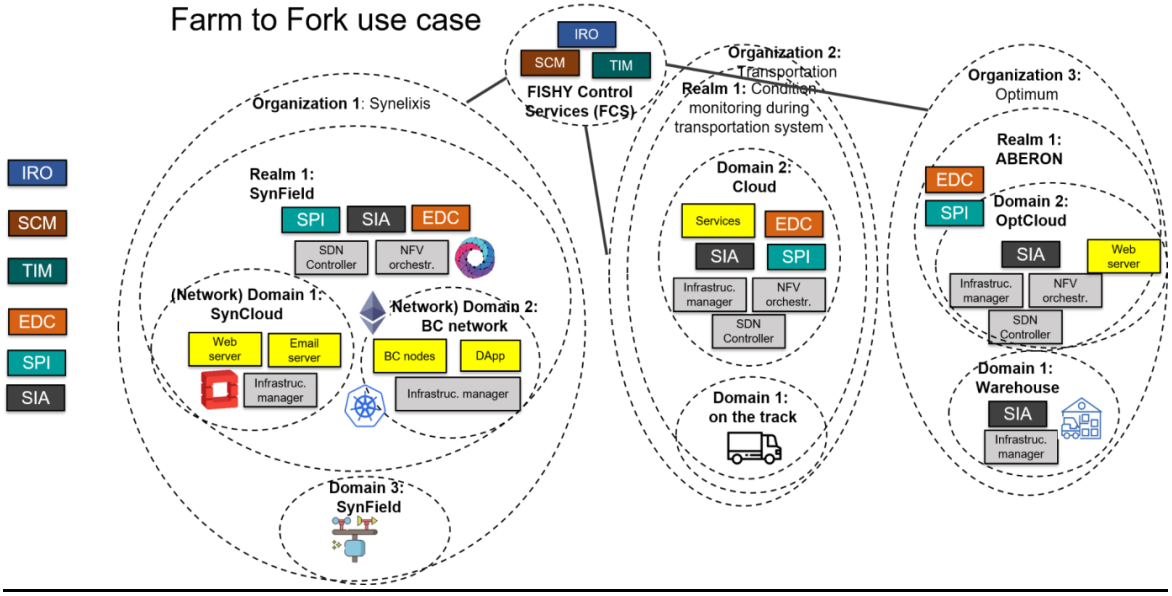

Figure 4: The deployment of FISHY platform components in the F2F use case

Figure 4 depicts how this use case takes advantage of the dynamicity of the architecture design of the FISHY platform. In particular, each organization can consist of one or more realms and, as in this case, each realm can consist of more than one domain. For example, organization 1 includes three domains that can be considered as three different networks: the first one (domain 1) includes servers deployed on top of the infrastructure manager (that in this particular organization is based on Openstack), while the second domain is based on Kubernetes clusters and hosts Ethereum instance along with several software blocks needed to support the services provided by this organization. Finally, domain 3 includes the geographically distributed devices that collect data from the fields. Similarly, each one of the rest, namely organization 2 and 3, as depicted in the above figure, includes several domains, hosting services and software components that collectively define the F2F supply chain.

It is also highlighted that Secure Infrastructure Abstraction (SIA) can be considered as part of some domains, where in other cases, one SIA instance can accommodate the needs (abstraction) of more than one domain (for example compare domains and SIA in organizations 1 and 3). Another important aspect of FISHY architecture that F2F use case takes advantage of is the dynamicity of the logical creation of realms, where SPI components can be hosted. Also, this use case considers that the FISHY platform will be an independent platform, offering a wealth of services based on the installed tools. In this first pass on how FISHY architectural components can assist the accomplishment of the F2F requirements, the "centralised part" of the FISHY platform is depicted to include the following components, named as FISHY Control Services (FCS): Intent-based Resilience Orchestrator (IRO), Trust and Incident Manager (TIM), and Security Assurance and Certification Manager (SCM).

The logical flow of data collection, processing, assessment/detection, and policy enforcement, depends on the specifications of each use case, but for the purposes of F2F use case, the flow is the following: SIA provides the abstraction to manage the resources of each infrastructure as well as the lifecycle of the running services. The SPI component defines the dataspace where the data lives in. These data-sets are

then forwarded or transmitted (according to different technical solutions supported by FISHY) to the TIM component that provides all the needed tools for vulnerability assessment, incident detection and mitigation, along with the responsibility to maintain the repository of threats and attacks and the SCM that manages the certification part of the supply chain mechanisms. Finally, upon the activation of predefined rules, the EDC component takes the responsibility to enforce the specific policy, followed by the appropriate configuration changes on the domain or the realm of the organization.

3.1.1.2 F2F use case constraints and requirements

To capture concrete requirements and constraints (using the abbreviations REQ and CON, respectively), we describe the details of this use case and two storylines: one referring to the F2F supply chain user and another referring to the FISHY platform user.

Storyline 1: Requirements and constraints arising from the F2F platform user

Step 1) The farmer cultivates an area of tens of hectares of vineyard, where she grows table grapes. In the cultivated area, a SynField smart farming installation exists to help her to control and monitor crop production and quality **[REQ-F2F-01] [REQ-F2F-02] [CON-F2F-01]**. The SynField node collects and transmits both climate and crop status data to the SynField IoT gateway cloud platform, providing security guarantees **[REQ-F2F-03]** and network resilience **[REQ-F2F-04]**. The producer has an agreement to sell her goods to a specific warehouse. At the suitable time, she communicates with a transportation company that also participates in the supply chain by obtaining boxes, which will be used for the transportation of grapes when they are ready to be harvested. Boxes are the property of the transportation company and each one is identified by a unique RFID tag that is attached to it **[REQ-F2F-05]**. When the producer fills a (group of) box(es), she accesses **[REQ-F2F-06]** the F2F platform web application to insert additional (meta)data about the contained product, e.g. define type and origin of product, location of the field, dates and types of fertilizers used, total weight of the product inside the box, etc.

Step 2) When the producer's goods are ready to be transported, she arranges for a transportation vehicle which is equipped with a unique RFID reader, a GPS device and a temperature sensor. Once the boxes are filled with grapes, they are loaded into the truck, so the RFID reader detects their presence. The producer and the transportation employee/driver (transporter A) access the F2F platform web application to initiate and confirm the transfer of responsibility for the detectable boxes **[REQ-F2F-06], [REQ-F2F-07]**. The F2F platform web-application flow guides the user to agree about the status of the product inside each box (e.g., weight of each box, ripening level of the product etc.). Once this has happened, the boxes are sealed.

Step 3) Transporter A drives the vehicle to the Warehouse (WH) **[REQ-F2F-08] [REQ-F2F-09] [CON-F2F-02]**. The temperature sensor inside the truck cabin continuously measures and transfers temperature values to the transportation IoT cloud platform along with the positioning of the truck. Once the vehicle reaches the Warehouse, the transporter A accesses the F2F platform web application to determine the specific boxes that will be delivered to the warehouse. The WH employee also accesses the F2F platform web application to confirm the transaction. While the boxes are stored at the WH, Aberon IoT collects information regarding the location of boxes, the temperature in the storage room and the storage duration. **[REQ-F2F-10]**

Step 4) When one or more boxes should be transferred from the warehouse to the supermarket **[REQ-F2F-11]**, the information of the boxes is updated with the information from the storage conditions at the supermarket and finally, the end user (product consumer) is able to retrieve all this information from the farm to his fork.

Storyline 2: Requirements and constraints arising from the FISHY platform user

Step 1) The FISHY user (F2F platform operator/administrator) signs in the platform, setting his username and password **[REQ-F2F-12]**.

Step 2) He also registers the F2F platform and the sub-systems (e.g., Aberon subsystem or SynField subsystem) that this platform consists of, so that they will be protected by the services offered by the FISHY platform. Moreover, for each sub-system he is capable of assigning responsible people (e.g. identifying their e-mail or some other ID) **[REQ-F2F-13]**.

Step 3) Each FISHY user is capable of configuring the details of the platform he is responsible for **[REQ-F2F-14]** including the types of sensors, the endpoints/APIs to be monitored, the network infrastructure to be monitored, the network service provisioning details, the software and hardware configuration and settings of each IT system, etc.

Step 4) Each FISHY user is allowed to set the appropriate access policies that will provide role-based privileges to each of the parties involved in the supply chain **[REQ-F2F-15] [CON-F2F-03]**. These systems include DLT-based platforms, IoT based platforms, etc. **[REQ-F2F-16]**.

Step 5) Each FISHY user receives through user-friendly interfaces alerts and notifications as well as recommendations about the platform he operates (as well as alerts and notifications about subsystems of the platform he operates) **[REQ-F2F-18]**. Alerts are sent when an attack that cannot be automatically handled by the FISHY platform is detected **[REQ-F2F-19]**. Notifications are sent when an attack has been mitigated by the FISHY platform in an automated manner without human intervention **[REQ-F2F-20]**.

Step 6) When an attack that can be mitigated has been detected, the FISHY platform enforces the mitigation measures **[REQ-F2F-21]** and the corresponding FISHY user (system operator) is notified. On the contrary, recommendations are indicating a state that has been predicted by the FISHY platform mechanisms and the user has to be informed in order to judge its criticality and take the appropriate measures **[REQ-F2F-22]**. In order to cope with different and sometimes conflicting demands of the users, the system is capable of providing dynamic reconfiguration (and lifecycle management, in general) of the functions comprising the network service **[REQ-F2F-23]**.

Step 7) The FISHY user can request and receive the outcomes of the cyber-security monitoring process (from assessment to mitigation) for the platform he is responsible for **[REQ-F2F-24], [REQ-F2F-25]** and also request a certificate for the platform he operates as a whole or for subsystems **[REQ-F2F-26]**, **[REQ-F2F-27]**.

Step 8) The FISHY user can also access the FISHY dashboard and have the results of the monitoring process visualized **[REQ-F2F-28], [REQ-F2F-29]**.

3.1.2 Use case 2: Wood-based Panels Trusted Value-Chain (WBP)

Sonae Arauco is developing a Plant 4.0 strategy that aims at developing a digitally connected and collaborative approach to manufacturing, exchanging data throughout the value-chain (upstream and downstream) in a fast, reliable and secure way.

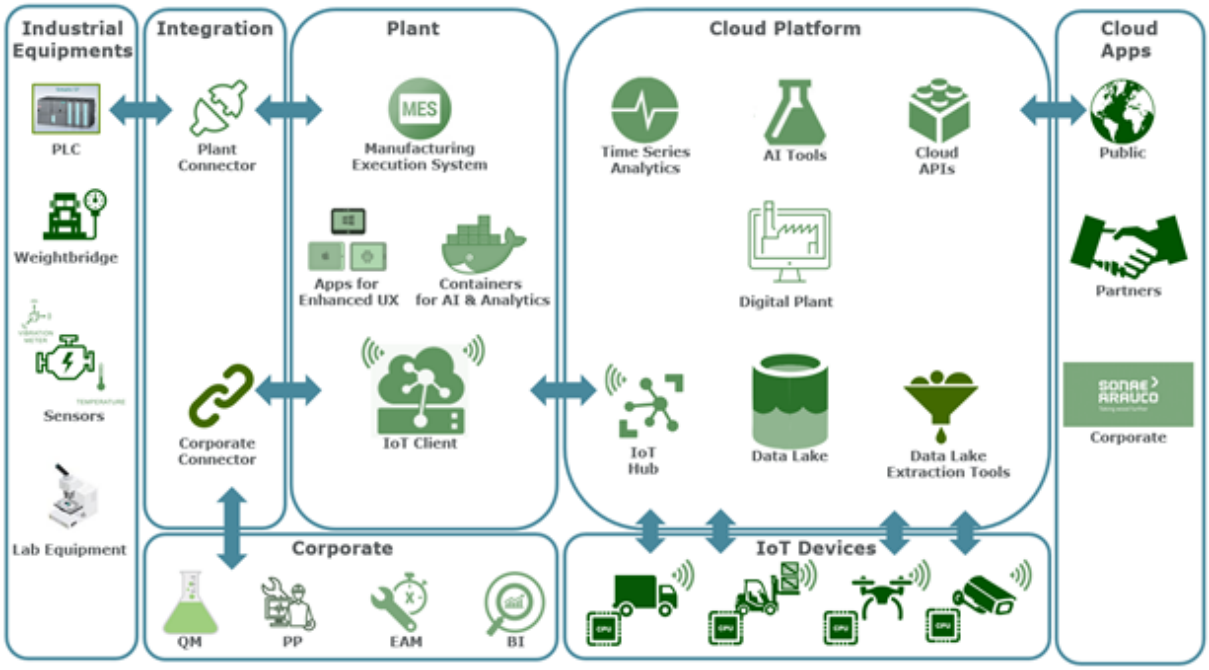

Figure 5: WPB use case architecture

This strategy means bridging the gap between two realities – Information Technology (IT) and Operational Technology (OT) – to ensure security, resilience and availability at all levels. This is achieved through an integration architecture that considers different layers, from the shop floor level to a corporate level (holistic view of different production plants), up to the external layer that enables data sharing and automation in a value-chain perspective (from raw materials suppliers, logistics providers and machinery maintenance companies to industrial clients). Implementing that Plant 4.0 strategy also implies ensuring the connectivity of those machines through sensors and IoT devices to enable data flows at the plant level (manufacturing floor). This was achieved through a Connected Factory architecture, which includes following relevant components:

- SAP ERP Server: Enterprise resource planning (integrated management software of main business processes)
- BigData/VISU Server: On-time Analytics of production figures
- OPC-UA Server: Collects IoT and Industrial automation data (OPC-UA is machine to machine communication protocol for industrial automation)
- SFC Server: Manufacturing Execution System (track and document the transformation of raw materials to finished goods)
- IoT Edge Server: Collects IoT telemetry and send to Cloud (to IoT Hub Server)
- IoT Hub Server: Collects IoT telemetry from all IoT Edge Servers

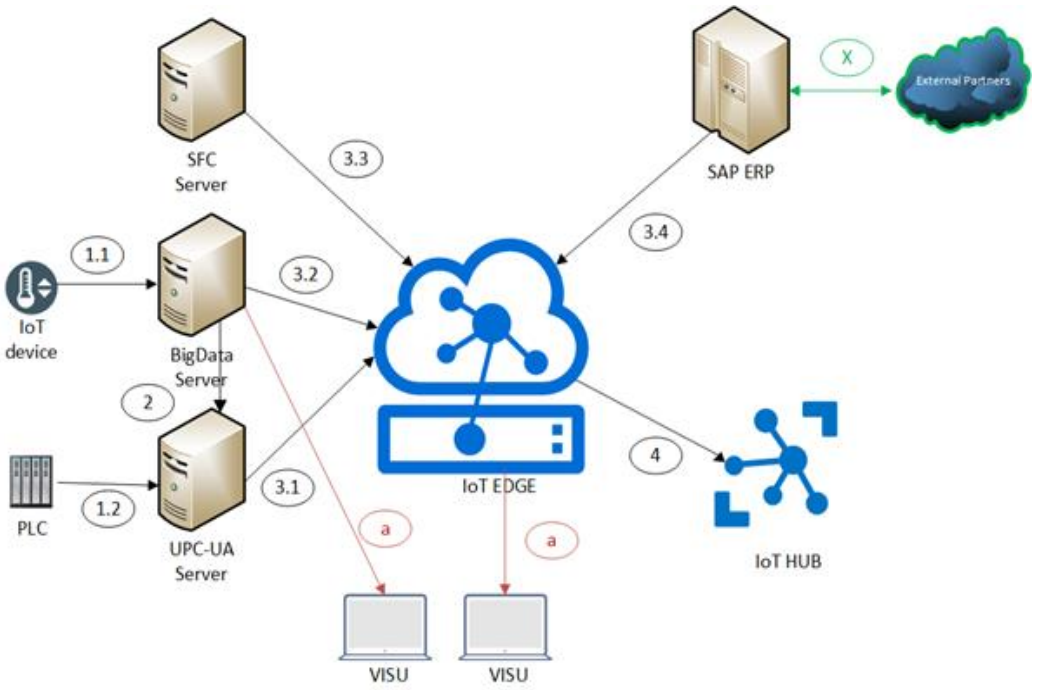

Figure 6: WPB use case data flows

A detailed mapping between the FISHY modules, which will later be described in Section 4, and the WBP data flows stated in the Figure 6 will be reported in D6.1., as a part of the use case 2 description plan for the validation of the FISHY platform IT-1.

3.1.2.1 Mapping of WBP use case to the FISHY reference architecture and platform structure

The envisioned way the FISHY platform components will be used in this validation scenario is show in Figure 7. WBP consists of one organization, Sonae Arauco, and 2 realms, one which corresponds to Information Technology and the other to Operational Technology. As said before, realms can consist of more than one domain, and in this use case we assume OT realm consists of Production and Edge domain, while IT realm consists of Cooperate and Cloud domain.

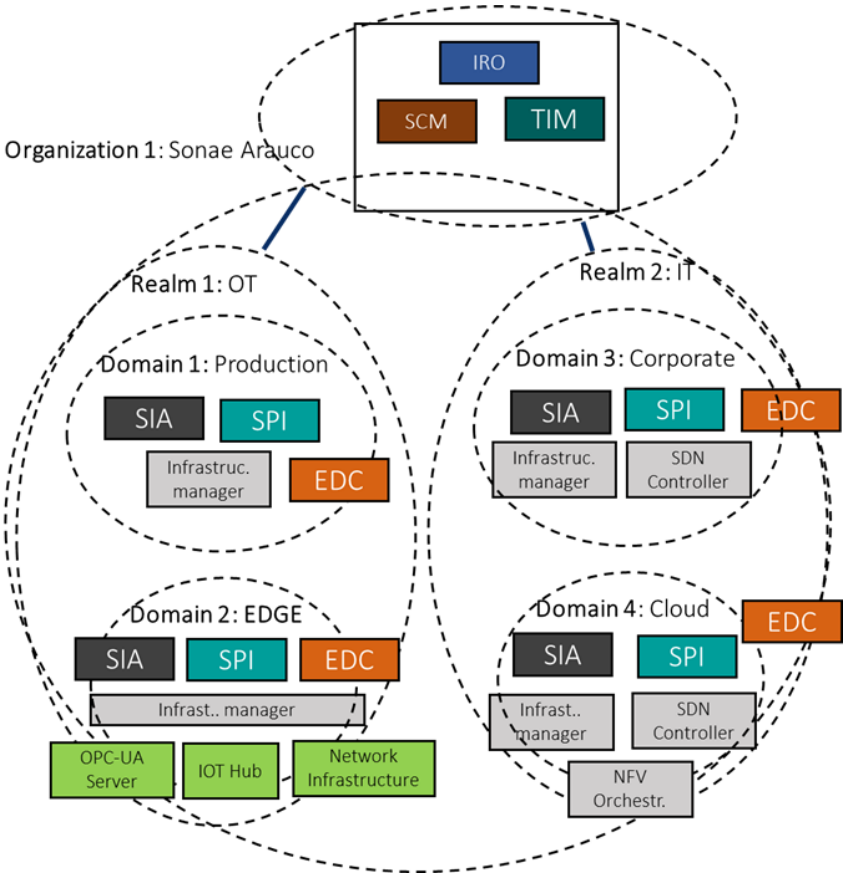

Figure 7: The deployment of FISHY platform components in the WBP use case

Existing IoT devices will be registered on SPI. The Sonae Arauco's "WLAN Controller" will monitor WLAN in real-time and will send information (LOGs) to SIA. If a new device is identified in the WLAN, then EDC will ask to SPI if the device is registered in FISHY. If not, TIM will open an incident to the administrator that will validate if the device is authorized or not. So, if the IoT device is correctly registered the logs will be collected by SIA, processed by SPI and verified if everything is correct using SCM. Finally, a status report will be generated and made available to the administrator through IRO. The logs will be collected (SIA), processed (SPI) and an event will be created (TIM) that will be sent to the administrator (IRO).

TIM will also be used to perform vulnerability scans to devices, classifying vulnerabilities (Risk based) and reporting on them. Moreover, FISHY will send alarms through EDC and/or open incidents through TIM if certain actions are not completed within the established rules.

Additionally, the IoT Hub collects the volume of telemetry (metrics) sent from Edge (OPC-UA server). SIA will be used to read telemetry from the IoT platform. SCM will verify if the volume of telemetry is lower than the minimum historic and, if yes, TIM will analyze the impact, open an incident and suggest to the administrator the mitigation plan.

3.1.2.2 WBP use case constraints and requirements

Storyline 1

Step 1) The process engineer identifies that it is necessary to collect additional data from the production line for higher efficiency or to improve the ML module. The data collected will have two purposes: to provide real-time values **[REQ-WBP-01]** to the plant operators and to be used by Azure Machine Learning module **[REQ-WBP-02], [REQ-WBP-03]** to obtain predictive insights.

Step 2) Plant Maintenance technician installs the new IoT device locally.

Step 3) Plant IT manager configures the IoT device **[REQ-WBP-04] [REQ-WBP-05] [CON-WBP-01].** IT handles all actions in Connected Factory so that the data captured is then used in the pre-defined purpose **[CON-WBP-02].**

Step 4) FISHY platform user registers information on the newly installed IoT devices (ex: Location, Tag ID, Criticality, SLA,…) **[REQ-WBP-06].**

Storyline 2

Step 1) A security incident is detected in one of the components of the IoT platform **[REQ-WBP-07].**

Step 2) TIM module of the FISHY platform performs the analysis of the impact that the incident may have on the organization **[REQ-WBP-08]** and will give instructions to the "incident team" on what actions are necessary to resolve or mitigate the incident effectively **[REQ-WBP-09]**.

Storyline 3

Step 1) A security vulnerability is detected in one of the components of the IoT platform **[REQ-WBP-10].**

Step 2) FISHY platform will classify the risk of the vulnerability **[REQ-WBP-11]** and suggest a mitigation plan **[REQ-WBP-12].**

3.1.3 Use case 3: Secure Autonomous Driving function at the Edge (SADE)

SADE pilot tries to safely manage the software of the sensors embedded within intelligent vehicles as well as the sensitive data that will allow powering on the vehicle, if the driver is authorized. The SADE use case aims to establish and demonstrate FISHY's capabilities as a platform in a kind of special supply chain where the intelligent vehicle is hosting the supply chain itself. The difficulty appears when the vehicle leaves the dealer and the software that the sensors which are embedded in the vehicle contain is no longer safe. This situation is very complex for car manufacturers because they have to control which vehicles, which were sold in different locations, become unsafe. To solve these problems, the FISHY Platform will receive information from the different IoT devices of the different vehicles, allowing it to be processed and to check if the received software is verified as safe by the manufacturers of those sensors. In case of detecting any discrepancy, policies will be launched to solve this situation, alerting users if necessary.

3.1.3.1 Mapping of SADE use case to the FISHY reference architecture and platform structure

The vehicles are distributed by geography connected to different MECs (Multi-Access EDGE Computing) and domains as shown in the Figure 8. Use case 3 benefits from the architecture of the FISHY platform in such a way that the federation allows to know where each vehicle is located. MEC is at the edge of the network, so it benefits from the reduction of latency and access to the data. Thanks to the deployment of the SIA, we will be able to carry out operations and visualize vehicle data risk safely.

The logic of the data collection flow will depend on the use case implementation details, but in general it can be abstracted as follows:

SIA will provide access methods to the information of the vehicles available in the domain. These methods will be both updating data on the vehicles and obtaining information from them, such as the software versions of the IoT devices installed in each vehicle in the domain. SPI defines the security space that will allow granting access to the resources or information available in the infrastructure, denying the request if it turns out unauthorized for the user who is accessing the resource.

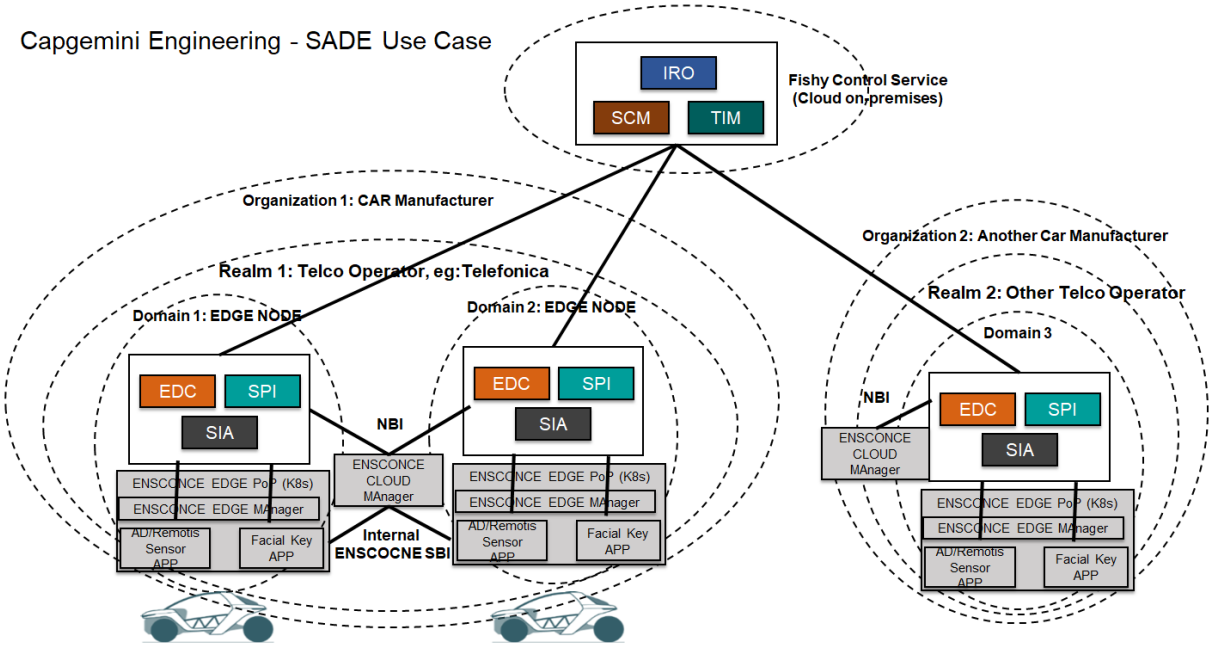

Figure 8: The deployment of FISHY platform components in the SADE use case

On the other hand, both TIM and SCM will be used as an automatic certification measure by monitoring vehicles, allowing the detection of attackers who make changes to the firmware of the devices or detect when a version has been compromised.

IRO module will allow us to have a vision of what is happening in the vehicle infrastructure regarding the risk of the fleet through a web interface. Besides, IRO provides a visual way to introduce the necessary data to carry out the management of the software (Vehicles to be monitored, software certified by the manufacturers, users with sensitive information such as their personal data or their facial identification through photos).

Finally, EDC will be in charge of applying specific policies when a vulnerability defined by the administrator is detected, sending a request to the infrastructure containing what is necessary to mitigate the detected risk.

3.1.3.2 SADE use case constraints and requirements

Storyline 1: Requirements and constraints arising from the SADE platform user

Step 1) The FISHY users (SADE platform operator/administrators/users) sign in the platform, setting his username and password and contact information **[REQ-SADE-01]**. These users could be:

- Administrator
- Car manufacturer
- Local Dealers
- Local Operator
- Owner of a car (who will also be a FISHY user; this is necessary so as to link the account user entity with registered vehicles, to allow information about the risk of his vehicle or allow to power ON the car.)

Step 2) Administrator will allow assign rights to the different accounts [REQ-SADE-13]:

- Car manufacturers are able to register new vehicles and assign local dealers **[REQ-SADE-04]** **[REQ-SADE-12]**
- Local dealers are able to assign a vehicle to a specific owner. **[REQ-SADE-12]**, **[REQ-SADE-11]**

Step 3) Each FISHY user is capable of configuring the details of the platform he is responsible for**.** Local dealers only can modify their assigned vehicles by the car manufacturer and cannot manage vehicles of other dealers **[CON-SADE-03]**. Local dealers will create a biometric model for the car's owner by uploading some images of the face. **[REQ-SADE-09]**

The owner of a car can list his vehicles and private data (Allowed drivers, Plate, email, etc.) associated to him **[REQ-SADE-11]**. He can allow other users to power on his car as well, those users must be previously registered in the FISHY platform and local dealers had to create a biometric model for them. Local dealers can add allowed drivers to their sold vehicles. **[REQ-SADE-12]**

Step 4) Each FISHY user is allowed to set the appropriate access policies that will role-based, providing privileges to each of the parties involved in the supply chain **[REQ-SADE-13]**

Step 5) Administrators define policies to apply measures to vehicles or EDGE infrastructure when defined conditions are met. **[REQ-SADE-14], [REQ-SADE-17]**

- Update firmware of a sensor if this info is available
- Send information email to the user related.
- Send email to the user related to recall vehicle to the dealer in order to perform security updates.
- Manage Car's application deployment to remove the instance or move it to another EDGE. **[CON-SADE-02]**.

Step 6) Car manufacturers must define safe IoT software versions allowed in their vehicles. They can also set a URL to a Safe file repository in order to perform Updates Over-The-Air. **[REQ-SADE-05] [REQ-SADE-06]**

Step 7) Registered and connected vehicles will publish information about software versions installed to each IoT device installed in the car **[REQ-SADE-03].** This information will be stored as metrics in a platform like Prometheus in a service deployed in the EDGE platform to be recovered from FISHY platform.

Step 8) Each FISHY user receives through a user-friendly interface alerts or notifications **[REQ-SADE-15]**.

- FISHY platform must alert when some certification mismatching is detected **[REQ-SADE-08]**.
- FISHY platform must notify whenever any policy is triggered.
- FISHY platform must alert when a human tries to power ON the vehicle and face recognition fails. **[REQ-SADE-10] [REQ-SADE-03]**.
- FISHY platform must alert when private data or system is compromised.

Step 9) When some discrepancy is detected, the FISHY platform applies measures without human intervention, following steps defined in the policy related **[REQ-SADE-16]** and notifying corresponding FISHY users.

Actions performed will be executed against a REST service deployed in the EDGE platform, executing those actions in a compatible way using ROS2 communications with the car. **[CON-SADE-01]**.

Step 10) The FISHY users can also access the FISHY dashboard and have the results of the monitoring process visualized (Software inventory of his vehicles) **[REQ-SADE-11]**. Filtering is mandatory and users could check information about sensors in the vehicles **[REQ-SADE-07]**. Administrators/Car Manufacturers could also filter information by location **[REQ-SADE-02].**

3.2 Functional constraints and requirements

The following table includes all the functional requirements of the three use cases.

Table 1: List of functional requirements

REQ ID	Name	Description	Priority	Component
REQ-F2F-01	Multi-device and multi-system protection	The FISHY platform must monitor the connectivity and security of multiple IT systems comprising of tens of sensors	MUST	SIA
REQ-F2F-02	Access to authentication and authorization events	The FISHY platform must be able to access information from the user authentication and authorization component of the infrastructure	MUST	SIA
REQ-F2F-03	Unauthorised device attempt detection	The FISHY platform must be able to detect the event where a device from an unauthorized platform attempts to enter information in the F2F solution	MUST	IRO
REQ-F2F-04	Network performance monitoring	The FISHY platform must ensure efficient monitoring mechanisms to timely identify network performance degradation	MUST	EDC
REQ-F2F-05	Surveillance of all nodes registering information	The FISHY platform must surveil all entities registering information in the databases (such as the consortium ledger)	MUST	SIA
REQ-F2F-06	Authentication mechanisms	The FISHY platform must be able to supervise different authentication and authorization mechanisms such as OAuth2.0, DIDs, digital signatures	MUST	SIA
REQ-F2F-07	Threat detection	The FISHY platform must be able to detect potential threats from external entities with respect to the above authentication and authorization mechanisms	MUST	SIA

REQ-F2F-08	Security auditing	The FISHY platform must be able to audit and certify the level of security provided by the user equipment and the gateways with respect to missing software updates, vulnerable APIs etc.	MUST	SIA
REQ-F2F-09	Data integrity	The FISHY platform must be able to assess the integrity of the exchanged information e.g. through digital signatures or hash	MUST	EDC
REQ-F2F-12	FISHY user authentication	The FISHY platform must support strong user authentication and authorization mechanisms.	MUST	Identity management (SPI)
REQ-F2F-13	FISHY user capabilities - 1	The FISHY platform must support the FISHY user in defining sub-systems of the platform they operate.	MUST	Identity management (SPI)
REQ-F2F-14	FISHY user capabilities- 2	The FISHY platform must support each user to configure the details of the system to be monitored/ assessed and analysed including also security metrics by the FISHY platform.	MUST	Security metrics (SCM), IRO
REQ-F2F-15	FISHY user roles	The FISHY platform must support role-based access management to support different levels of privileges for the supply chain actors	MUST	Access Policy
REQ-F2F-16	Policy configuration	The FISHY platform must allow the individual infrastructure operators (SynField operator, ABERON operator, user application operator) of the supply chain to set network forward graph policies	MUST	IRO, EDC
REQ-F2F-19	Notification/ recommendation provisioning	The FISHY platform notifies/ alerts/ recommends (events or predictions) the user about attacks and reconfiguration of the platform he operates and its subsystems.	MUST	IRO

REQ-F2F-20	Alert provisioning	The FISHY platform alerts the user when an attack that cannot be automatically handled by the FISHY platform is detected (so that he takes action)	MUST	TIM, IRO
REQ-F2F-21	Reconfiguration notification	The FISHY platform notifies the user when an attack that caused a reconfiguration/ mitigation measure was detected	MUST	TIM, IRO
REQ-F2F-22	Attack detection	The FISHY platform should recommend (inform) the user when a state that has been predicted by the FISHY platform mechanisms to lead to an attack	SHOULD	Incident manager (SPI), IRO
REQ-F2F-23	Network reconfiguration	The FISHY platform must be able to enforce the network reconfiguration of the infrastructure in case of a threat detection	MUST	EDC
REQ-F2F-24	Security reporting	The FISHY platform must allow the user to obtain the results of the cyber security monitoring process of the FISHY platform	MUST	TIM
REQ-F2F-25	Reporting per subsystem	The FISHY platform must offer the ability to the user to request audit per subsystem or system	SHOULD	SCM
REQ-F2F-26	Certificate provisioning	The FISHY platform must provide (upon request) the user with certificates of the platform he operates (certificate issuing)	MUST	SCM
REQ-F2F-27	Certificate provisioning per subsystems	The FISHY platform should provide (upon request) the user with certificates of the sub-systems of the platform he operates (certificate validation)	SHOULD	SCM
REQ-F2F-28	Security results presentation	The FISHY platform could offer visualized view of the results of the monitoring process to the FISHY user	COULD	IRO (Dashboard)

REQ-F2F-29	Dissemination of new attacks information	The FISHY platform could disseminate the detected threats or attacks to FISHY users when deemed relevant (e.g. similar platforms)	COULD	TIM, IRO
REQ-WBP-01	New devices: real-time monitoring	The FISHY platform will detect and alert whenever the information is not collected in real-time	MUST	SIA
REQ-WBP-02	New devices: destination	The FISHY platform will alert whenever the collected information does not arrive at the destination.	MUST	TIM
REQ-WBP-03	New devices: access management	The FISHY platform must ensure that the information only be used/accessed by those authorized.	MUST	SPI
REQ-WBP-04	New devices: detection of new devices	The FISHY platform must identify and alert the existence of new IoT devices/sensors. Operator ACK: - If an authorized device, add to the database. - If not authorized device, open incident.	MUST	TIM, SCM
REQ-WBP-05	New devices: configuration	The FISHY platform will enforce that the new devices are properly configured	MUST	SCM
REQ-WBP-06	New devices: connectivity and security monitoring	The FISHY platform must monitor the connectivity and security of multiple IoT devices/sensors	MUST	TIM, SPI, SIA
REQ-WBP-07	Security incidents: detection	The FISHY platform must be able to detect security incidents in components of IoT platforms	MUST	TIM
REQ- WBP-08	Security incidents: impact analysis	The FISHY platform must analyze the impact that an incident may have on an organization	MUST	TIM, SCM
REQ- WBP-09	Security incidents: recommendations for mitigation / resolution	The FISHY platform must recommend needed actions to "incident teams" on what is necessary to resolve or mitigate an incident effectively	MUST	TIM

REQ- WBP-10	Vulnerability detection: assessment	The FISHY platform must assess vulnerabilities in IoT platforms	MUST	TIM
REQ- WBP-11	Vulnerability detection: risk classification	The FISHY platform MUST classify the risk of a detected vulnerability	MUST	SCM, TIM
REQ-WBP-12	Vulnerability detection: mitigation plan	When a vulnerability is detected, the FISHY platform MUST suggest a mitigation plan	MUST	EDC, TIM IRO
REQ-SADE-01	Users Sign-up	The FISHY platform must provide a way to register users.	MUST	Identity manage ment (SPI), IRO
REQ-SADE-02	Geographic dispersion support	The FISHY system must take into consideration geographic dispersion of the supply chain entities	MUST	IRO, EDC
REQ-SADE-03	System access from top and bottom	The FISHY platform must be able to access information from the user side and vehicle side.	MUST	EDC, SIA
REQ-SADE-04	Vehicle registration	The FISHY platform must provide a way to register new vehicles.	MUST	Dashboa rd, SPI
REQ-SADE-05	Add certified IOT software versions	The FISHY platform must provide a way to manage and register lists of certified software versions and keep it securely saved. Should contain version, manufacturer and model. Optionally, checksum or link to a safe storage with the update file.	MUST	Dashboa rd, SCM
REQ-SADE-06	Revoke certified IOT software versions	The FISHY platform must provide a way for SW administrators to revoke or update specific versions from the certified list (or allow FISHY to do that automatically).	MUST	Dashboa rd, SCM

REQ-SADE-07	Filtered search	The FISHY platform must provide a way to filter lists of certified software versions by Sensor, Car, Vendor, Country, etc.	MUST	Dashboard, SPI
REQ-SADE-08	IOT Software Version monitoring	The FISHY platform must be able to audit and certify that the level of software patches of each IOT device in every vehicle is aligned to security versions provided by manufacturers. The platform must monitor and take metrics from the vehicles to check this certification.	MUST	Security metrics (SCM), SIA
REQ-SADE-09	Sensitive data secure storage	The FISHY platform must provide a proper mechanism to save biometric data (from images) and sensitive data about users in a secure manner.	MUST	Dashboard, SPI
REQ-SADE-10	Sensitive data secure access	The FISHY platform must allow check sensitive data by the system in order to perform biometric checks.	MUST	SIA, SCM
REQ-SADE-11	Vehicle configuration for car owners	The FISHY platform must support each owner user to configure and see information about his owned vehicles.	MUST	Identity management (SPI)
REQ-SADE-12	Vehicle configuration for privileged users	The FISHY platform must support each dealer/car manufacturer user to configure their vehicles	MUST	Identity management (SPI)
REQ-SADE-13	Role model for users	The FISHY platform must support role-based access management to support different levels of privileges for actors	MUST	Access Policy (SPI)
REQ-SADE-14	Policies definition	The FISHY platform must provide a way to define policies and actions to be performed when some conditions are taken.	MUST	TIM, IRO
REQ-SADE-15	Notifications about actions	The FISHY platform notifies/ alerts the users about policies triggered to vehicles or EDGE infrastructure.	MUST	TIM, IRO

REQ-SADE-16	Policies enforcement into elements	The FISHY platform must be able to enforce policies into the isolated devices and to group elements: Sensor, Car, Vendor, Country, etc.	MUST	SIA, SPI
REQ-SADE-17	Allow several kind policies	The system should include policies that will not only block certain traffic, or users from the car itself but eventually will ensure that only selected encryption mechanisms are used, or algorithms are updated.	MUST	SIA, SPI

Table 2: List of functional constraints

CON ID	Name	Description	Component
CON-SADE-02	App provisioning in the EDGE	The FISHY platform Upon integration with EDGE Computing platforms systems has to be able to trigger the provisioning of APP	SIA
CON-WBP-O1	IoT sensors registration	IoT Sensors are not registered in the IoT platform but on UPC-UA server.	SIA
CON-WBP-O2	Connected Factory security design	Existing Connected Factory architecture did not address security in its design	SIA

3.3 Non-functional constraints and requirements

Table 3: List of non-functional requirements

REQ ID	Name	Description	Priority	Component
REQ-F2F-10	Extension/ Expansion scalability	The FISHY platform must support service /mobility. For example, if a new IT system is connected to an existing supply chain, the FISHY platform must be able to handle this as a whole	SHOULD	IRO, audit certificate
REQ-F2F-11	Geographic dispersion support	The FISHY system must take into consideration geographic dispersion of the supply chain entities	MUST	SIA
REQ-F2F-18	User friendliness	The FISHY platform must offer intuitive user-friendly interfaces.	SHOULD	Dashboard
REQ-SADE-18	Trustworthy mechanisms and collaboration among stakeholders	The system must support trustworthy mechanisms and facilitate collaboration among stakeholders based on trust and evidence comprising the supply chain	MUST	SIA

Table 4: List of non-functional constraints

CON ID	Name	Description	Component
CON-F2F-01	Lightweight data collection	The FISHY platform must include lightweight software blocks for data collection	EDC
CON-F2F-02	Support of software Virtualisation technologies	The FISHY platform should be based on containerized software blocks in order to support service mobility in a timely manner	SPI
CON-F2F-03	Compliance to legal legislation	The FISHY system must comply with the legal framework with respect to information dissemination	SIA
CON-SADE-01	Consider constraints of communication type	The system must take into consideration the constraints inherited from the ROS2 protocol used in the REMOTIS CAR as communications framework	SIA
CON-SADE-03	Compliance to legal legislation	The FISHY system must comply with the legal framework with respect to information dissemination	SIA, EDC

4 Architectural Design

In this section, we will describe an overview of the FISHY main modules, shown in reference architectural design for IT-1, shown in Figure 1. The initial designs of the FISHY Trust Manager (including its two main modules Trust & Incident Manager and the Security & Privacy Data Space Infrastructure) and Security & Certification Manager (including its two main modules Security Assurance & Certification Management and the Enforcement & Dynamic Configuration), with the details on their internal blocks, modules, as well as interaction workflows have been reported in deliverables D3.1 [1] and D4.1 [2], respectively. When designing the FISHY platform numerous technologies and tools for managing security and trust have been leveraged with the current agreement that the set of external tools will be integrated in FISHY. The detailed description of these tools and their utilization in specific components have also been described in said deliverables.

4.1 Main architectural building blocks

4.1.1 Trust & Incident Management

TIM is the module responsible for security assessment of stakeholder's devices, components, and infrastructures. It consists of components for anomaly detection and assessment of vulnerabilities and incidents and components for providing mitigation response activities (including Vulnerability Assessment, Incident Detection, Impact Assessment, Trust Monitor, Threat/Attack Repository Mitigation).

Vulnerability Assessment

The functionalities of this module cover three important sub-processes: 1. Determining and establishing assets on the infrastructure; 2. Determining, naming, and prioritizing the vulnerabilities found in the analysed system, component, or environment and 3. Proposing the most effective mitigation actions.

Vulnerability assessment will be in charge of providing the insight of how the detected vulnerabilities may entail a risk and understanding the degree of weakness the monitored infrastructure may present. Applying this to the FISHY supply chain platform will allow for the supply chains to be more resilient to threats and, more specifically, to vulnerabilities. To perform a vulnerability assessment, it is important to have an accurate idea of how every piece fits in the infrastructure and what flaws could this piece have. Besides, the fact that several IoT or other insecure devices can connect to the supply chain platform make this module essential, especially if we understand what vulnerabilities these devices may have.

Although there are various kinds of vulnerability assessments (performed on network, host, database, applications etc.) from the FISHY perspective, an assessment of the monitored ICT platform of the supply chain would make more sense, given that supply chain platforms usually are made up of various components. It could also be appropriate to assess IoT devices if they are going to take part in the ICT infrastructure of the supply chain. The functionalities of Vulnerability Assessment module will be complemented with the help of a risk assessment module to identify information at risk, concerned components and potential damage the platform may suffer. To this end, a Risk Assessment Engine (RAE) tool will be used (described in Section 4.2.1 of D3.1 [1]).

Incident Detection

As important as vulnerability, risk and impact assessment are in preventing security breaches, incident detection is just as important as the first step of a recovery process if or when a breach does occur. In FISHY, the main role of this module will be maintaining pace with the threats and attacks, that are

becoming increasingly more sophisticated and harder to detect, and as traditional tools, (such as antivirus and anti-malware software) are unable to sufficiently guard a company or organizations from unwanted access. To design an efficient process of intrusion detection, appropriate data sources have to be made available, along with the capability to parse, filter, and analyse the incoming information.

FISHY plans to integrate incident detection into a holistic process of cybersecurity hardening, increasing resilience and enabling faster response time to incidents over the whole ICT infrastructure of a supply chain by leveraging existing open-source technologies, such as Wazuh [3], and integrating and expanding the capabilities of components developed within the framework of European research projects, for example, XL-SIEM (Cross-Layer Security Information and Event Management), an event management tool oriented to enhancing normal SIEM capabilities (described in Section 4.2.2 of D3.1 [1]).

Impact Assessment

One of the most accurate ways to determine and try to foresee effects of changes in the system is through impact assessment. In FISHY, the functionality of this module is oriented to defining and outlining the existent relation between status of the system and the changes happening, involving the employment of both **qualitative and quantitative data**, which are normally expected to be faced to various indicators within the assessed item.

Regarding the FISHY project, Impact Assessment module will help in determining how and to what extent the supply chain will be affected should a change happen in the overall platform. In fact, the Trust Incident Manager and, by extension, the Trust Manager can significantly benefit from the results of the impact assessment to make the ICT platform of the supply chain more resilient as far as issues of any kind are concerned. So, the main idea behind this module is its role in clarifying how FISHY could help the monitored supply chain to improve and benefit and provide an answer to following questions:

- What changes can affect the supply chain infrastructure?
- Are these changes going to influence positively or negatively?
- What is the expected extent of the changes in the supply chain platform?
- Is there a way for the changes to be intended? Or are they totally or partially unexpected?
- Would it be possible for the supply chain infrastructure to improve, strengthen and be more resilient after suffering the consequences of these changes?

The functionality of performing the assessment within this module will be guided and assisted by cybersecurity tools such as the Risk Assessment Engine (RAE), as they can enhance the results in terms of accuracy, saving of time and reliability. Although the tool is mainly devoted to the risk assessment process, the impact assessment benefits from the outline of the risks and threats that the supply chain platform faces, including the analysis of how the changes can impact the supply chain. The main outcome of the impact assessment will help in defining how risks and changes can influence confidentiality, integrity, availability (CIA) and even traceability of the information. Besides, it is possible to categorize the impact in various categories, such as: financial, commercial (or reputational) and legal and regulatory impact.

Mitigation

The Mitigation module of Trust and Incident Manager focuses on detecting anomalies from network/IoT data based on Machine Learning algorithms. Here, mitigation mechanisms based on ML algorithms will work in two different ways: online mode and offline mode.

Offline mode: A Database/Repository of the net entries is read and based on these data, an expected behaviour pattern will be established in order to protect the devices. The models used for detecting anomalies and categorizing will be adjusted offline and once they have been trained, the online mode can be used.

Online mode: A user interface will monitor system data in order to detect attacks. This interface will then select and use fitted model that has been trained offline.

Threat/Attack Repository

The Threat/Attack Repository module will serve as the information storage for the TIM component of the FISHY platform. While having its own structure and operations, the data contained here will be made available for the rest of the components of FISHY. It is key for TIM to have a place where to gather important information for various purposes including:

- Harnessing of the information by the other TIM components.
- Gathering and collection of evidence.
- Storage of information to be further analysed and/or processed. This could include very different activities such as auditing, monitoring, and counterintelligence.
- Central repository of information for TIM, especially of those resulting from the activities performed by the TIM modules.

In addition, having a place where information produced is gathered and centralized, will allow TIM to benefit from Threat/Attack Repository module through:

- Increased performance based on the ability of collecting and retrieving centralized information quickly and easily.
- Improved results:
  - Data is available for analysis when requested.
  - Outcomes of the tasks performed are centralized and can be correlated (if needed).
- TIM can deliver better outcomes to the FISHY platform (and, therefore, it can be reconfigured promptly) based on the event and threat information hosted in the database.

The threat/attack repository will be available for all the components of TIM with the aim of easing the use of the information stored, while a pub/sub layer will allow components to subscribe to relevant data sources and receive real-time updates when new data is available for processing, or analysis results that require further action, such as enacting new policies on the supply chain ICT.

Smart Contracts

The Smart Contract module receives information from a distributed application (DAPP) as happens in all blockchain based solutions, executes the coded logic and inserts in the blockchain a specific "transaction" which can be a decision or an event depending on the logic implemented in the smart contract. In FISHY, as detailed in D3.1 [1], the code of the smart contract will include functions targeting the detection of security-relevant events (for example, persistent hit from a certain IP to a node of the F2F supply chain IT system) as well as functions registering in the blockchain the decisions made by FISHY platform (e.g. IP ban). Different functions per use case can be implemented in principle, although in the piloting activities we will focus on the F2F use case first. Having the decisions made and registered in the blockchain ensures that the decision issuer cannot be disputed and the functionality (implemented logic in the smart contract) can be modified only by the blockchain administrator.

4.1.2 Security & Privacy Data Space Infrastructure

The Security & Privacy Data Space Infrastructure (SPI) component consists of three modules, Identity Management, Access Policy and Data management and is responsible for the implementation of system requirements related to:

- Identity Management (IdM)
- Access Control (AC)

- Privacy enforcement
- (Low-level raw) Data Management

Here, IdM and AC are related security functions. By definition, AC includes Authentication, Authorization, and Auditing. Whenever a subject (user or process) wants to access an object (any accessible system resource), the AC assures the subject is legitimate and has the requested access rights before granting it access. Furthermore, AC must also log all the access operations allowing posterior auditing operations. This way, AC is a first-line defense mechanism and one of the most important preventive security controls [4].

Within the AC function, Authentication is the operation responsible for checking if the attributes (credentials) presented by a requesting subject ID correspond to those previously stored, during registration, with novel architectural solutions, such as the one contemplated for FISHY, pushing this authentication function to a dedicated server for identity management, in particular when dealing with users. The server in question will be capable of aggregating different sets of attributes for different application environments and providing a federated single sign-on solution when authentication delegation is desirable, for performance and efficiency reasons [5], [6].

Identity Manager

For Identity Manager module we analysed several technological solutions available today, which support both IdM and AC, deciding on the implementation of the OpenID Connect standard [7] as it is frequently pointed as one of the best. It implements a flexible identification layer on top of OAuth 2.0 protocol, which is also a well-known robust token-based protocol for authorization, fitting the needs of all distributed frameworks [7][8]. There are several OpenID Connect implementations and for FISHY's Identity Manager module we propose to use KeyCloak, an open source solution with a large supporting community and several success use cases already documented [9], as it fulfills all system requirements related to IdM and AC, listed before.

Dividing all FISHY components (or modules) into two classes, Data Consumption (DC) and Data Provider (DP), the IdM main function will be to:

1. Receive a request from DC
2. Authenticate the request (both through a previously shared secret key, or through a user authentication process) -- this requires an initial registration operation
3. Check the request against access policies defined
4. If authorized, provide the DC a token
5. The DC sends a data request to the DP passing the token as a parameter
6. The DP checks the token validation (signature validation), and if it is valid returns the data.

Since the token has timing limits it can be re-utilized, meaning steps 1 through 4 are necessary only at the beginning of a data transaction operation. OAuth 2.0 is very flexible concerning token parametrization, allowing to add specific parameters, and also supporting a token re-validation operation. All these details need to be addressed at the AC Policy level, according to specific requirements of each module/component.

Concerning user authentication, this solution will rely on OpenID protocol [10], a well-known and universally used mechanism on Internet-based applications. It allows a user to be registered in only one place (OpenID server, or ID Provider, which is also implemented in KeyCloak, in our case). When an application (relying party) previously registered receives a request from a client on behalf of a user, it is automatically redirected to the ID Provider, which performs the user authentication. OpenID does not enforce any specific authentication mechanism, being possible to implement anyone -- the standard only imposes the format of the returned message.

OpenID Connect implementation supports several minor variations of the general protocol description above, both for authentication and authorization. Following the OAuth 2.0 nomenclature, those

variations are designated by flows, and the ones to be used depend on the implementation decisions, which will be addressed in WP3.

Access Policy

Access Policies module will be responsible for preserving user's privacy accessing the data according to specific policies and will be configured and stored in IdM (OpenID Connect). Concerning specification and automatic deployment, KeyCloak does not provide a formal mechanism. An administrator, through a user interface, and using the main concepts related to AC (subject, object, credentials, roles, composite roles, realms, operations, and logging), define all the Access Policies required. However, since there is an API that gives access to the core policy engine and storage, it is possible to deploy automatic mechanisms developing the required interfaces. This way, and since the main concepts are fully supported, it is even possible to develop an interface to import (and export) policies, e.g., in EXtensible Access Control Markup Language (XACML) format - which may be relevant if other FISHY components need to use that format.

At this stage, there is no proposal for an autonomous component to manage Access Policies. However, if along the development process that module becomes necessary, it will be easy to incorporate with a dedicated interface.

Data Management

This module's main task is to assure that the data is formatted and categorised in the correct way, according to system requirements. It consists of two main functionalities: adaptation and anonymization. The adaptation process will be responsible for mapping the raw data received through the infrastructure abstractions and the edge elements onto data structures, include all necessary attributes to support functional and non-functional system requirements regarding data formatting, filtering and aggregation. On the other hand, the anonymization process will focus on designing suitable anonymization techniques for FISHY, which will ensure appropriate privacy and anonymization levels of the data shared among different organizations or entities. More details on these two submodules can be found in Section 3.1.4 of D3.1 [1].

4.1.3 Security Assurance and Certification Management

The Security Assurance & Certification Management (SACM) is responsible for monitoring, testing and assessing critical components and processes of the ICT infrastructure under the scope of the FISHY project. The security assurance and certification management module is engaged in four important submodules: Security Metrics, Audit, Evidence Collection Engine and Certification management.

Security Metric

This module contains a set of tailored measurable metrics, with respect to the needs of FISHY pilots. These metrics, expressed in an XML format, are selected for measurement and evaluation by the administrator and through the IRO dashboard. These XML files of the latter metrics contain information regarding the type of the metric, the period of the evaluation and the type of the target asset that the security metric is referred to.

Audit

The Audit module will be responsible for initiating, coordinating, and reporting to the IRO dashboard the auditing process results. Based on a monitoring framework called EVEREST (EVEnt REaSoning Toolkit), the audit sub-module consists of two components, the main auditor and an audit-manager orchestrator. Taking as input the respectively selected security metric from the previous sub-module, it translates/maps the latter to an event reasoning level while the audit orchestrator initiates the corresponding instance for the auditing procedure. Each instance reflects the selected security metric, while it pushes the metric info to the evidence collection sub-module and awaits for its findings.

Evidence Collection Engine

This module is responsible for aggregating the required evidence from multiple sources. It is responsible to collect in real-time, evidence-based information regarding the operational status of ICT infrastructure components along with their related data. Evidence Collection Engine is built around Elasticsearch while all the evidence-based information is represented using the open-source specification of Elastic Common Schema (ECS).

Certification

Last, the Certification management module provides evidence-based security reporting and certification to the needs of different stakeholders ranging from senior management to external auditors and regulators, incorporating different access levels to the respective users. The latter component supports the creation of specialized reports based on the findings of the Audit component while it may inform the IRO component for the former results.

The detailed description of each of these modules has been provided in Section 4 of D4.1 [2].

4.1.4 Enforcement and Dynamic Configuration

The Enforcement and Dynamic Configuration (EDC) is the component ensuring that the **networked** infrastructure protected by the FISHY framework has been correctly configured and that is working as expected. In order to achieve these goals, the EDC will leverage the concept of *policy*, that is a configuration rule for a (physical or virtual) network component written in a formal language, unambiguous and easy to parse for automated tools. From a security point of view, the EDC will handle mostly two types of policies:

- *authorization policies*, that are policies used to configure the security controls allowing or denying a connection (e.g. firewalls and WAFs);
- *channel protection policies*, that are policies used to configure the security controls in charge of protecting a communication session by safeguarding its confidentiality and/or integrity (e.g. VPN terminators).

The EDC consists of three main modules strictly interlaced together: Controller, Planner and Enforcer.

Figure 9 depicts the global work-flow of the EDC and its interactions with other FISHY components.

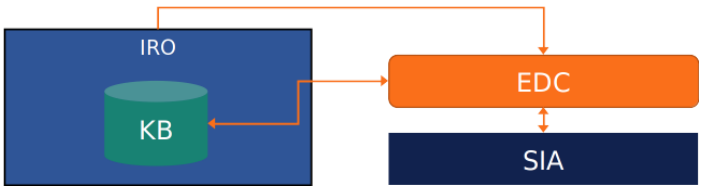

Figure 9: Interactions between IRO, EDC and SIA

The EDC will be initially triggered by IRO (see Section 4.4.5) when new intents or policies are created or the existing ones are updated. These updates can be due to an automatic reaction (e.g. an automatic response to an ongoing attack) or manually by the administrators (via the dashboard). IRO will then compile the intents into a set of high-level policies, and it will store them into the Knowledge Base (KB). The EDC will then react to the Knowledge Base change and will *enforce* the high-level policies by deploying and configuring the appropriate NSFs via the SIA (see Section 4.4.6).

Controller

The Controller module is the first coordination and refinement module of the EDC , whose main job is to perform the initial policy refinement and coordinate the other EDC modules. Its detailed description and general work-flow have been provided in Section 3.4 of D4.1 [2].

When an intent is added or updated, IRO will produce one or more high-level policies that will be stored into the IRO's knowledge base and will call the Controller to start the enforcement process. The Controller will then start to perform several queries in order to gather enough information to generate a set of medium-level policies. It will ask the Knowledge Base the set of high-level policies to analyze, information about the current *landscape* (e.g. the currently deployed security controls) and their medium-level policies. It will also query the Planner about the list of the available NSFs.

Finally, the Controller will refine the high-level policies into medium-level policies that will be stored into the knowledge base. The last step of the process is to contact IRO to signal that the refinement process has been completed.

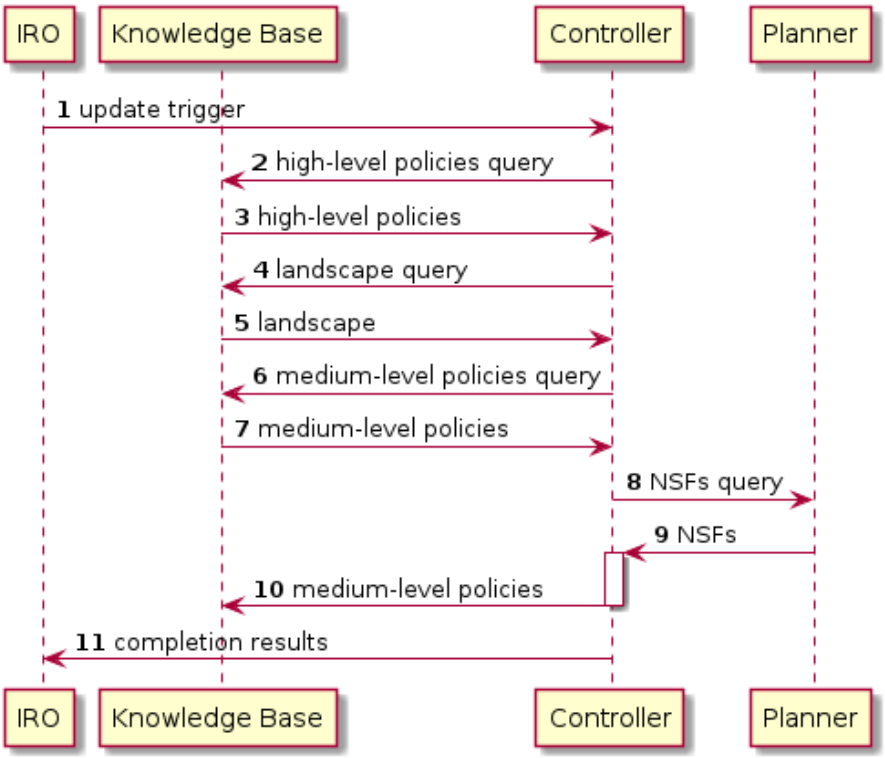

Figure 10: Controller workflow

Planner

The Planner is a dynamic catalogue that stores the list of the NSFs available to the EDC. Through a special API, an NSF can register (or unregister) itself into the Planner. During the NSF registration process, the appliance must also give to the Planner the list of its supported capabilities (e.g. a firewall supporting the rate limiting action or a VPN terminator supporting a specific cryptographic suite). The list of capabilities will allow the Controller first and Enforcer later to perform a suitable generation of policies.

Enforcer

The Enforcer is the last component called in the EDC work-flow. Its main job is to finalize the configuration of the NSFs, as shown in Figure 11.

The Enforcer reacts when triggered by IRO, usually because a new or updated medium-level policy has been pushed into the Knowledge Base by the refinement process of the Controller. When such an event

is detected, the Enforcer will download the medium-level policies and the current landscape information from the Knowledge Base, it will contact the Planner for querying about the capabilities of the involved NSFs and will produce a set of low-level configurations, ready to be ingested by a specific security control (e.g. iptables or strongSwan). Such configurations will be then stored into the Knowledge Base and IRO will be signaled about the completion of the medium-level policies translation process.

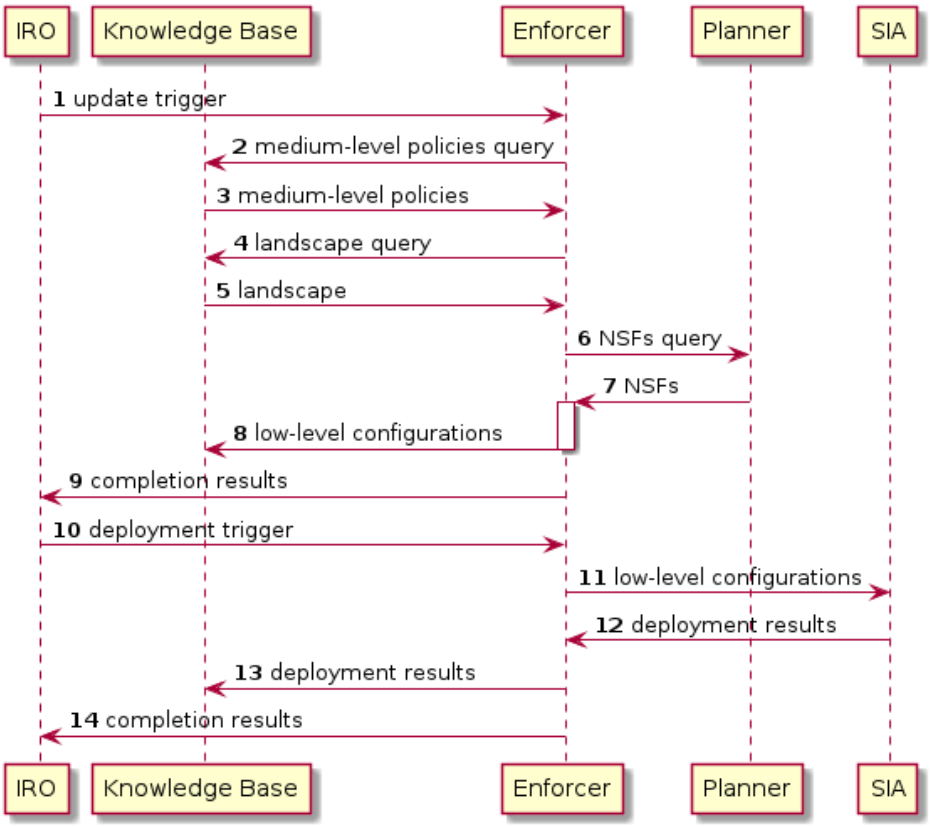

Figure 11: Enforcer workflow

Next, IRO will contact the Enforcer again to start the final deployment of the NSFs. This process might be fully automatic or it can be manually triggered by the administrator only after that the low-level configurations have been properly validated. The deployment process starts with the Enforcer sending the configurations to the SIA via its API (in order to set up the NSFs). The SIA will send back the configuration results that will be stored in the Knowledge Base for future references. Finally, the Enforcer will contact IRO to signal the completion of the deployment process.

4.1.5 Intent-based Resilience Orchestrator and Dashboard

Intent-based Resilience Orchestrator (IRO) component is responsible for mapping high-level intents into configured policies, receiving the intents from users as plain text and then using appropriate ML techniques to translate these user requirements into structured policies compatible with the Enforcer component. It consists of the following sub-modules: Intent Manager, Policy Configurator, Intent Compiler, Learning & Reasoning, Knowledge base and Dashboard.

Intent Manager

Intent Manager parses and stores the high-level input text read from the user in a structured format containing user requirements. This component is also responsible for creating, reading, updating and

deleting intent structure, which contains the necessary information that defines an intent (name, id, description, attributes and constraints).

Policy Configurator

This module will be responsible for matching requirements provided by the Intent Manager with the existing policies stored in the Knowledge base and then configuring the selected policies to be triggered.

Intent Compiler

This module is responsible for verifying the configured policies received from the intent manager, checking the conflicts, and validating its coherence with the requirements. After that it triggers the controller from the EDC component to execute the policies.

Learning & Reasoning

Learning & Reasoning is the component responsible for receiving feedback from other blocks such as TIM, where IRO needs to validate some decision with the intervention of the user. Rules or metrics are received, then alerts and recommendations are sent to the dashboard for validation. Learning & Reasoning uses AI and ML algorithms to analyse the history of executions and decisions, then predict the best decisions to be made and to help the administrator understand what policies to choose.

Knowledge base

Knowledge Base will be a database of knowledge required to manage intents and policies by other IRO components, such as Intent Manager and Learning & Reasoning. Intents can be defined in different JSON format depending on the type of intents, which is related to the requirements and the expected policies to be executed. As the policies are numerous, intent formats can also be numerous and for this reason it is important to store the intent structure, to be extracted from the high-level input text, in the knowledge base. An example of intent that can be defined by an administrator for data anonymization purposes in a high level form (source, destination, level/reason of anonymization,..). Such intent had specific structure and designed for a specific goal, where it will be matched with predefined policies. Policies will also be stored in the knowledge base in order to be used to map intent requirements into applicable configured policies. The previous example of intent will need policies that can ensure the anonymization of data and based on the level/reason of anonymization a policy will be chosen (Pseudonymization, Character Masking,..). Elasticsearch will be used to implement the knowledge base component, where JSON objects will represent intents and policies.

Dashboard

Dashboard (as a web application running in the browser on client side) directly communicates with the back-end (FISHY core services) via HTTP (REST API) and websocket and will serve as the main access point for the FISHY platform, including among other functionalities, authentication and authorization mechanisms for FISHY users. As different users may have distinct roles, this entails various access levels. For instance, some users can only be able to see alerts, events, whereas other user can be allowed to give feedback to the events, such as intent creation.

The Dashboard will be built modularly where each FISHY core component provides its own functionality and integrated into the Dashboard, with the examples bellow provided for tools, which are already agreed to be used as internal part of TIM modules.

XL-SIEM dashboard

The XL-SIEM comes with a GUI aimed at providing data in a user-friendly mode. Amongst the various kinds of data displayed by the GUI we can include:

- Events and alarms: can be set to be released hourly, weekly or even montly
- Charts (to provide overview of the supervised system)
- Graphics
- Reports

- Raw logs to be accessed by the user

The following image shows an example of the top alarms detected by XL-SIEM and displayed to the user by means of the GUI:

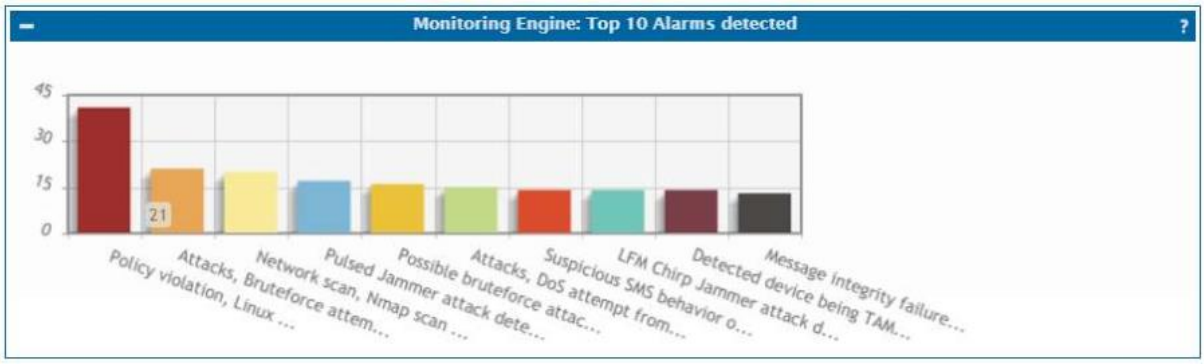

Figure 12: XL-SIEM GUI

In addition, the user interface provides enhanced capabilities including:

- The possibility for the XL-SIEM administrator to perform changes and adjustments on the tool settings
- Provides a totally configurable way of displaying widgets that contain information about security trends, KPIs, etc.
- The XL-SIEM allows the user to produce reports in PDF format, Usually, these PDF reports contain a summary of the analysis performed by the SIEM.
- Information of the EP directives (i.e., correlation rules) that the engine is going to process.

The dashboard is made up of a database and some folders, placed on a docker container.

Regarding the FISHY platform, the dashboard can be coupled with other data presentation tools, always considering the standardization of the data that is to be presented.

Wazuh dashboard

Wazuh provides an interface to the events/alerts triggered by the system using Kibana dashboard. This dashboard can be integrated with the rest of the FISHY dashboard by redirecting the user from the main FISHY IRO dashboard to the dashboard provided by the tool. However, there is a problem with the authentication step that needs to be done after this redirect. For the authentication we will need to use a dedicated user created while creating/deploying the tool in the domain of the client using FISHY Agent.

VAT dashboard

The Vulnerability Assessment Tool (part of TIM) provides a web-based graphical interface, allowing users to see the status of scans, their schedules, past results and reports. While the most relevant scan tasks will be set up by the FISHY Platform, users can take advantage of the VAT webUI to run their own scans on-demand, or even custom scripts written in Bash, Python or Metasploit. Similarly, to other tools that provide their own dashboard, integration of VAT can be achieved with a redirection from the main FISHY user interface.

The integration of other FISHY core components and tools into the Dashboard will be further extended in D5.1, due to be released in M15.

4.1.6 Secure Infrastructure Abstraction

The purpose of the Secure Infrastructure Abstraction (SIA) module is twofold. On the one hand, it implements a data-plane interface that supports secure end-to-end communications across the ICT supply chain (including IoT/Edge/Cloud infrastructures). On the other hand, it provides the point-of-access to interact with the NFVI infrastructure resources offered by FISHY adopters.

The SIA module encompasses the following logical components, which are subsequently described throughout this section:

- The Secure Edge Node (SEN).
- The Network Edge Device (NED).
- The Monitoring and Telemetry component.

The SIA module offers a *northbound interface* to the other blocks and components of the FISHY platform, particularly the Enforcement & Dynamic Configuration and the Trust & Incident Manager. As it is commented in Section 2, the FISHY approach considers every organization to be structured as a set of realms and domains, where domains may deploy a SIA module to incorporate their NFV infrastructure resources to the FISHY platform. In this respect, the SIA northbound interface provides a technology agnostic view of the NFV infrastructure resources available at an organization domain, enabling the management of network services and VNFs by other relevant FISHY entities. The SIA northbound interface is subsequently described in Section 5.

It is important to emphasize that the SIA module is defined as a set of logical components, each providing well-defined and independent functionalities. An actual implementation of the SIA module may opt to deploy these logical components as separate software functions or combine their functionalities at specific locations into a reduced set of software components (e.g., the NED may integrate monitoring and telemetry functionalities). In the following, we describe each of the SIA components in more detail.

The Network Edge Device (NED)

The *Network Edge Device* (NED) is a data-plane element that will be deployed at the network edge of every organization domain. This component will primarily be responsible for a) providing the data-plane interface to support inter-domain communications within the FISHY platform, e.g., between an IoT/edge infrastructure and a cloud infrastructure, or between multiple cloud infrastructures; and b) controlling the network access to the infrastructure resources (IoT/edge/cloud) of every organization domain, protecting data traffic entering and leaving the domain.

The NED component will provide the abstraction of a layer-2 switch, supporting the automated configuration of virtual LANs (VLANs) that span multiple organization domains. This way, the NED will support data-plane communications among VNFs and devices deployed at remote domains (VNFs can simply be attached to the NED upon their instantiation). More concretely, the NED will effectively enable the disaggregation of inter-domain VNF communications into isolated and secure virtual networks, as well as the configuration of these virtual networks on demand, under the control of the FISHY platform. Secure virtual networks will be built on top of the physical networks that interconnect the FISHY organization domains, which may be provisioned by untrusted Internet service providers.

To automate the management of NED instances, they will be deployed and configured as VNFs on top of the ICT supply chain (i.e., on the IoT/Edge/Cloud infrastructures integrated in the FISHY platform). This avoids the installation and maintenance of additional equipment at every organization domain, as NED functions can be provisioned as any other VNFs on the NFV infrastructures of the organization domains, using the corresponding SIA northbound interfaces. In addition, NED functions will be provided with a management interface, which will be reachable by other relevant components of the FISHY architecture. This way, security decisions made by other FISHY blocks and components can be enforced at the NED functions whenever appropriate. The deployment of NED functions as VNFs also enables appropriate

scaling operations to react to varying traffic demands, either adapting the resources allocated to NED functions (vertical scaling) or increasing/decreasing the number of NED instances at a given domain (horizontal scaling).

Finally, the SIA northbound interface will be used by other FISHY blocks, such as the Enforcement & Dynamic Configuration (EDC), to solicit the deployment of network security functions and services on the NFV infrastructures of the organization domains. These functions can be attached on demand to the layer-2 switch offered by the NED and be provided with connectivity towards other security functions and services operated by the FISHY platforms.

Monitoring & Telemetry

Unlike current commercial solutions, the Monitoring and Telemetry component of the SIA module is: a) able to dynamically monitor deployment changes enforced by continuous dynamic scheduling, provisioning and auto-scaling; b) lightweight, yet effective, and non-intrusive; and c) independent of a specific infrastructure technology. FISHY will containerize a monitoring and telemetry solution collecting and storing data from different sources, including NFV infrastructure monitoring, NFV management and orchestration service monitoring (e.g., Open-Source MANO, OpenStack or Kubernetes), VNF monitoring, SDN monitoring, etc.

5 Communication and interfaces definition

5.1 Interfaces

Interfaces for Trust & Incident Management

The main method of communication with other components within TIM will be the Threat/Attack Repository, which itself will be part of a centralized storage component within the FISHY Platform. Inputting data, such as new cybersecurity metrics, or viewing data on-demand, for example a user navigating to a vulnerability status view on the dashboard, will be handled by an HTTP(S) REST API allowing CRUD operations to be performed by authenticated and authorized parties.

The pub/sub layer of the Threat/Attack Repository will provide an AMQP-based interface, which can be used to subscribe to relevant data sources and receive updates when new data is available, both within TIM and components that are part of other architectural blocks in the FISHY Platform. Examples of pub/sub communication include tools, such as the Risk Assessment Engine (RAE) or PMEM - an R-based application usable as an ML-base mitigation tool (both described in details in D3.1 [1]) subscribing to events related to storing new security metrics, gathered from the ICT infrastructure of the monitored supply chain, and performing an analysis of the data. Storage of the analysis results would then again trigger an update over the pub/sub layer, to be received by a component that can generate new intents or policies to mitigate the perceived risks discovered by the processing of gathered metrics. In case a high priority threat is detected, the pub/sub mechanism can be used to alert an administrator of the supply chain ICT infrastructure of the presence of a threat and recommendations for mitigation.

Using the same pub/sub mechanism, TIM can also receive inputs from other components. For example, when IRO stores a compiled intent in the Knowledge Base (KB), being itself a part of the centralized storage component, TIM is notified and determines what metrics-gathering tools are needed to satisfy the requirements of the intent.

Interfaces for Security & Privacy Data Space Infrastructure

The SPI block implements two main functions requiring different types of interfaces:

- AC and IdM, which, as explained before, do not take part in any data flow operation, being responsible for providing tokens involved in the AC function; from this perspective, the technology used (KeyCloak) provides an endpoint URI through which all required operations are performed.
- Monitoring interface, consisting of capturing measurements from the infrastructure (low-level) components, normalizing the data in a common and standard format, and making it available to high-level modules for posterior processing. This operation involves two interfaces:
  - With the monitoring devices located in the SIA, these devices may need to work in **polling mode** (measurements without real-time requirements), or in **interrupt mode** (critical signals that require immediate attention), for which a call back mechanism must be implemented. Taking the heterogeneous nature of the diverse infrastructures we will face in different supply chains into consideration (well-illustrated by the three use cases present in FISHY), these low-level interfaces need to be tailored to each case. Anyway, these measurements need to be classified according, at least, to the data source, and for making it more informative when transformed to metrics.
  - With the upper-level FISHY components, mainly TIM and SCM, which will consume the measurements as events and metrics. This interface will be implemented by a queue-based mechanism, organized around the data classification (e.g., network events, host events,

performance, security, critical, or even by source components). Concerning the proposed technology and at this stage, we are proposing to use the **RabbitMQ message broker**.

Interfaces for Enforcement and Dynamic Configuration

The EDC will be implemented in Python 3 and will leverage the capabilities offered by the RabbitMQ with MQTT protocol, thus implementing a message broker API available to the other FISHY components.

Once a component (primarily IRO) is subscribed to the EDC queue, it can send to the EDC three main types of messages for:

- sending the high-level policies (stored in the Knowledge Base) and starting their refinement process in order to produce the medium-level policies.
- forwarding the medium-level policies (stored in the Knowledge Base) to invoke their refinement process and generate and store the NSFs' configurations.
- dispatching the configurations (also stored in the Knowledge Base) and beginning their final deployment — this method will trigger the EDC to call the SIA for the final NSFs' deployment.

All the communications with RabbitMQ will be bidirectional, since the EDC will also transmit the success or failure statuses of an operation to the calling component.

The management of NSFs (i.e., running, configuring and eventually stopping them) will be performed by the EDC by leveraging the APIs provided by the SIA using its northbound interface.

Interfaces for Security Assurance and Certification Management

The Audit subcomponent, which is responsible for initiating, coordinating, and reporting the monitoring process will be implemented in Java including capabilities offered by the RabbitMQ message protocol. It includes an API for initiating the auditing procedure based on the selected security metric. Once a security metric is selected for further evaluation, the auditing manager initiates the auditing procedure by creating the corresponding auditing instance, while it sends through the respective API the corresponding message to the Evidence collection engine in order to start collecting evidence that corresponds to the security metric.

Evidence collection engine is also developed in Java and offers its own API interface for communicating with the Audit component.

Interfaces for Intent-based Resilience Orchestrator and Dashboard

The IRO will be based on a Flask server written in Python 3. It will communicate with the dashboard, TIM, and EDC. The dashboard, being the interface to the user, will send the users inputs using REST HTTP to the Intent Manager. This module in turn writes its output to the Knowledge Base via the knowledge base's specific interface which is intended to be based on the Elasticsearch API. The output can consist of multiple kinds of data and will be written to the specific database inside the KB, in this specific case, the intents would go to the Intent Store.

To send policies, the IRO also publishes to the EDC via any of the protocols supported by RabbitMQ with MQTT being recommended. A subscription to the same system is also necessary to be able to receive feedback about the success of a policy planning or enforcement.

The exchange of events between the IRO and TIM is managed by subscribing and publishing to the TIM's RabbitMQ broker. In all subscription-based interfaces both directions are handled by different topics so both parties always know who the intended recipient of a message is. Within the IRO, the internal communication between components is intended to be structured following the JSON format

Interfaces for Secure Infrastructure Abstraction

As it has been commented, the SIA module offers a northbound interface to the other blocks and components of the FISHY platform. The SIA northbound interface provides an abstract and technology agnostic view of the NFV infrastructure resources available at an organization domain and supports the

management and orchestration of network services and VNFs making use of that infrastructure resources.

More concretely, the northbound Application Programming Interface (API) offered by the SIA will support the following functions within the scope of an organization domain: a) management of NFV descriptors (e.g., upload/delete/update) that define network services, Virtual Network Functions (VNFs), and Physical Network Functions (PNFs); b) lifecycle management of network services that operate on the organization domain; c) collection of performance information regarding the execution of network service instances; d) issuing notifications under fault conditions (e.g., a failure on a virtual network that impacts the connectivity of a network service or VNF); and e) provision of information on the capacity of NFV infrastructure resources within the organization domain.

To keep compatibility with relevant standardization efforts on NFV, NFV descriptors (network service descriptors, VNF descriptors, and PNF descriptors) will follow the YANG models defined by ETSI [11], [12]. In addition, the SIA northbound API will be aligned with the RESTful API specification defined by ETSI to support the interaction between an Operation/Business Support System (OSS/BSS) and an NFV orchestrator [13].

It is important to observe that the SIA module must provide the aforementioned API functions regardless of the underlying NFV management and orchestration technologies (e.g., Open-Source MANO, OpenStack or Kubernetes) and SDN controllers (e.g., OpenDaylight, ONOS, Ryu) that may be in use within a FISHY domain. To this purpose, the design of this module considers the utilization of an adaptable southbound interface. The SIA southbound interface will support the interaction with specific management and orchestration software stacks and SDN controllers that exist in a domain, through the utilization of pluggable translating modules (plug-ins). This plug-in-based approach decouples the design and the implementation of the SIA module from specific NFV management and orchestration technologies, which can be supported as add-ons to the southbound interface.

5.2 Communication

As commented in previous sections, the SIA northbound interface provides the point-of-access in the FISHY architecture to interact with NFV infrastructure resources of FISHY adopters in the ICT supply chain. This interface will offer a technology-agnostic mechanism to manage the deployment of network services over FISHY organization domains.

Following ETSI NFV principles, a network service can be conceptually represented as a composition of interconnected network functions, particularly VNFs [14]. Within every FISHY organization domain, VNFs may be interconnected through virtual links as required. These links will be configured by the SIA module using the NFV management and orchestration platform available at the domain (e.g., as self-service virtual networks, in case of an OpenStack virtual infrastructure manager). VNFs that require external connectivity towards other FISHY domains will be attached to a NED instance in their home domain (i.e., the FISHY domain where that VNFs are deployed).

NED instances from different FISHY domains will be interconnected with point-to-point links, building an overlay network that will be used to support inter-domain data-plane communications. Point-to-point links between NED peers will be established using IP tunnels (e.g., based on Virtual eXtensible Local Area Networks, VXLAN[14], or Generic Routing Encapsulation, GRE [16]). These links will be protected through specific security mechanisms, e.g., using IP security (IPsec) [15]. The configuration of the overlay network may be adapted to accommodate different deployment use cases. As examples, a NED peer may be interconnected with every other peer, in case of a reduced number of FISHY domains; or NED peers may be connected to only a small subset of other peers, in larger-scale FISHY deployments. In either case, the overlay network will enable end-to-end data communications between FISHY domains.

In addition, every NED instance in the overlay network will provide a programmable switching function, which may be flexibly configured to enforce forwarding rules for the traffic received onto the point-to-

point links to its NED peers. This way, the overlay network of NED instances will effectively provide a programmable data-plane: VNFs requiring external connectivity are attached to a NED instance, and data traffic among remote VNFs flows according to forwarding state instantiated on the programmable switching functions of the overlay. This state will be configured on each NED instance by an inter-domain connectivity orchestration function, which will be provided by the FISHY architecture.

Each NED instance offers a set of access ports on a FISHY domain that VNFs can attach to. The overlay network can then be configured to forward traffic entering an access port of a NED instance to specific access ports of other NED instances, where other VNFs are attached. From the perspective of the FISHY organization domains, NEDs jointly provide the abstraction of a layer-2 switch, capable of providing link-layer connectivity to VNFs deployed at different domains. The forwarding rules configured on each NED instance allow isolating the traffic exchanged among communicating VNFs, which are effectively connected to the same virtual LANs. Data traffic among NED instances is sent over protected IP tunnels. Hence, inter-domain traffic is securely delivered over potentially untrusted network domains owned by Internet service providers oblivious to FISHY operations. Figure 12 outlines the communication model enabled by SIA components.

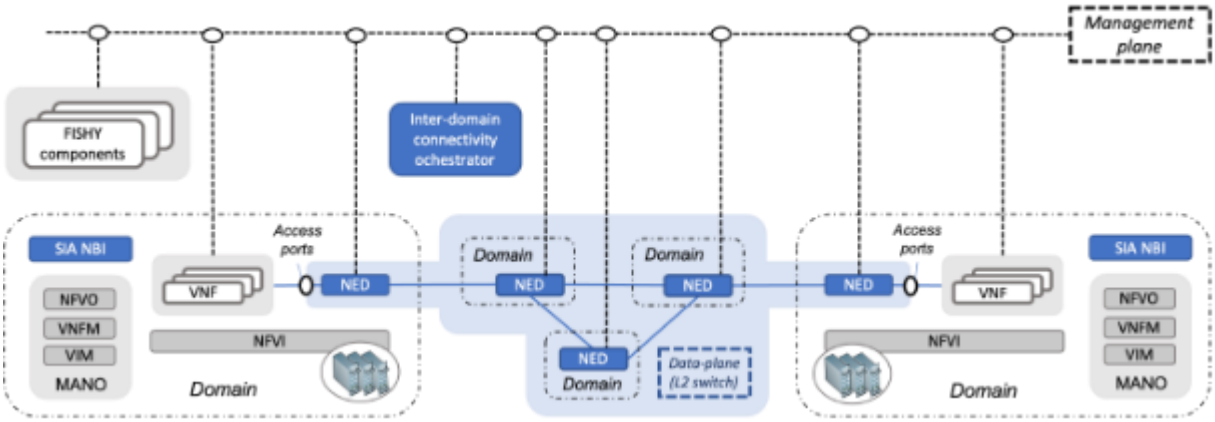

Figure 13: Outline of the management and the data-planes

Following previous experiential reports on the operation of multi-site NFV ecosystems [16], the FISHY platform will incorporate a management plane providing an interface to each VNF. The management plane will also offer interfaces to relevant blocks and components of the FISHY platform. This way, the management plane will: a) provide an effective mechanism to collect significant security-related information, not only from the NED and other network security functions, but from any other VNF deployed through the SIA northbound interface, providing that VNFs support FISHY security related procedures; and b) enable the enforcement of security decisions made by the FISHY platform on the VNFs. In particular, the management plane will support communications between NED instances and the inter-domain connectivity management function, as shown in Figure 13.

6 Hello-world workflow

The main objective of the hello-world workflow is to check the basic functionality of all modules in the FISHY architecture while determining the operational data flow. To this end, the hello-world workflow is intended to be considered as a template for specific workflows at lower level and also to be used when creating specific workflows for the use cases.

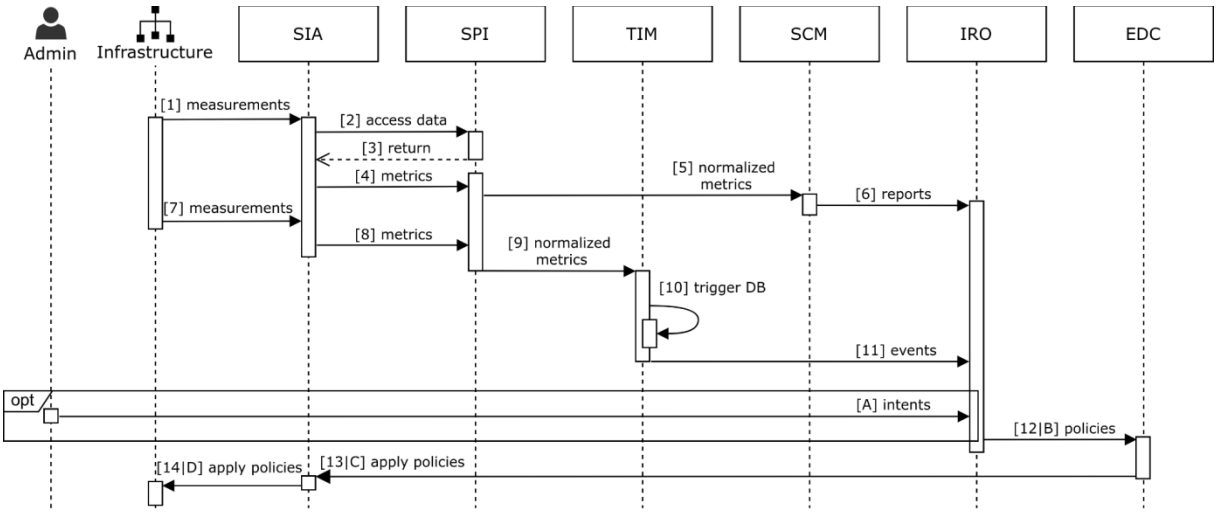

Figure 14: FISHY operational data flow

Any workflow in FISHY is started by:

- either SIA (1), when performing measurements from the infrastructure,
- or by an Administrator using intents through the Dashboard in IRO (A)

Both cases are shown in Figure 13. (For the workflow initiated by the SIA, we use numbers to order the messages while for the workflow initiated by the administrator, we use letters instead, in the figure).

Starting from the infrastructure, SIA is responsible for gathering the measurements (1) from the infrastructure and, together with the monitoring tools, to generate metrics. Before sending these metrics to upper-level components, SIA needs to be authorized by SPI (2,3) and then the metrics are sent to the SPI anonymization component (4). SPI performs anonymization and normalization tasks for the received metrics and sends the resulting corresponding metrics to a queue-based message broker subsystem, where either the SCM or TIM has access (with proper authentication). In the case of SCM (5), the normalized metrics are processed, and the generated results are sent to the dashboard in the IRO as reports (6). In the case of TIM (9), it triggers the database (10) to generate certain events to be sent to the IRO (11). An example of reported events can be information about vulnerability scanning (targeted application, scanning task status, reason of error, logs, recommendations, etc.). At this point, the administrator may perform some action into the system based on the current events and reports, using high level intent to solve the error by setting security requirements (target, error Type, recommended solution/security level indication).

If so (A), the IRO generates the policies accordingly and sends them to the EDC (B). If not, the IRO has the option to select default policies based on the current state of the system (12). In both cases, the EDC is responsible for applying the policies to the system through the SIA (13|C, 14|D).

7 Conclusions

This deliverable provides the specifications of the initial version of the FISHY architecture. Different FISHY components and interfaces, as well as different types of functional and non-functional cybersecurity requirements and constraints imposed by potential services running in the FISHY framework have been analysed and defined in this document.

in Section 2 we focus on the high-level conceptual specification of FISHY architecture, and mapping of different FISHY components and action areas, which is in turn used to define which components will be logically centralized.

In Section 3 we specify general requirements and constraints, which affect the high-level architectural design, and as such need to be addressed for a successful development of the FISHY architecture. Next, modules of the FISHY architecture are described in detail in Section 4, with the initial communication specification and individual interfaces described in Section 5, and the initial operational data flow given in Section 6.

The proposed architectural solution given in this document will serve as input for further work towards the implementation of the FISHY platform modules. The architecture is expected to be updated and refined throughout project duration, as the experiences in the project's integration and implementation phase are bound to bring some modifications. To this end, a final architectural design will be reported in D2.4 (M30).

References

[1] FISHY. *D3.1 Trust Manager components design and implementation (IT-1*). López Diego, Pastor Antonio and Conteras Luis. 2021.

[2] FISHY. *D4.1 Security and Certification Manager components design and implementation (IT-1).* Basile Cataldo. 2021.

[3] Wazuh. *The Open Source Security Platform*. https://wazuh.com/, retrieved 2021-06-04.

[4] Sandhu, R. S. and Samarati, P. (1994). Access control: principle and practice, *IEEE Communications Magazine, 32*(9), 40–48.

[5] Laurent, M. and Bouzefrane, S. (2015). Digital identity management. Elsevier.

[6] Cao Y. and Yang L.. (2010). A survey of Identity Management technology, *IEEE International Conference on Information Theory and Information Security,* pp. 287-293.

[7] OpenID Foundation. *OpenID Connect Core 1.0.* https://openid.net/specs/openid-connect-core-1_0-final.html, retrieved 2021-06-04.

[8] Basney, J., Ananthakrishnan, R., and Koranda, S. (2017). Federated identity management for research organizations, *Illinouis.edu*.

[9] Divyabharathi D. N., and Cholli, N. G. (2020). A Review on Identity and Access Management Server (KeyCloak). *International Journal of Security and Privacy in Pervasive Computing (IJSPPC), 12*(3), 46-53.

[10] OpenID Foundation. *OpenID - The Internet Identity Layer.*https://openid.net/, retrieved 2021- 06-05.

[11]ETSI Group Specification NFV-SOL 005, Network Functions Virtualisation (NFV) Release 3; Protocols and Data Models; RESTful protocols specification for the Os-Ma-nfvo Reference Point, version 3.2.1, September 2009.

[12] ETSI Group Specification NFV-SOL 006, Network Functions Virtualisation (NFV) Release 3; Protocols and Data Models; NFV descriptors based on YANG Specification, version 3.3.1, August 2020.

[13]ETSI Group Specification NFV 002, Network Functions Virtualisation (NFV); Architectural Framework, version 1.2.1, December 2014.

[14] Mahalingam M, Dutt D., Duda K., Agarwal P., Kreeger L., Sridhar T., Bursell M. and Wright C. (2014). Virtual eXtensible LocalArea Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks, RFC 7348.

[15] S. Frankel, S. Krishnan, IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap. RFC 6071, 2011, doi:10.17487/RFC607

[16] Li T., Farinacci D., Hanks S. P., Meyer D. and Traina P. S. (2000). Generic Routing Encapsulation (GRE), RFC 2784.

[17] Martinez, J. R, Lopez D.R. ,Tranoris C., Vidal I. and Gavras A. (2019). Experimentation over distributed 5G NFV-based environments. Zenodo.