Funded by the European Union

Horizon Europe Project TELEMETRY
HORIZON-CL3-2022-CS-01-02

TELEMETRY [101119747]: Trustworthy mEthodologies, open knowLedgE & autoMated tools for sEcurity Testing of IoT software, haRdware & ecosYstems


# D1.1 Use Cases, Requirements and Architecture Specification and Evaluation Plan

| | |
|---|---|
| **Project Reference No** | TELEMETRY - 101119747 |
| **Deliverable** | D1.1 Use Cases, Requirements and Architecture Specification and Evaluation Plan |
| **Work package** | WP1: Requirements, Use Cases and Evaluation |
| **Type** | R - Document, report |
| **Dissemination Level** | PU - Public (fully open) |
| **Date** | 31/05/2024 |
| **Status** | Final v1.0 |
| **Editor(s)** | Robert Seidl (NOKIA), Jörg Abendroth (NOKIA) |
| **Contributor(s)** | Norbert Götze (Nokia), Oleg Zaritskyi (WRCVE), Andrey Kuznetsov (WRCVE), George Triantafyllou (ATC), Antonios Mpantis (ATC), Ravi Borgaonkar (SINTEF), Martin Gilje Jaatun (SINTEF), Lars Flå (SINTEF), Rosella Omana Mancilla (ENG), Francesca Giampaolo (ENG), Francesca Costantino (ENG), Dmytro Prosvirin (ANT), Paolo De Lutiis (TIM), Steve Taylor (UoS), Bernd Ludwig Wenning (MTU), Oscar Garcia (i4RI), Sayon Duttagupta (KUL) |
| **Reviewer(s)** | Rosella Omana Mancilla (ENG), Francesca Giampaolo (ENG), Francesca Costantino (ENG), Steve Taylor (UoS) |
| **Document description** | Stakeholder analysis and use case scenarios definition. Specification of TELEMETRY requirements and reference architecture with a detailed evaluation plan. |

# Disclaimer

The TELEMETRY project is funded by the European Union under grant agreement identifier (ID) 101119747. The information and views set out in this website are those of the TELEMETRY Consortium only and do not necessarily reflect those of the European Union or the European Research Executive Agency (REA). Neither the European Union nor the granting authority can be held responsible for them.

# Document Revision History

| Version | Date | Modifications Introduced | |
|---|---|---|---|
| | | Modification Reason | Modified by |
| V0.1 | 03/04/2024 | Table of content | NOKIA |
| V0.2 | 04/04/2024 | Adding content | NOKIA |
| V0.3 | 31/04/2024 | Partner contributions | All partner |
| V0.4 | 17/05/2024 | Partner contributions | All partner |
| V0.5 | 22/05/2024 | Version for review | Nokia |
| V0.6 | 24/05/2024 | Reviewed version | ENG/UoS |
| V0.7 | 29/05/2024 | Pre-final version | Nokia |
| V1.0 | 31/05/2024 | Final version | Nokia |

# Executive Summary

In this document, 3 example real world use cases, the TELEMETRY requirements, the tools TELEMETRY partners will provide, the TELEMETRY architecture and an evaluation plan are introduced.

The presented use cases represent different industrial domains, namely Aviation, Manufacturing, and Telecommunication. The application of the TELEMETRY tools will facilitate the automation of security testing of these ecosystems.

The architecture was developed iteratively by taking into account the TELEMETRY requirements and through many discussions among the partners. Thus, the interaction of individual tools and their application in the use cases was analysed. Due to these discussions, it was possible to identify potential problems and challenges right from the beginning of the project, allowing early reaction and planning. Furthermore, this architecture serves as a basis for the different implementations of the use case storylines. However, questions were still raised that could not be clarified at the time of writing this document. These questions will be the subject of further discussions and will be addressed in later deliverables.

Finally, an initial evaluation plan to test the interworking of the TELEMETRY tools in the different storylines of the use case was defined. The evaluation would verify the applicability and the impact of the TELEMETRY tools in various situations.

**List of Figures:**

**List of Tables:**

# List of Terms and Abbreviations

| Abbreviation | Definition |
|---|---|
| **AI** | Artificial Intelligence |
| **AMB** | Automated Model Builder |
| **API** | Application Programming Interface |
| **BME280** | Barometric Sensor (from Bosch) |
| **OFIRE** | Onboard Flight Information Recording Equipment |
| **CI/CD** | Continuous Integration / Continuous Delivery / Continuous Deployment |
| **CPU** | Central Processing Unit |
| **CSD** | Circuit Switched Data |
| **CSV** | Comma Separated Values (file format) |
| **CVE** | Common Vulnerabilities and Exposures |
| **CVSSv3** | Common Vulnerability Scoring System v3 |
| **DevSecOps** | Development, security, and operations |
| **DHCP** | Dynamic Host Configuration Protocol |
| **DLT** | Distributed Ledger Technology |
| **DNS** | Domain Name Service |
| **DoA** | Description of Action |
| **Docker** | Open platform for developing, shipping, and running applications |
| **DUT** | Device Under Test |
| **ELK** | Elasticsearch, Logstash and Kibana |
| **EASA** | European Union Aviation Safety Agency |
| **ETSI** | European Telecommunication Standards Institute |
| **FAA** | Federal Aviation Administration |
| **FiaB** | Factory in a Box |
| **GPS** | Global Positioning System |

| | |
|---|---|
| **GUI** | Graphical User Interface |
| **HG** | Home Gateway (same as RGW) |
| **HTTP** | Hyper Text Transfer Protocol |
| **ID** | Identifier |
| **IDS** | Intrusion Detection System |
| **IoT** | Internet of Things |
| **ISO** | International Standards Organization |
| **ISP** | Internet Service Provider |
| **IRIDIUM** | Global satellite communication system |
| **JSON** | JavaScript Object Notation |
| **Kafka** | Open-source distributed event streaming platform |
| **KER** | Key Exploitable Result |
| **KPI** | Key Performance Indicator |
| **Kubernetes** | Open-source system for automating deployment, scaling, and management of containerized application |
| **LAN** | Local Area Network |
| **MES** | Manufacturing Execution System |
| **ML** | Machine Learning |
| **OR** | Operational Requirement |
| **PCAP** | Packet Capture |
| **PDCA** | plan-do-check-act |
| **PNG** | Portable Network Graphic |
| **QEMU** | Quick Emulator (a computer program) |
| **QR** | Quality Requirement |
| **RAM** | Rapid Access Memory |
| **REA** | Research Executive Agency |
| **REST** | representational state transfer |
| **RESTFUL** | An interface that has REST properties |
| **RGW** | Residential Gateway |

| | |
|---|---|
| **SBD** | Short Burst Data |
| **SBOM** | Software Bill of Material |
| **SIEA** | Security Information and Event Acquisition |
| **SMB** | Sever Message Block |
| **SNORT** | Open-source network intrusion detection system |
| **SQL** | Structured Query Language |
| **SSM** | Spyderisk System Modeller |
| **STRIDE** | Spoofing, Tampering, Repudiation, Information disclosure, Denial of services, Elevation of privilege |
| **SUT** | System Under Test |
| **TCP** | Transmission Control Protocol |
| **TIM** | Telecom Italia Mobile |
| **TLS** | Transport Layer Security |
| **TR** | Technical Requirement |
| **TRL** | Technical Readiness Level |
| **TSP** | Time Series Processor |
| **UAM** | User Activity Monitoring |
| **UDP** | User Datagram Protocol |
| **UEBA** | User and Entity Behaviour Analytics |
| **UPnP** | Universal Plug and Play |
| **URL** | Unified Resource Locator |
| **VPN** | Virtual Private Network |
| **WAN** | Wide Area Network |
| **Wazuh** | Open-source security platform |
| **XSS** | Cross Side Scripting |

# 1 Introduction

TELEMETRY will provide trustworthy tools that enable the continuous assessment of heterogenous, interlinked components & systems that constitute Internet of Things (IoT) ecosystems (interconnected IoT devices with hardware, software, services and communications infrastructure). Addressing all aspects of their lifecycle, the TELEMETRY holistic methodology and toolkit incorporates testing for component development, testing & monitoring for component integration into systems, and testing & monitoring for operation of systems. TELEMETRY will deliver advances in cybersecurity testing and runtime monitoring through the use of novel machine learning models and algorithms for real-time anomaly detection; dynamic risk assessment to simulate the likelihood and severity of threat consequences; reputation management and privacy-preserving data sharing across independent entities (e.g. supply chains), IoT device emulation and analysis environment and lightweight approaches for trusted updates; all of which promote a cycle of continuous improvement and assurance across design and runtime phases. TELEMETRY will leverage 3 example use cases representing diverse, complex IoT ecosystems and IoT supply chains in the aerospace, smart manufacturing and telecommunications domains to drive the design and validation of the proposed tools and methodologies. This will lead to significant improvements with respect to the accuracy of threat and vulnerability detection, response time and cost of testing and verification of IoT ecosystems. TELEMETRY will promote open source and knowledge sharing through engagement with relevant communities throughout the project for consultation, dissemination and exploitation of its results.

## 1.1 Purpose and Scope

Deliverable D1.1 provides a detailed description of the use cases including a specific use case architecture, the components and the story lines each use case will realize in order to demonstrate the capabilities of the TELEMETRY tools. The document concentrates also on the requirements, the TELEMETRY tools and architecture, concluded by the description of the validation scenarios.

The use cases addressed in TELEMETRY address different domains: Aviation, Smart Manufacturing and Telecommunication. For each of these domains, TELEMETRY offers a comprehensive toolkit for identifying, assessing, and documenting potential cyber security threats.

By analysing the network traffic in real-time and system parameters of network elements, TELEMETRY pinpoints any irregularities that may arise. Furthermore, it monitors the operational parameters, flagging any deviations from expected behaviour.

TELEMETRY also examines the software components of factory devices and cross-referenced them with a centralized repository of known vulnerabilities.

Subsequently, the findings of the above activities are integrated into a centralized risk management system, which assesses the identified events based on their potential impact, presenting them via a dashboard, including recommendations for risk mitigation activities.

Moreover, TELEMETRY ensures all events are logged in a distributed ledger, ensuring audit compliance and providing a comprehensive record of activities.

## 1.2 Relation to other Work Packages and Deliverables

Since D1.1 describes the use cases and tools, lists the requirements and specifies the architecture, it requires input from all project participants. It touches all use cases and tools. The activities and the outcomes of WP1 "Requirements, Use Cases and Evaluation" are highly dependent on receiving input from all work packages and tasks of the TELEMETRY project. On the other hand, WP1 provides guidance to the technical WPs. This is also indicated in Figure 1.

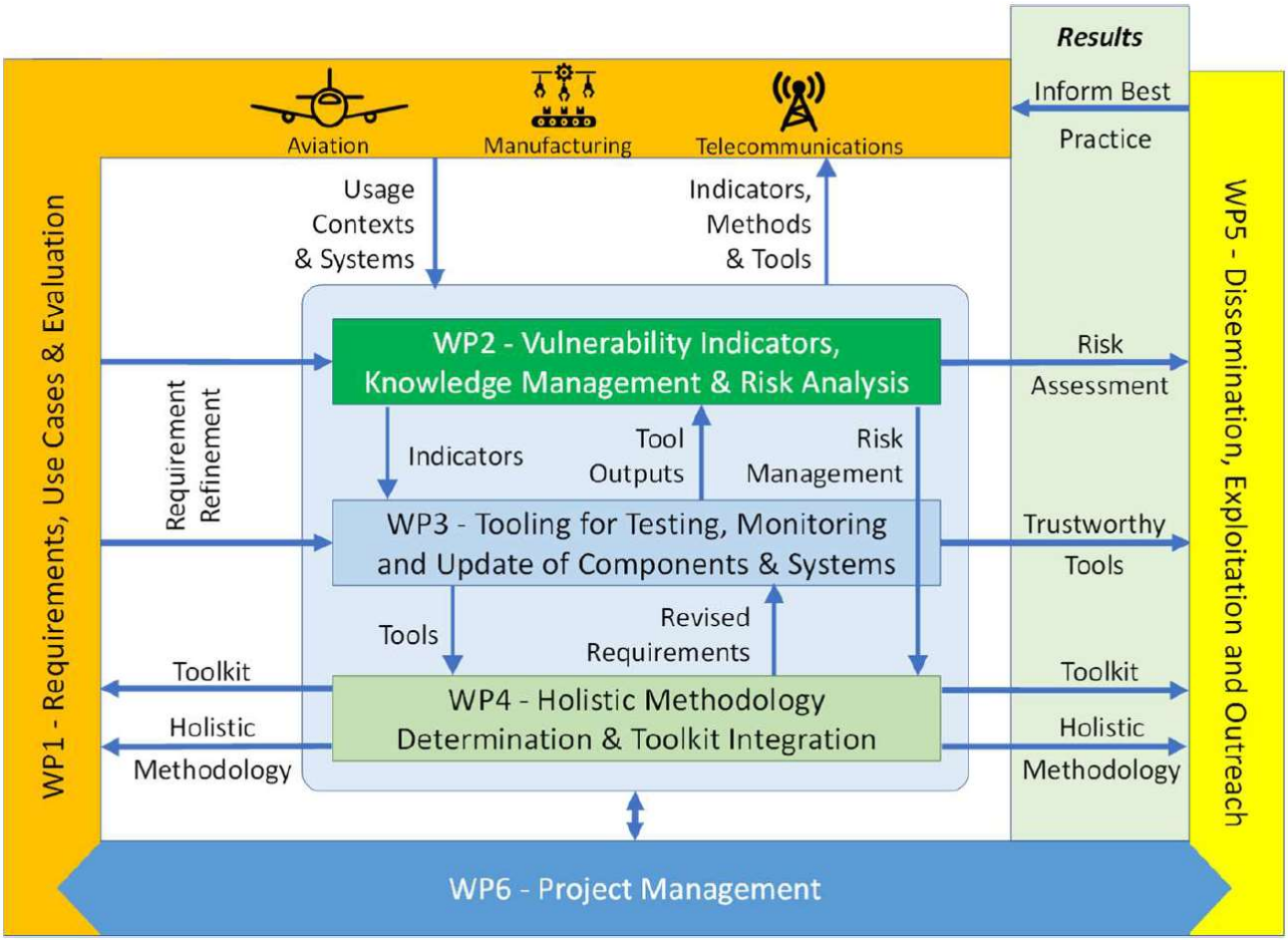



**Figure 1: Relation of TELEMETY Work Packages**

Due to the delivery of this report in an early phase of the project, there is likely to updates regarding the use cases and the architecture. These updates will be reflected in D1.2 "Use Cases Demonstration & Prototype Evaluation", which provides a first demonstration prototype evaluation for the three TELEMETRY use cases at M18 followed by a second prototype evaluation at M24.

## 1.3 Structure of the Deliverable

This deliverable concentrates on the use cases, requirements, TELEMETRY tools and architecture specifications followed by the evaluation plan. The chapters of this deliverable are structured as follows:

- Section 2 provides a detailed description of the use cases including the specific use case architecture and the components and the storylines each use case will realize in order to demonstrate the capabilities of the TELEMETRY tools.
- Section 3 concentrates on the TELEMETRY requirements, which provide the basis for the functionality the TELEMETRY tools need to offer.
- Section 4 describes the initial TELEMETRY architecture that offers a holistic view and includes the interworking of the TELEMETRY tools.
- A description of the TELEMETRY tools is provided in Section 5. Here the functionality of the tools as well as the main technical features are listed.
- Section 6 describes the evaluation of the functionality offered by the TELEMETRY tools in the context of the storylines within the use cases.
- Finally, brief conclusions are drawn in Section 7.

# 2 The TELEMETRY Use Cases

The aim of TELEMETRY is to develop three real-world use cases coming from different industrial domains:

- Use Case 1 Aviation (Antonov)
- Use Case 2 Smart Manufacturing (Nokia)
- Use Case 3 Telecommunication (Telecom Italia)

Each use case is structured in:

- Use case description:
  Describes the overall use case
- Use case architecture:
  Introduces the individual use case architecture
- Use case components:
  Lists the components appearing in this use case
- Storyline(s):
  Specifies the individual scenarios addressed and realised in this use case

## 2.1 Use Case 1 Aviation

Use case 1 focuses on flight data/cargo monitoring by Antonov Company. The main purpose of this is the automated processing of flight data transmitted from the aircraft in online mode (24 hours per day/7 days per week/365 days per year) in order to improve flight safety, and operational communication between the aircraft crew and ground personnel to increase the efficiency and profitability of commercial transportation of ANTONOV Airlines. For this an in-flight communication link of short burst data (SBD) from the plane and circuit switched data (CSD) toward the plane is established along with the existing use of global positioning system (GPS). The principal scenario is illustrated in Figure 2.

Flight data/cargo monitoring is a tool for:

- effective management and analysis of processed flight information (parametric, voice and video information) registered by the onboard flight data recorder, both during normal operation of the fleet and during investigations of flight accidents and incidents
- fulfilment of the 4D flight monitoring function

### 2.1.1 Use Case Description

The flight data/cargo monitoring system is designed for:

- fulfil the requirements of the governing normative documents regarding continuous monitoring of the aircraft position and its technical condition by the operator
- creation of a unified information space, which will allow the Antonov specialists involved to interactively receive the necessary information (parametric, voice and video) about each flight of the Antonov aircraft, including technical characteristics of the condition of the aircraft and its systems, at any moment, while ensuring protected personalized access of different categories of users to the information

- independent automated collection and accumulation of technical information from aircraft on Antonov hardware with its subsequent processing and distribution to certain groups of users within Airlines departments
- increasing the level of informing Antonov specialists by providing the Antonov with operational information on the aircraft movement and requested technical characteristics of the aircraft and its systems in flight
- providing information support to the aircraft crew during the flight or at the airfield in preparation for the flight by transmitting operational data for the forthcoming flight, aeronautical and other information from the Antonov to the aircraft
- operational assistance to aircraft crews and on-board engineering staff in the form of recommendations for actions in the event of special situations during flight and/or technical operation of aircraft outside the Antonov base to eliminate defects after completed flights
- collection and provision of accumulated information from the aircraft to the Antonov management in the current situation that requires prompt response for decision-making
- flight safety management, taking into account the introduction of algorithms for automated processing of flight data related to the condition of aircraft equipment, monitoring of compliance with the rules of aircraft operation by the crew in flight and by the engineering staff on the ground, as well as assessing the level of training of Antonov flight personnel
- ensuring automation of risk management in planning and performing commercial flights of Airlines aircraft, taking into account the objective situation along the route and technical condition of the aircraft, qualification and readiness of the crew and technical personnel on board
- ensuring automated maintenance of the “Electronic Form” of aircraft in order to control the consumption of the actual life of the airframe and engines, as well as to analyse their technical condition, quality of production and repair

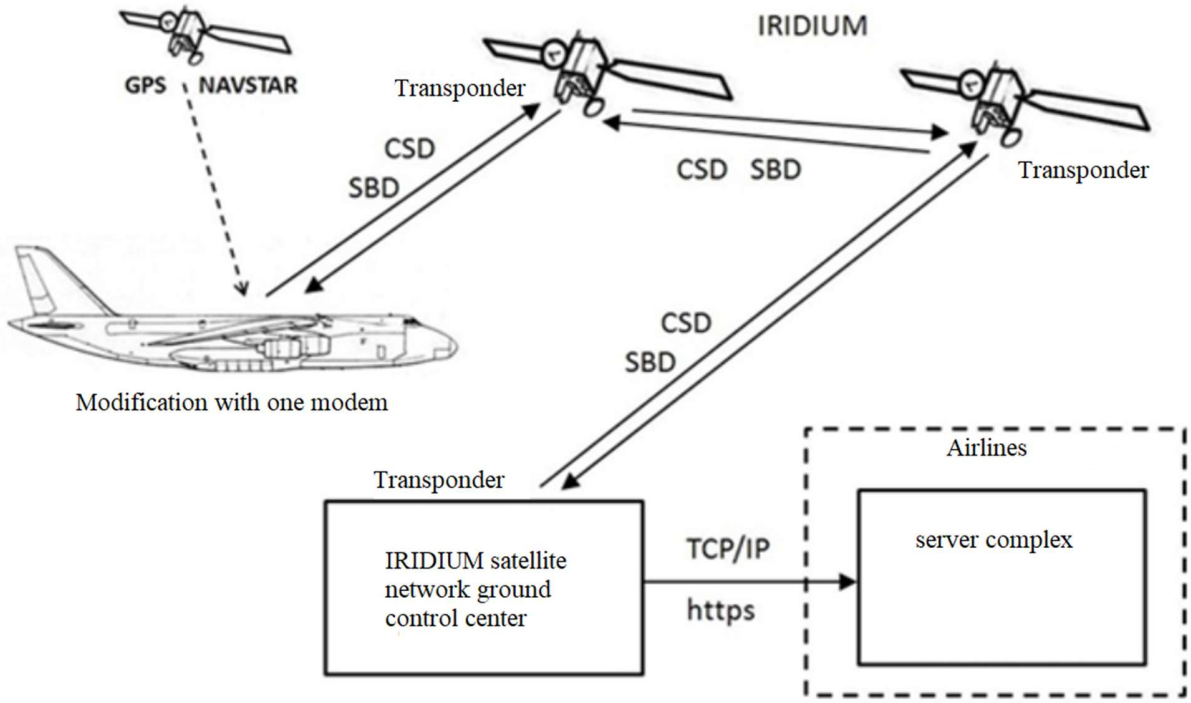



Figure 2: The Scheme of Interaction between on-Board Equipment and Ground Station

The general requirements for the ground-based software and hardware complex are as follows:

- availability of a mechanism for receiving in real-time mode on a permanent basis via satellite communication channels the required volume of specified types of information from the airline aircraft
- availability of a mechanism for receiving the full volume of flight information from the aircraft after the flight, including when it is outside the Airlines base, via Internet channels
- availability of a mechanism for the exchange of information on requests between aircraft and the Airlines via satellite communication channels in sufficient volume provided by the IRIDIUM (global satellite communication) system (at the first stage 9.6 kbit/sec and at the second stage 100 kbit/sec) and Internet channels
- availability of a mechanism for unambiguous interpretation of the data collected by the onboard system and registration transmitted to the Airlines for storage, processing, distribution and analysis
- ensuring the possibility of storing the entire volume of all types of flight data (parametric, voice and video) received from the Airlines fleet (AN-124-100) equipped with onboard flight information and recording equipment (OFIRE) and working with it in real-time for at least three months
- ensuring the possibility of archiving the entire volume of flight data (parametric, voice and video) received from the aircraft of the fleet (AN-124-100) of Airlines equipped with OFIRE and working with it, if necessary, for at least three years

- provision of the required information security level for storage of the Airlines technical flight data, access to it and work with it
- availability of automated standard services for work with flight information in accordance with the needs of existing user groups
- availability of settings of standard automated services of work with flight information, which will provide the necessary completeness of information services to users for the performance of their functional duties and solution of production tasks
- the structure of information provision to the user and user interfaces for access to existing flight information resources and services should be intuitive
- the possibility of subsequent expansion of functional capabilities without structural changes in the software and hardware components of the complex

The main principles of ground-based software and hardware complex creation are:

- use of generally accepted and widely used standards of information structuring and service provision
- unification of formats and protocols of information exchange
- compatibility with the existing Airlines hardware and software complex
- ensuring operation on the basis of the local area network of the Airlines high degree of scalability of software and hardware
- use of effective methods of protection against unauthorized access to information resources and compliance with the information security policy of the enterprise

### 2.1.2 Use Case Architecture (Cargo Monitoring)

Cargo monitoring system are designed to track and register a certain set of parameters.

The system consists of the following main software and hardware elements as illustrated also in Figure 3:

- a computer with a recording device and a parameter indicator
- two digital pressure, humidity and temperature sensors
- digital three-axis overload sensor
- a special program for monitoring parameters, which is installed on a laptop

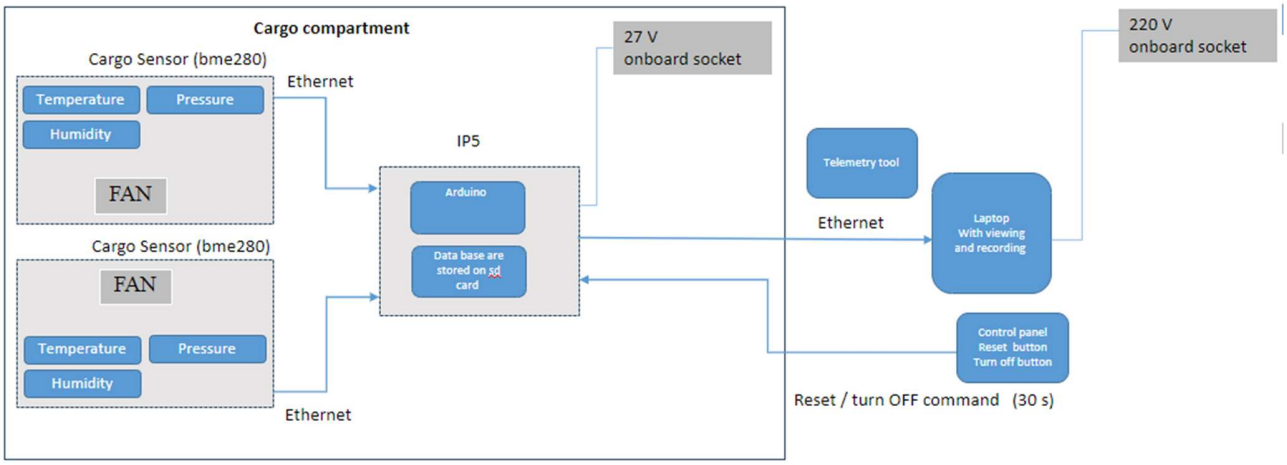



Figure 3: Use Case 1 Architecture Diagram

### 2.1.3 Use Case Components

The components appearing in this use case are described in Table 1.

Table 1: Components of the Use Case Aviation

| Component Name | Description |
|---|---|
| Cargo Sensor (BME280) | Digital pressure, humidity and temperature sensor |
| IP5 | A computer with a recording device and a parameter indicator |
| Control panel | For reset or/and turn off command |
| Laptop | For visual monitoring parameters |

### 2.1.4 Storyline Flight Certification and Safety Testing of Flight Systems

Under current regulations, safety must be considered not only as part of the aircraft type certification in accordance with RTCA DO-326 and EUROCAE ED-202, but also as part of ongoing airworthiness certification. It is proposed that all IoT (Internet of Things) devices on board an aircraft should be certified for security, generally in accordance with RTCA DO-178 and EUROCAE ED-12 for software and RTCA DO-274 and EUROCAE ED-80 for hardware.

In order for a new aircraft to be allowed to fly and transport cargo, especially passengers (so that the company can make a profit), the aircraft must undergo a series of tests and certification of all flight systems. Therefore, certification is indirectly an important factor potentially affecting airline profits.

The security assessment for integrated and complex aerospace systems will be provided via a risk assessment methodology that will use integrated modelling tools to determine a systemic cybersecurity risk assessment.

To meet the requirements of the European Union Aviation Safety Agency (EASA) and Federal Aviation Administration (FAA), the development of Avionics software will be done in Safety Critical Application Development Environment (SCADE) therefore checking for compliance with DO-178C to be done optional, the third year of the AN-124 retrofit program after implementation of flight monitoring system - two different satellite communication channels - INMARSAT (standard, based on the HD-710 unit) and IRIDIUM) use of effective methods of protection against unauthorized access to information resources and compliance with the enterprise's information security policy.

TELEMETRY will provide risk assessment and use anomaly and misuse detection, in components & systems to recognize hazardous conditions at component and system level. Also, TELEMETRY will identify vulnerabilities in access control to system components and will monitor the creation of temporary or permanent users with different access levels and sets of rights. TELEMETRY will also control typical or atypical behavior, provide testing of response to incidents at the HW and SW level (closing user rights, ensuring the safety of logs, sending clusters to quarantine, notifying responsible employees, setting timings, issuing recommendations in case of violation of the conditions or rules for resolving an incident, logging actions when resolving an incident, etc.).

### 2.1.5 Storyline Effective Access Control to IT System Components

Providing secure and efficient access to information resources is an important component of the aircraft production process and its subsequent operation. Testing and optimization of access control systems to information resources will make the work of an airline company more efficient.

The problem with an airline company's complex information access control system is that multiple accesses to information give the user mechanisms to obtain permissions from multiple policies, leading to an accumulation of effective permissions and collectively presenting a certain level of risk. A universal and intuitive tool for testing access control systems will allow the network administrator to improve the quality of decisions made and reduce response time to incidents.

Testing the cybersecurity of IT systems will allow to identify vulnerabilities in access control to system components and regulate the creation of temporary or permanent users with different access levels and sets of rights. It will also allow to control their typical or atypical behaviour, provide testing of response to incidents at HW and SW level. Examples may include:

- closing user rights
- ensuring the safety of logs
- sending clusters to quarantine
- notifying responsible employees
- setting timings
- issuing recommendations in case of violation of the conditions or rules for resolving an incident
- logging actions when resolving an incident

## 2.2 Use Case 2 Smart Manufacturing

### 2.2.1 Use Case Description

The smart manufacturing use case is a portable production facility fit into a standard shipping container, from here on called “Factory in a Box” or short FiaB. The FiaB is a Nokia concept study to investigate the capabilities of such portable smart factories for disaster recovery, and quick ramp-up of local production capacity and serves as a lab to test and develop novel concepts for smart factory operations before they are rolled out. It consists of tools, machines and infrastructure needed to manufacture goods and a software stack, that controls and automates the production process. Time and production critical parts of the software, e.g. Robot Control or worker safety functions, run in the container on a local computing resource. The software includes components like Manufacturing Execution System (MES), Health and Safety applications, Dashboards, etc., and it holds order data and other business-related data. In this use case, the FiaB serves as a realistic representation of a real factory infrastructure to implement the concepts developed in the TELEMETRY project.

### 2.2.2 Use Case Architecture

Figure 4 shows the model of the use case, whereas Figure 5 illustrates the architecture of the use case. In the FiaB, a production robot is being controlled by the MES, which is getting work orders from the dashboard.

To ensure that factory workers are not harmed by the robot, they must wear security gear if they are in the vicinity of the working robot. This is controlled by a video analytics system, which analyses a video feed from a camera overlooking the robot production area. The stream is analysed for people and the security gear they are wearing. This information is sent to the Health & Security system, which decides whether it is safe to run the robot or if it must be stopped (in case a worker without security gear is detected). In implementation terms, the individual components in the FiaB exchange messages in javaScript Object Notation (JSON) format via a Kafka message bus, and a virtual private network (VPN) ingress point acts as a remote access authorization point for users, that connect to the network from remote.

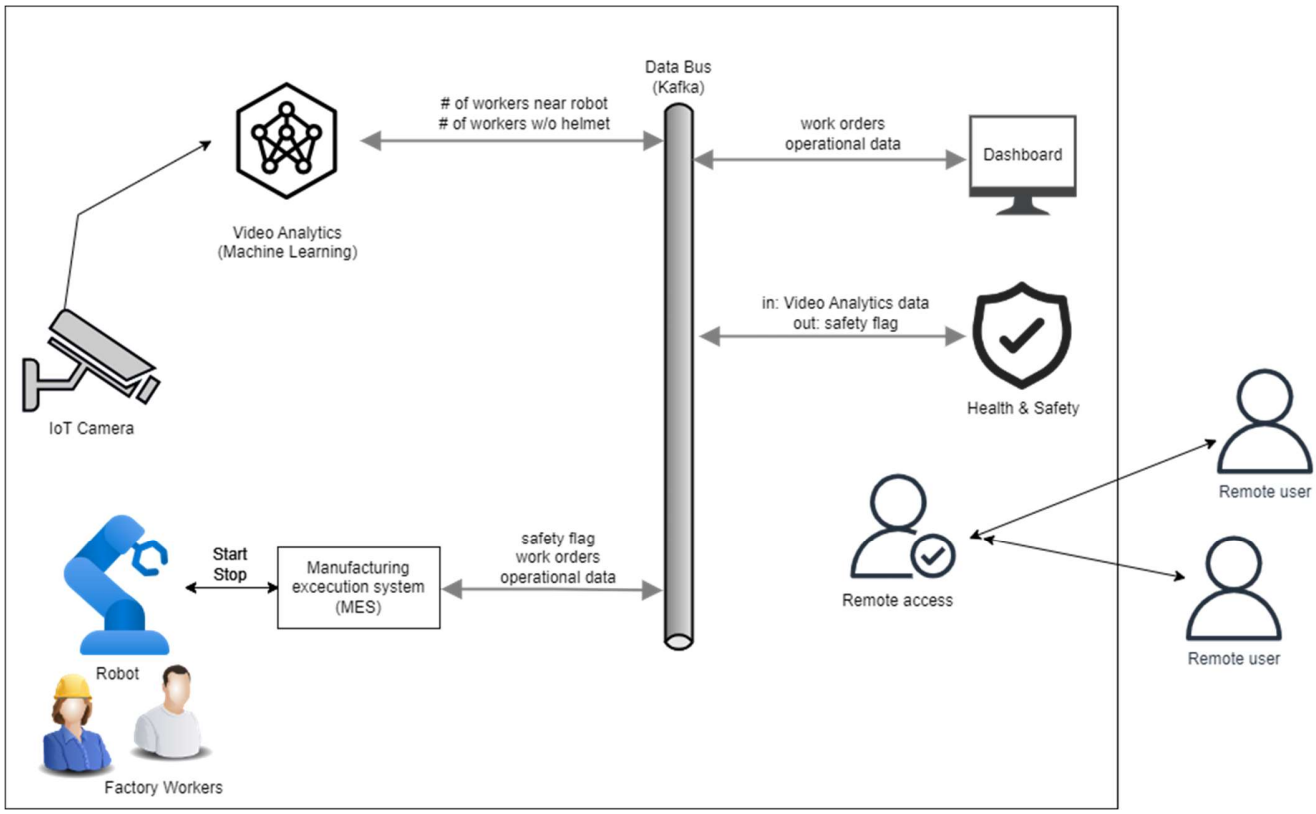



Figure 4: Use Case Model Smart Manufacturing

### 2.2.3 Use Case Components

The components appearing in this use case are described in Table 2.

Table 2: Components of the Use Case Smart Manufacturing

| Component Name | Description |
|---|---|
| Robot | Industry grade manufacturing robot |
| Factory Dashboard | Displays factory and machine operational data |
| Factory Workers | Personnel present in the factory. If they are close to the robot, they must wear security gear. |
| Remote users | Users who are connected from remote to the factory network |
| Remote Access | System to authenticate remote users |

| | |
|---|---|
| IoT Camera | The camera observes the space around the robot and sends a video stream to the video analytics device. |
| Video Analytics | Hardware specialized for machine learning to analyse the camera video stream for factory workers near the robot.<br><br>Provides the number of detected workers and the detected security gear to the Health & Safety application. |
| Health & Safety | System responsible for safe machine operation in the factory. Uses the input from the video analytics to decide if the robot may operate. Informs the MES about the result |
| MES | Manufacturing Execution System, which controls the operations of the factory and is creating execution orders for the robot.<br><br>Receives work orders via the factory dashboard and input from the Health and Safety application. |
| Data Bus | Kafka bus for message exchange between factory components |

### 2.2.4 Storyline

The scenarios addressed in this use case are:

- Network traffic anomaly
- Device operation anomaly
- Rogue device, device behaviour anomaly (“Supply chain risk”)
- 0-day exploit handling
- Event documentation
- Event and risk assessment

Smart factories are a heterogeneous entity consisting of many different individual components. Examples of these components are production machines and network components such as routers, IoT devices, and user devices such as laptops and computers that host software for controlling and operating the factory. As most of these components are "smart" and "online", they are at risk of being attacked or becoming attackers themselves.

Examples of attacks on components are as follows:

- IoT devices whose outdated firmware enables an attacker to take over the device, or the manipulation of control data from production machines with the aim of damaging them or influencing product quality.

- Production machines that collect and send confidential operating parameters can themselves become an attacker. This can be initiated by an untrustworthy manufacturer, or by a legitimate firmware update that has already been compromised by an attacker at the manufacturer ("supply chain attack").
- Devices that are introduced into the network without authorization (e.g., by a disgruntled employee) and launch cyber-attacks from within the network are another potential source of danger to the integrity and security of a smart factory.

Dangers and attacks as described above can already be countered today: Firewalls, rule-based traffic scanners or network segmentation are some of the most common measures with which network administrators try to contain risks of this kind, but these methods are mostly static and only defend against known or anticipated attack vectors. In addition, they must be manually maintained and expanded to be effective in the long term. To address this, TELEMETRY will investigate how to extend this approach with methods that attempt to recognize yet unknown variants of cyber-attacks through machine learning techniques. Specific examples of the approaches to be evaluated in this storyline are as follows:

- Attacks on the production process are detected by analysing the operating parameters of a robot in real-time using a machine learning-based model and detecting anomalies in its operating behaviour.
- To detect covertly operating devices in the network, or to detect suspicious communication behaviour of known elements in the network, TELEMETRY will also use machine learning methods to detect anomalies in network traffic in real-time.
- A Software Bill Of Material (SBOM) of the network elements is regularly compared with the Common Vulnerabilities and Exposures (CVE) database to automatically provide the administrator with information on exploits.
- The central evaluation of the detected attacks and threats is carried out by a central risk assessment system, which collects the findings of the elements described and creates a final risk analysis for the smart factory.
- The result of this analysis, together with information on countermeasures, is then displayed to the administrator on the dashboard.
- A distributed ledger technology system is used to securely and safely document all events that occur.

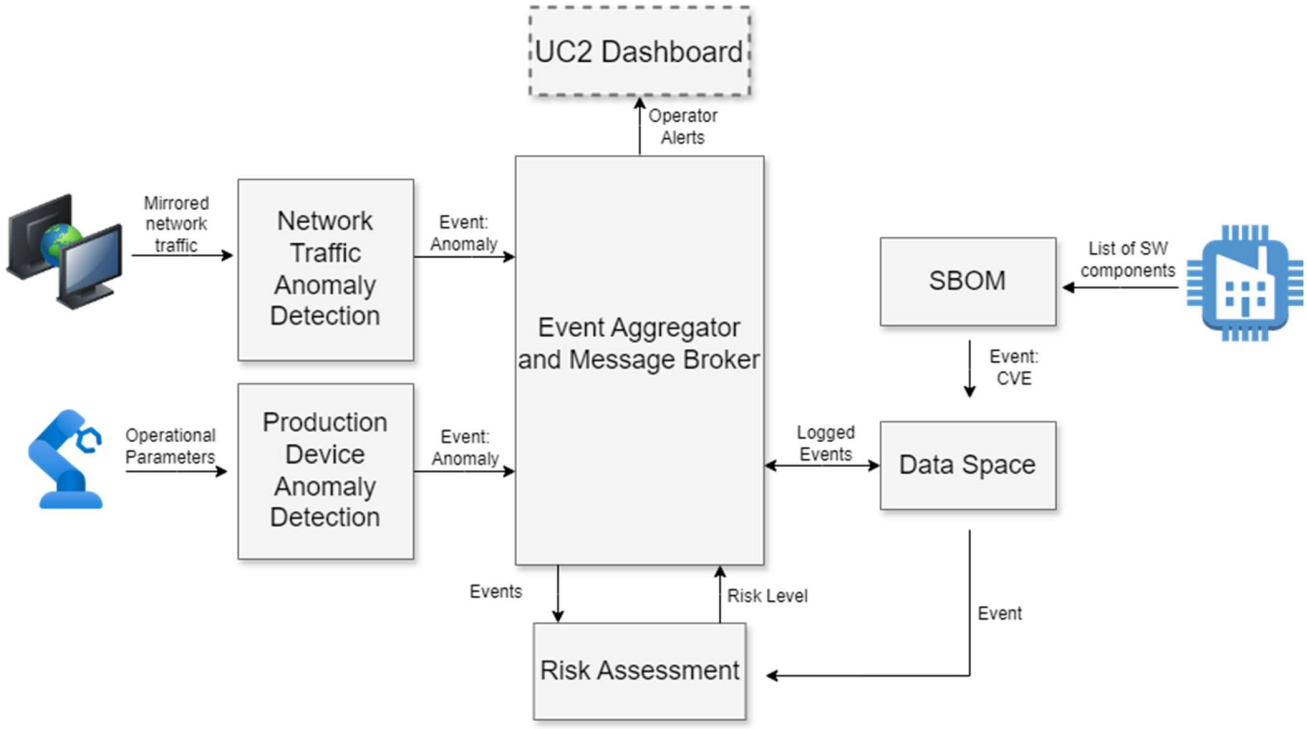



Figure 5: Use Case 2 Architecture Diagram

## 2.3 Use Case 3 Telecommunication

### 2.3.1 Use Case Description

The Telco use case (the third use case proposed in the project, UC3) has been defined to evaluate the TELEMETRY framework in a Residential scenario where Device Under Test (DUT) is a Residential Gateway (RGW), also known as a domestic router. The evaluation context is a testbed simulating a real environment where the RGW will be placed, i.e. at the border between the Telco networks and the customer's devices such as PCs, Smartphones, IPTV, Cameras and so on. The RGWs used are low-cost devices that deliver broadband data, voice, and video services to a home or customer premises, usually via wireless local connectivity (i.e. Wi-Fi). The RGW devices are supplied by third-party vendors and distributed by Telecom Italia to their broadband subscribers. These devices must be tested for vulnerabilities before being passed for distribution, and the testbed enables this testing, TELEMETRY tools will be evaluated as candidates for an updated testbed offering enhanced testing capability. The real environment will be simulated in a testbed where the DUT will be interconnected from one side to the internal Local Area Network (reproducing the home context) and from the other side to the external environment, the public Internet. The testbed will be available to the other partners' labs via secure VPNs.

The main objectives of the Telco test bed are the following:

- To provide an environment to evaluate the TELEMTETRY tools' ability to test the RGW and to produce meaningful and valuable test results.
- Reproduce the real target network environment (i.e. residential customer scenarios).
- Permit remote access to the DUT from the TELEMETRY tool owner.
- Deploy commercial Residential Gateway devices supplied by third-party vendors and distributed by Telecom Italia to the residential customers. Such RGWs will be the targets to be tested during the validation of the TELEMETRY testing tools.
- Provide an isolated environment where it is possible to replicate realistic network IP traffic by means of traffic generators (only synthetic data traffic will be used).
- Host TELEMETRY tools (HW/SW) needed by the tool owners to perform the security tests or host specific sensors/probes needed to accomplish the various testing tasks.
- Provide separate environments for different tool owners when needed. For each tool owner, a dedicated virtual machine will be assigned to avoid possible overlapping among them. Moreover, also a dedicated RGW when needed will be allocated to each tool owner.
- Protect the testing environments from unauthorized access by means of firewalls and secure VPNs.

### 2.3.2 Use Case Architecture

The Telco use case has been defined to reproduce the real process used by Telecom Italia to test from the cyber security point of view its devices prior to deploying them in production (in this case prior to the distribution of the RGW to the actual residential customers). Hence it is related to the design time phase only, without considering the production time, which remains out of the UC3 scope. The testing environment is a simulation of a real environment with normal household traffic, including streaming videos, video conferencing, web browsing, and IoT device communication, although no real users or home devices are actually present. The architecture of this use case reflects such an assumption, and its block diagram is depicted in Figure 6 below.

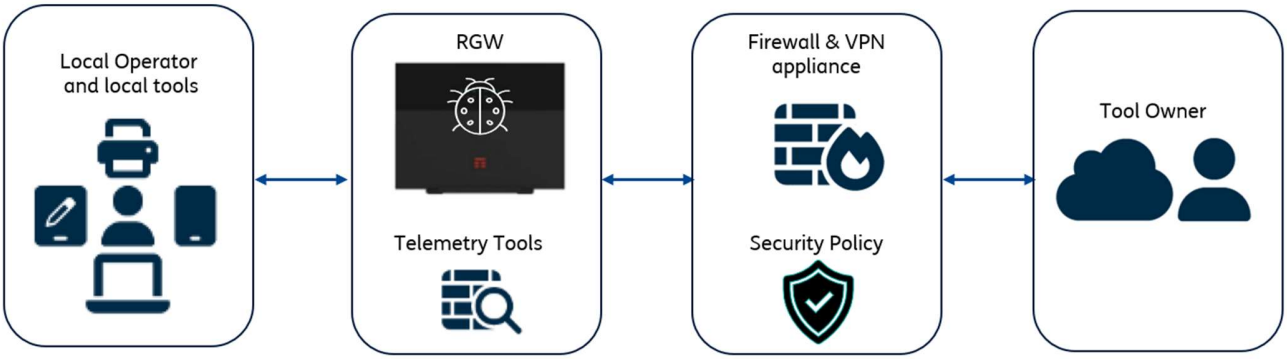



Figure 6: UC3 Block Diagram

TIM (Telecom Italia Mobile) laboratories in Turin host the testbed to be used for the validation phase.

### 2.3.3 Use Case Components

The following table lists the components use in the telecommunication use case.

Table 3: Components of the Use Case Telecommunication

| Component Name | Description |
|---|---|
| Residential Gateway (RGW) | It is the Device under Test that needs to be tested for vulnerabilities and malware. The Residential Gateway is a low-cost commercial product to be tested and validated before deployment in the production environment and provided to subscribers. Following the tests and any remediation work resulting from the tests, the RGW must be free of known vulnerabilities and properly configured to secure posture by default. The wide area network (WAN) interface of the RGW can be directly connected (through the firewall) to the public Internet. At least two different commercial RGW products will be made available for evaluating TELEMETRY tools. |

| | |
|---|---|
| Local tools | The testbed is already equipped with testing tools such as a traffic generator, to simulate local area network (LAN) and WAN traffic, or a sniffer to capture the network traffic. Such tools are managed by the local operators. |
| TELEMETRY tools | They are the tools developed within the project scope and deployed in the test bed and will be evaluated for their ability to perform security testing of the RGW and to produce meaningful and useful results. |
| Firewall & VPN appliance | The testbed is protected from unauthorized access by means of security elements, the Firewall for access control and OpenVPN to secure the network tunnels. Only authorized entities (i.e. TELEMETRY partners) can access the testing environment. |
| Security Policy | The security policy, firewalling and remote access rules, are defined by means of a specific central point. Local operators are in charge of managing all the security policies (e.g. permitted protocols, VPN users, etc.) |
| Local operator | The testbed is managed by local operators, in charge of managing all the aspects of the testbed: system, network and security configuration. The local operator manages also the local tools, e.g. the traffic generator or the sniffers. |
| Tool Owner | Tool owners can manage their tools remotely via secure VPN connections over the public Internet. In some special cases (e.g. tool crash), the tool owner interacts with the local operator to perform specific local tasks (e.g. to restore the RGW configuration, cabling, etc.). |

Several storylines have been identified to better describe the needs and requirements expressed by the Telco Operator TIM for its internal testing processes. Such processes are designed to validate all the products (software or hardware) coming from the suppliers prior to their deployment in the production environments. Security testing is an integral and important part of such processes and is based on internally defined procedures and international standards.

The storylines help to analyse better the Telco use case and hence help to identify the right tool to be used. Moreover, each storyline has been assigned a priority, reflecting the level of importance from the Telcom operator's point of view.

In the specific case of the security testing of the RGW the following storylines have been identified:

Table 4: Prioritization of Storylines of the Use Case Telecommunication

| Storyline | Priority |
|---|---|
| Vulnerability prioritization | High |
| Backporting | Medium |
| Zero-Day or Fuzzing | Medium |
| Virtualization | Medium |
| Access control risks | Medium |
| Supply Chain risks | High |

### 2.3.4 Storyline Vulnerability Prioritization (Priority High)

Vulnerability identification and fixing are time-consuming and costly activities. To optimize both resources and time to market, it is necessary to have a vulnerability prioritization methodology that takes into account the specific devices under consideration and goes beyond the pure application of static score values to incorporate their actual deployment environments.

Currently, identified vulnerabilities are classified according to the common vulnerability scoring system (CVSSv3[1]) base score values methodology. Unfortunately, this method does not take into account possible relationships or dependencies between vulnerabilities and software modules. For example, a "Low" rated vulnerability could be used to exploit a "High" rated vulnerability that is not directly exploitable on the device; this should increase its actual severity. Conversely, there could be "High" rated vulnerabilities that are not directly exploitable due to the device configuration (e.g., usually the user does not have privileged access to the device and cannot change the initial security configuration), reducing their severity. It should be noted that the integrative approaches to CVSSv3 that are typical in the Information Technology world and related to the availability of "Proof of Concepts" in the wild or the detection of exploitation of specific vulnerabilities are not sufficient in this scenario. This is because devices are often customized for individual telecom operators, reducing the effectiveness of these "standard" approaches or the availability of statistical data from other

1 See FISRT organization documentation available at https://www.first.org/cvss/v3.0/specification-document

environments. Hence patching prioritization should be based on the specific target environments.

### 2.3.5 Storyline Backporting (Priority Medium)

During a Security Assessment typically a set of vulnerabilities (i.e. CVEs[2]) is identified, via e.g. an authenticated scan by means of a network vulnerability scanner based on the DUT product and version. Unfortunately, any patch installed by the vendor that does not necessarily trigger a software version change (as is often the case in backporting) is not detected and may produce a false positive because the system has an old version number triggering the vulnerability alert, but the vulnerability may well be fixed by the patch. The result is costly and unnecessary investigation of the false positive. These situations should be handled automatically or semi-automatically, avoiding the current expensive manual approach or just relying on the vendor declarations.

Currently backporting is managed manually case by case by the security testers, hence the need to solve it by means of a tool able to identify the mismatch between the software version and its actual content e.g. by comparing the hash values of the detected libraries with the official ones. Backporting causes an additional effort to analyse all the identified CVEs. Moreover, it might require the vendor's support and cooperation, at least to avoid a full low-level code analysis.

### 2.3.6 Storyline Zero-Day (Priority Medium)

Identifying unknown (so-called zero-day) vulnerabilities is critical in security testing campaigns because of their potential disruptive negative impacts and consequences. In this research area, network fuzzing approaches play a prominent role, as confirmed in the past by important vulnerabilities identified with this technique. In this case we assume the scenario from the network to the device.

The goal of a security testing campaign is of course to identify known and unknown vulnerabilities in the DUT. For the latter, the fuzzing approach typically provides excellent results. The fuzzing approach can be used for a wide range of services, addressing the entire attack surface of the device. For example, it can be used to analyse all services and network protocols exposed on both the LAN and WAN side, such as hyper text transfer protocol (HTTP), transport layer security (TLS), domain name service (DNS), dynamic host configuration protocol (DHCP), Server Message Block (SMB), Universal Plug and Play (UPnP), and so on. The same approach can also be used for testing various software modules and binaries capable of accepting different types of input.

### 2.3.7 Storyline Virtualization (Priority Medium)

Many times, security testers are bound to work with a limited number of actual devices or DUTs, which leads to problems with performing some special kinds of attacks because they

2 https://www.cve.org/

causing device crashes and then require time-consuming and labour-intensive restoration of the DUT's original state before restarting the security testing operations.

The proposed solution is to create a virtual environment where they can perform attacks on the source code and try to exploit zero-day attacks as well as well-known vulnerabilities with no impacts on the actual devices and then delay the testing procedures. Firmware virtualization could allow security testers to try to break the DUT with aggressive attacks without physical consequences. Moreover, the availability of a virtual DUT that can be instantiated multiple times allows for greater test parallelisation and faster test execution, which is very useful especially in fuzzing tests.

### 2.3.8 Storyline Access Control Risks (Priority Medium)

The residential gateway access control system plays a key role in protecting the device, reducing the attack surface and the likelihood of successful attacks. A weak access control system on the part of the corporate client is one of the factors of the attack and creates additional conditions for exploiting the gateway vulnerabilities.

The problem is the increased risk of exploiting gateway vulnerabilities in the case of a weak access control system on the corporate client's side. The risk assessment of the access control system involves taking into account the presence and level of software and hardware vulnerabilities. The problem is tackled by analysing the network architecture, access control system and data provided by the TELEMETRY tools of partners (WAZUH, SNORT, Machine Learning, short ML, ...) regarding the level of vulnerabilities as input data for the risk assessment methodology, to provide recommendations to the system administrator and internet provider to fix the identified security issues.

The methodology for assessing the risks of an access control system on the part of corporate clients involves the use of a wide range of analysis tools of the state of the residential gateway, analysis of access policies, analysis of the level of the access control system to formulate the magnitude of the risk of exploitation of detected vulnerabilities, taking into account the real state of the system. The methodology involves issuing a recommendation to the system administrator on how to change access policies to increase the level of security of objects with a high risk of attack, if the existing policy contributes to the realization of such a risk.

### 2.3.9 Storyline Supply Chain Risks (Priority High)

The firmware of the Devices Under Test contains a large number of commercial and open-source software components or libraries. A successful attack on the manufacturer's infrastructure or a public repository of any of these components or libraries could lead to the inclusion of malicious components on the DUTs which, if undetected, could have major consequences for the operator and its customers. A malicious component can for instance collect user data and send it to an external server, change the device configuration to perform illegal interception (for instance changing the legitimate DNS servers), and so on. Moreover, the scale of the impacts is another big concern, considering the huge number of residential gateways to be distributed in production to the customers.

The solution should be able to identify the chain of the suppliers, e.g. with a SBOM approach, and correlate such information with threat intelligence feeds that provide information on the

latest threats and vulnerabilities affecting the identified firmware/software. Other possible solutions can collect and analyse data in real time and identify unusual patterns or anomalies that may indicate a security issue.

# 3 The TELEMETRY Requirements

## 3.1 Defining Requirements

The definition of requirements is essential for the technical developments in the project. A requirement describes an attribute of TELEMETRY components or system that needs to be fulfilled to achieve a certain outcome. It serves as well as the basis for the use cases. This subsection covers the requirements TELEMETRY solutions will have to satisfy. The requirements are summarized in Table 5 and detailed by the Requirement Description, the Type of Requirement, the Source/Reason/Explanation and the Owner of the Requirement columns.

## 3.2 Types of Requirements

We divided the requirements into three different types:

- Technical Requirement (TR)

  A technical requirement describes a functionality a TELEMETRY system, a TELEMETRY tool or a function should provide. Normally it is coupled with research and development effort.

- Operational Requirement (OR)

  An operational requirement defines administrative tasks, like processes or outreach activities TELEMETRY needs to address in order to allow effectiveness and support publicity.

- Quality Related Requirement (QR)

  A quality related requirement defines a specific quality characteristic of a TELEMETRY system, a TELEMETRY tool or a function. These characteristics could be defined in a quantitative way (e.g. 10 times faster).

## 3.3 Table of Requirements

The following section list the requirements addressed by the tools developed and used by TELEMETRY. Most of them are derived from the Description of Action (DoA) and the use cases. More details can be found there.

**Table 5: TELEMETRY Requirements**

| ID | Requirement Description | Requirement Type | Source | Requirement Owner |
|---|---|---|---|---|
| 1 | Automatic detection of cybersecurity threats, risks and controls to address risks | TR | DoA | UoS, ATC |
| 2 | Component Level Anomaly Detection | TR | DoA | Nokia, ATC |
| 3 | System Level Anomaly Detection | TR | DoA | ATC, ENG |

| | | | | |
|---|---|---|---|---|
| 4 | Detect misuse of components & systems<br>Detect misuse of software components & systems | TR | DoA | i4RI |
| 5 | Provide synthetic data to simulate misuse<br>Provide a synthetic data generator | TR | DoA | i4RI, MTU |
| 6 | Effective access control to IT system components | TR | DoA Project Objective O3 | WRCVE |
| 7 | Trusted mechanism to facilitate distributed sharing of testing, verification and security-related information<br>Distributed sharing of testing, verification and security related information | TR | DoA Objectives Section | MTU |
| 8 | Automated design time, vulnerability-driven risk assessment for components & systems<br>Risk assessment at component and system level, with recommendations of controls to lower risk levels<br>Semi-automated risk management of devices, components & systems Automated model-based risk management of devices, software and systems<br>Semiautomated risk management of devices, components & systems | TR | DoA Objectives Section<br>DoA Progress beyond the state-of-the-art | UoS |
| 9 | Common information model to represent testing and verification outcomes | TR | DoA Objectives Section | MTU |
| 10 | Create tools and techniques to test, detect and address cybersecurity and privacy vulnerabilities & risks in components, data and systems at both design time and runtime | TR | DoA Objectives Section | UoS, SINTEF, NOKIA, ENG, WRCVE, ATC, i4RI |

| | | | | |
|---|---|---|---|---|
| 11 | Provide tools & techniques for vulnerability / anomaly detection in components & systems to recognize hazardous conditions at component and system level | TR | DoA Objectives Section | ENG, ATC, i4RI, NOKIA |
| 12 | Provide tools & techniques for testing components & systems to cover gaps in the testing tools available today | TR | DoA Objectives Section | ALL |
| 13 | Provide tools & techniques for secure updates of components & systems | TR | DoA Objectives Section | KUL |
| 14 | Provide a cybersecurity testing toolkit platform and testing environment | TR | DoA Objectives Section | ATC |
| 15 | Trustworthy framework for sharing security-related information | TR | DoA | MTU |
| 16 | Automation of secure data management services, policy enforcement and governance | TR | DoA | MTU |
| 17 | Secure backbone for data flows across independent entities | TR | DoA | I4RI |
| 18 | Automate governance, auditing and assurance processes | QR | DoA | MTU |
| 19 | Reference ontology for data sharing | QR | DoA | MTU |
| 20 | Automated recognition of security events | TR | DoA | ENG |
| 21 | Hybrid approach to combine signature-based detection with behaviour-based detection | TR | DoA | ATC, ENG |
| 22 | Develop and evaluate a security analyser for Application Programming Interfaces (APIs) | TR | DoA | MTU |
| 23 | Assess the risk of different access control policies | TR | DoA | WRCVE |

| | | | | |
|---|---|---|---|---|
| 24 | Database for generating test-cases | TR | DoA | ALL |
| 25 | Secure remote distribution of firmware updates | TR | DoA | KUL |
| 26 | Design a lightweight key management solution for distributed and decentralized systems | TR | DoA | KUL |
| 27 | Design a lightweight cryptographic MAC algorithm | TR | DoA | KUL |
| 28 | Toolbox and methodology that promotes continuous improvement at design time and runtime | TR | DoA Impact Section | SINTEF |
| 29 | Automated detection of anomalous behaviour | TR | DoA | ENG, ATC |
| 30 | Efficient and secure patching and updating | TR | DoA | KUL |
| 31 | More resilient digital infrastructures, systems and processes | QR | DoA | ALL |
| 32 | Increased software, hardware and supply chain security | QR | DoA “Outcome” | TIM |
| 33 | Secured disruptive technologies | TR | DoA | ALL |
| 34 | Smart and quantifiable security assurance and certification shared across the EU | QR | DoA | ALL |
| 35 | Reinforced awareness and a common cyber security management and culture | QR | DoA | ALL |
| 36 | Management of trustworthy updates | TR | DoA "Outcome" | KUL |
| 37 | Modelling of security and privacy properties | TR | DoA "Outcome" | UoS |
| 38 | Frameworks for validating and integrating the testing process | TR | DoA "Outcome" | ATC |

| | | | | |
|---|---|---|---|---|
| 39 | Integrated process for testing, formal verification, validation and consideration of certification aspects (including potential synergies with the EU cybersecurity certification framework, as established by the EU Cybersecurity Act) | OR | DoA "Outcome" | SINTEF |
| 40 | Tools providing assurance that third-party and open-source components are free from vulnerabilities, weaknesses and/or malware | TR | DoA "Outcome" | SINTEF, UoS, ATC, ENG, KUL, WRCVE |
| 41 | Automated anomaly detection at both device / component and system level | TR | DoA "Outcome" | ATC, ENG |
| 42 | Data security “by design” e.g. via secure crypto building blocks | TR | DoA "Outcome" | KUL |
| 43 | Instrumentation and secured communication with system components for dynamic testing | TR | DoA "Outcome" | MTU |
| 44 | Methods and environments for secured coding by-design and by-default and secure hardware and software construction | TR | DoA "Outcome" | SINTEF |
| 45 | Effective audit procedures for cybersecurity testing | TR | DoA "Outcome" | MTU, SINTEF |
| 46 | Reduce the energy cost of data authentication | QR | DoA | KUL |
| 47 | 20%+ increase in detection of known vulnerabilities (UC1) | QR | DoA | ANT |
| 48 | 20% reduction in maintenance and monitoring effort for keeping smart factories secure (UC2)<br>20% increase in detection of known vulnerabilities (UC2)<br>20% reduction of time to take actions on heterogeneous networks safety (UC2) | QR | DoA | NOKIA |

| | | | | |
|---|---|---|---|---|
| 49 | 20% + increase in the detection of security issues in the domestic context (UC3)<br>20% + increase in the detection of security issues at device level (UC3) | QR | DoA | TIM |
| 50 | 10% increase in initial identification of anomalies | QR | DoA | ENG, ATC |
| 51 | Vulnerability detection shows detection improvement of 20+% and 10% reduction in false positives over the current approach for known vulnerabilities | QR | DoA Objectives Section | ENG, ATC |
| 52 | Cost for securing updates is 20% lower than state of the art cryptographic techniques for securing data | QR | DoA Objectives Section | KUL |
| 53 | Demonstration of integration of 3+ third party tools.<br>Demonstration of 3+ third party components in TELEMETRY environment. | QR | DoA Objectives Section | ALL |
| 54 | Effective communication and dissemination strategy | QR | DoA Objectives Section | ALL |
| 55 | Effective impact maximization strategy | QR | DoA Objectives Section | ALL |
| 56 | Robust and effective and fit stakeholders' needs tools & methodologies | QR | DoA Objectives Section | ALL |
| 57 | User trials successfully undertaken.<br>Documented user stories demonstrating TELEMETRY tools, toolkit & methodologies proof of effectiveness, utility & value | QR | DoA Objectives Section | ALL |
| 58 | Integration of 8+ Telemetry tools with distributed data space. Provide | QR | DoA Objectives Section | ALL led by MTU |

| | | | | |
|---|---|---|---|---|
| | interfacing with 3+ external information sources | | | |
| 59 | Strengthened EU cybersecurity capacities and European Union sovereignty in digital technologies | QR | DoA | ALL |
| 60 | Improve security of an air cargo monitoring system | QR | UC1 | ANT |
| 61 | Detection atypical behaviour of onboard information sensors | TR | UC1 | ANT |
| 62 | Combine multiple detection mechanism to create a comprehensive threats picture that stands complex and changing environment of a FiaB | QR | UC2 | NOKIA |
| 63 | Support for backported vulnerability identification | TR | UC3 | TIM |
| 64 | Support of vulnerability prioritization | TR | UC3 | TIM |
| 65 | Support of DUT virtualization | TR | UC3 | TIM |
| 66 | Identification of 0-day vulnerabilities | QR | UC3 | TIM |

# 4 Initial TELEMETRY Architecture

This section describes the initial starting point for the architecture of the TELEMETRY tool suite. The architecture will be developed and altered as necessary during the course of the project and updates will be reported in subsequent deliverables as appropriate. The architecture is intended to provide an understanding of the interactions between the TELEMETRY tools and infrastructure in order to enable the combined result to support operators in testing and monitoring the cybersecurity of IoT and other devices either alone or deployed into operational systems.

## 4.1 Design-Time versus Operation-Time

TELEMETRY partners are making tools that are applicable for use in design time or during operation. The main difference is whether (end-)users are able to interact with the software.

### 4.1.1 Design-Time

Design-time tools can be used in various phases of the (secure) software development lifecycle up until deployment (possibly including deployment, but not after deployment). Depending on the phase, the software may not yet exist (requirements, design), be incomplete (security testing), or completely (or as near as) finished (penetration testing).

This can include "conformance testing" of third-party hardware before inclusion in a system.

### 4.1.2 Operation-Time

Tools for operation are aimed at finished software running in its intended environment. Such tools may include post-deployment penetration testing tools, intrusion detection tools, and various monitoring tools. Such tools may need real traffic for their operation.

TELEMETRY is focusing on tools in Operation that enable us to detect bugs and flaws that we can feed back to the development phase for rectification ("feedback from the field" to quote Gary McGraw).


**Figure 7: Design-time vs. Operation from the Perspective of the Microsoft Security Development Lifecycle**

Figure 7 shows the Security Lifecycle where during design time TELEMETRY Tool training takes place, as well as deriving requirements, design, implementation, verification and release of the tools and ruleset or policies. Then during operation, the responses are being recorded, both for operation of the full system, as well as input for the next iteration of the TELEMETRY Security Lifecycle.

Hereby yellow denotes where data collection takes place and green where non-real-time activities take place.

## 4.2 Initial Generic TELEMETRY Architecture

TELEMETRY combines tools located at different phases of the software life cycle. Prior to deployment, TELEMETRY supports testing of a new component (e.g. home gateway or component of a smart factory) for security properties. In the Testing phase, different testing tools are used, but importantly also insights from the operation phase, such as past incidents or operation anomalies are considered. Once the device is in operation, the operational phase employs components that monitor the device to detect incidents, events and anomalies. This occurs in real-time events, where one event may not generate an important alarm, but the conjunction and correlation of different events cause an important alarm to be raised.

### 4.2.1 Conceptual Architecture

The TELEMETRY conceptual architecture is shown below in Figure 8. The device/system under test is shown in green, and the TELEMETRY tools for both testing and runtime monitoring are shown in blue, both of which provide input to a risk model for the system under test (SUT) shown in purple. The TELEMETRY infrastructural components, (the Distributed Ledger Data Space, the Dashboard and the Workflow Orchestrator) are shown in different shades of grey. Red arrows denote configurations/setups for tools, blue arrows denote inputs and outputs for tools that are typically operated by people, e.g. the testing tools or the risk model, and orange arrows denote events automatically generated by monitoring and anomaly detection tools, and finally, grey arrows denote execution control messages for both testing and monitoring tools based on workflow instructions from the orchestrator.

In Figure 8 the monitoring tools are illustrated as a block, representing a placeholder for several tools, namely Anomaly Detection Pipeline, BACON, Wazuh Security Platform, Misuse Detection ML Toolkit, SNORT + Ruleset, r-Anomaly Detection- and Network Anomaly Tool. Similar Testing Tools substitutes any or a combination of SBOM Generator, Digital Twin Emulated Infrastructure, IOT Fuzzer, Cyber Range in the Box and Cyber Security Testing.

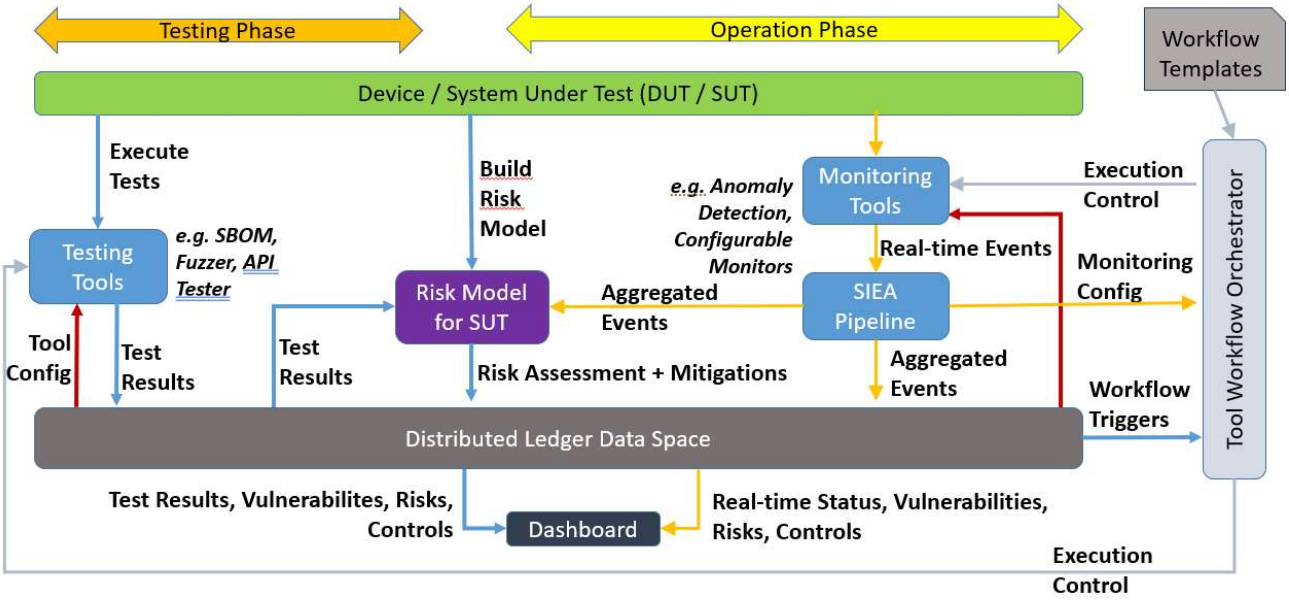



Figure 8: Initial TELEMETRY Conceptual Architecture

The purpose of the conceptual architecture is to provide a basis that supports tools running flexible configurations and sequences triggered either automatically by events, under workflow control or manually by the operator and the architecture is intended to support configurable sequences (workflows) of tool executions where the output of one tool provides input to another. For example, there may be natural sequences of tools, which can be encoded as Workflow Templates, a human operator may choose to run one tool and from observation of results choose to run another tool, an event from monitoring may trigger a workflow, or an event may be output to the dashboard and the operator runs another tool to find more information. An aim of the conceptual architecture is to be flexible enough to support interactions between tools and the operator, so that information gathered in one phase may inform another phase – e.g. monitoring results in the operation phase can inform subsequent testing.

A key integration point is the Distributed Ledger Data Space. This provides a common repository of events, configurations and results for the tools. By virtue of it being a Distributed Ledger, the Data Space supports the immutability of data and non-repudiation, both of which are important for storing evidence to support audit, compliance and certification. A Dashboard is connected to the Data Space (and possibly other tools if they have their own graphical user interface, short GUI) to provide a single point to get all relevant information, both static and dynamic, regarding the testing and monitoring of the DUT/SUT.

The risk model consumes many results from other tools, for example, test results from testing tools and events from monitoring tools. The risk simulation tool, from the University of Southampton (UoS) Spyderisk, has a mechanism in place where parameters may be adjusted to reflect changes in vulnerability, and this mechanism will be exploited and extended as the project progresses to accommodate information from these different tools. The simulator can then reflect these changes in vulnerability into risks, which can be stored in the Data Space and reported to the operator via the Dashboard.

### 4.2.2 Detailed Architecture

Within the high-level architectural components, there exist more TELEMETRY components, the tools, that reside either in the Testing Phase or the Operation Phase. Figure 9 shows details of the testing phase, with details on the input and output interfaces. Blue lines denote interfaces where two tools interact with each other, such as retrieving information or causing actions. Black lines are a standard interface that several tools utilize, such as the Aggregator and Distributor or the Data Space. In case of Data Space, Tools can insert or retrieve data from the Data Space.

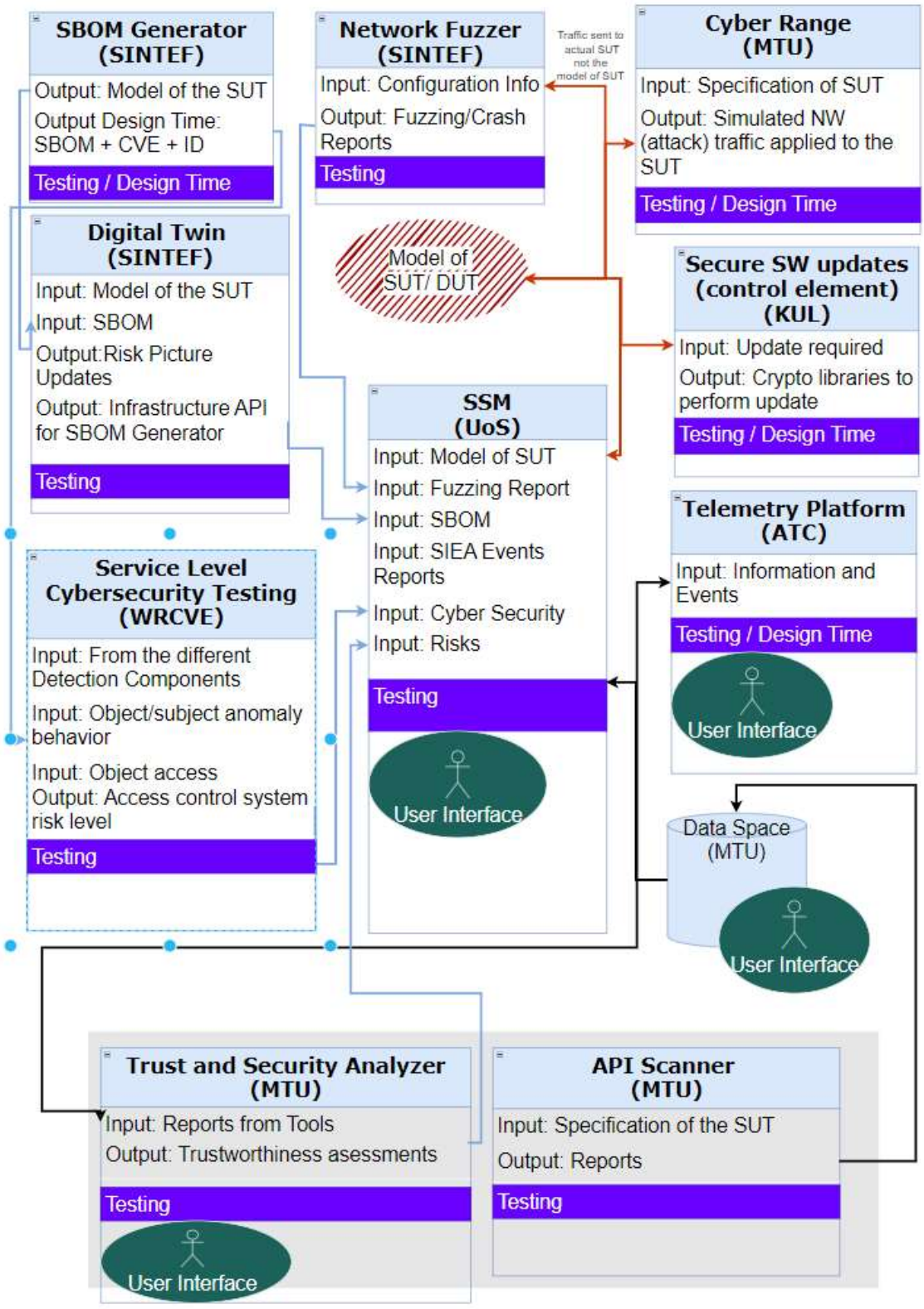



Figure 9: Architecture Testing / Non-Realtime Phase

The orange line indicates which tools interact with the system under test or a representation of it. The line does not denote a well-defined interface but depends on the concrete system under test. It can be additional sensors, environment observation or interaction with an existing interface that this line represents. The lack of a standardized testing interface poses unique challenges but cannot be averted. Each TELEMETRY tool that interacts with the system under test defines its own interaction mode. The aim of the digital twin is the creation of a well-formed system under test model, e.g. by way of firmware virtualization. The interfaces from the testing tools toward the Data Space are well-defined and, as seen in the high-level architecture, are the means of distributing information from the Testing Phase to the Operation Phase.

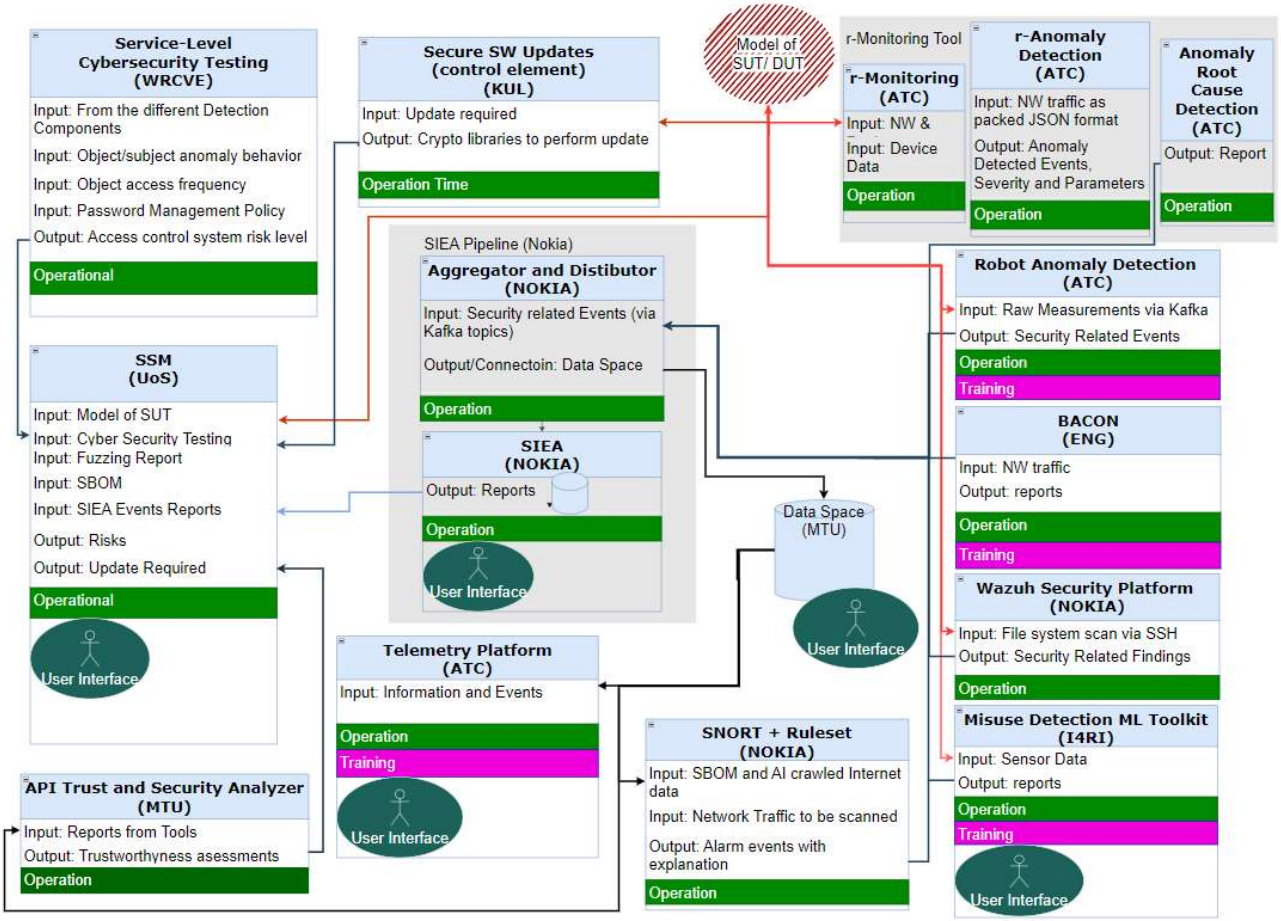



Figure 10: Architecture Operation / Realtime Phase

Figure 10 above shows the Operation or Realtime Phase. The security information acquisition (SIEA) Pipeline is an important element, which can handle real-time events and prevent issues that can arise from the real-time nature to have negative consequences. The Spyderisk system modeller (SSM) and TELEMETRY Platform (Dashboard), along with other tools, both display information to the user, e.g. security administrator. The SSM perspective is the risk levels,

while the Dashboard focuses on the Alarms and Events, including the correlation of low-level events to form higher-level events.

The interaction with the system under test, again, depends on the actual system and might also more focus on sensor information in the environment, such as network traffic, that is being collected.

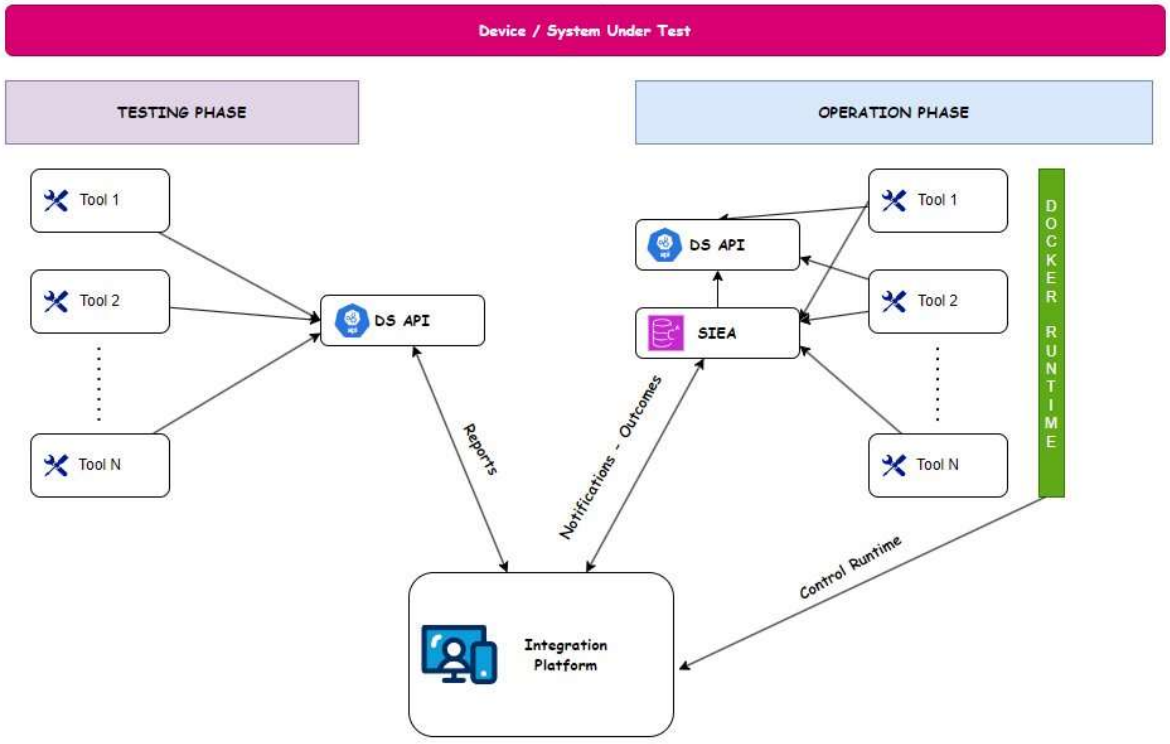



Figure 11: Architecture TELEMETRY Platform

The TELEMETRY platform is an infrastructure provided to all tool owners within the project, enabling them to manage the tools created during the project's lifespan. Figure 11 above illustrates how the TELEMETRY Platform handles both the operation and testing phases.

Through this platform, users can access and manage all tools currently operational within the TELEMETRY ecosystem. From the service point of view, the platform applies rules based on the methodologies created during the project, to implement flows necessary for the administrator to validate the security level of his installation. To this direction according to output parameters of each tool the next necessary tool is invoked, the parameters are directed to the next tool and overall operation is orchestrated. The platform also features a comprehensive dashboard that allows users to view which TELEMETRY tools are up and running, what is the output status of each one and also control run time of the tools (e.g., start, stop).

## 4.3 Technology Considerations

### 4.3.1 Best Practice

Security tools are essential to protect an organization's data, systems, and networks from various threats, including cyber-attacks, data breaches, and other vulnerabilities. Their purpose is to ensure the confidentiality, integrity, and availability of organizational assets while complying with legal and regulatory requirements. Effectively deploying security tools demands a strategic approach tailored to an organization's specific and evolving security needs. Best practices entail a comprehensive assessment of system vulnerabilities, and the selection of appropriate security solutions to mitigate these risks. It is essential that these tools are adaptable, capable of integrating new technologies and scaling with the organization as it grows. Key strategies include regular security assessments, implementing a multi-layered defence, and focusing on protecting critical assets. To maintain effectiveness, benchmarking against industry standards helps identify areas for improvement. It is crucial to integrate these tools seamlessly into the current operational infrastructure, that includes all tools and systems actively employed by the organization, without causing disruptions. Regular updates and effective patch management are essential to guard against new threats. Additionally, training staff to recognize and appropriately respond to security risks significantly boosts the effectiveness of the security measures in place. This comprehensive approach not only protects but also fortifies an organization’s security posture in a constantly changing threat landscape.

### 4.3.2 Containerisation

Containerization is a powerful technology that packages software and all its dependencies into isolated units known as containers. Each container functions independently, ensuring that they do not interfere with one another, which facilitates consistent operation across varied computing environments. This isolation not only simplifies the deployment process and enhances the scalability of applications but also significantly boosts security measures. By segregating applications, containerization reduces the risk of systemic failures or security breaches spreading from one container to another. Moreover, it allows security tools to operate within these isolated environments, which prevents them from impacting other parts of the system. This setup not only supports easier updates and scalable responses to evolving security challenges but also helps maintain the integrity of applications by clearly segregating duties within the containers.

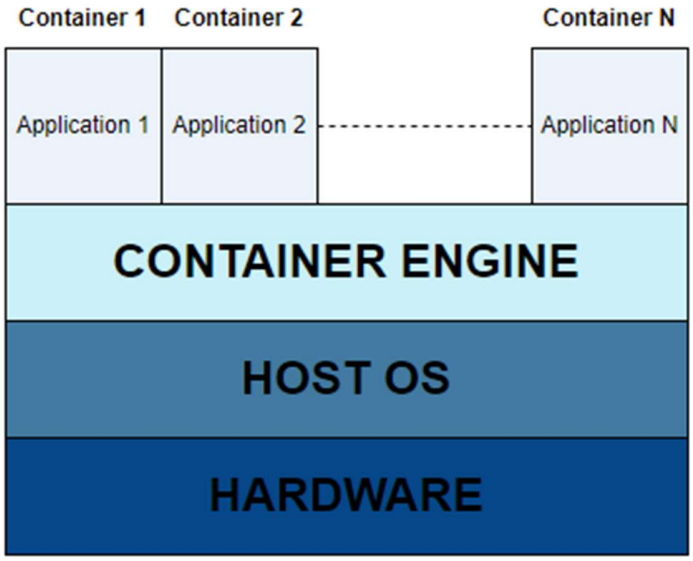



Figure 12: Generic Architecture of Containerisation

Figure 12 illustrates the generic architecture of containerization. Containers are managed by a container Engine, which operates atop the Host Operating System and system's hardware. Containerization enables multiple applications to run independently in isolated environments on the same physical server. This setup enhances efficiency and security by ensuring that each container has access only to the resources it needs, preventing interference and resource conflicts between applications.

Docker containers will be used for TELEMETRY tools. Docker is well-suited for such tasks because it offers lightweight, consistent environments for applications, regardless of the underlying infrastructure. This compatibility ensures that Docker is a highly efficient choice for deploying applications that need to operate across different systems without issues. Moreover, Docker's widespread adoption and robust community support make it a reliable and well-documented solution for containerization needs, further proving its effectiveness in managing and deploying isolated, secure environments.

### 4.3.3 Operation

Each tool will be containerized to ensure it operates independently from other tools and the system environment. This separation allows for easier management and updates. Containers provide functionalities that enable tools to access system parameters such as resources, files, etc., without limitations, ensuring they function optimally within their designated environments. Additionally, this setup enables tools to exchange data among themselves efficiently, enhancing overall system integration and functionality.

### 4.3.4 Security (Securing Communications)

When Docker was introduced in 2013 it brought us the modern era of the container and ushered in a computing model based on microservices. Since containers do not depend on the operating system, they facilitate the development of loosely coupled and scalable

microservices by allowing teams to declaratively package an application, its dependencies, and configuration together as a container image.

Yet as applications grew in complexity to hold containers distributed across numerous servers, challenges arose, including how to coordinate and schedule multiple containers, how to enable communications between containers, how to scale container instances, and more. Kubernetes was introduced as a way to solve these challenges.

To provide a further level of security, a Development, security and Operations tool (DevSecOps tool) that enables easy and secure orchestration deployment of the TELEMETRY toolkit in production facilities will be developed. This tool will be built on top of Kubernetes, which will be extended to address cybersecurity issues including container image scanning and recommendations for the optimum host operating system hardening with sufficient controls to limit system calls and file system access. Kubernetes will also help in automating the deployment and revisions to the target platform having the ability to manage multiple versions of software.

Further, with a focus on security, Kubernetes and the target platform are sandboxed by default and built to allow a secure connection using the Transport Layer Security protocol (TLS) between services where it can monitor and maintain the connections between them.

# 5 TELEMETRY Tools

The TELEMETRY project research develops and applies tools to analyse risks and detect security issues. This chapter gives a brief description of all the TELEMETRY tools.

The TELEMETRY tools address the following Key Exploitable Results (KER):

### Table 6: KER Addressed by TELEMETRY Tools

| ID | Description of KER | Source | Owner |
|---|---|---|---|
| 01 | System Security Modeler Knowledge & Tooling Extensions | DoA "Key Exploitable Result (KER)" | UoS |
| 02 | Cryptographic protocols for firmware updates and device access control | DoA "Key Exploitable Result (KER)" | KUL |
| 03 | Methods and Tools for the protection of Telecom network infrastructures | DoA "Key Exploitable Result (KER)" | TIM |
| 04 | Distributed-Ledger-Technology (DLT)-based Data Space for data sharing | DoA "Key Exploitable Result (KER)" | MTU |
| 05 | API Trust and Security Analyzer | DoA "Key Exploitable Result (KER)" | MTU |
| 06 | ML framework for software misuse detection | DoA "Key Exploitable Result (KER)" | i4RI |
| 07 | Secure deployer tool | DoA "Key Exploitable Result (KER)" | i4RI |
| 08 | IoT fuzzer | DoA "Key Exploitable Result (KER)" | SINTEF |
| 09 | Digital Twin virtualized test environment | DoA "Key Exploitable Result (KER)" | SINTEF |
| 10 | Time Series Processor – Auto Model Builder pipeline (TSP-AMB pipeline) enhancement | DoA "Key Exploitable Result (KER)" | NOKIA |
| 11 | Importer Service for the TSP | DoA "Key Exploitable Result (KER)" | NOKIA |

| | | | |
|---|---|---|---|
| 12 | Real Time Cybersecurity Monitoring<br>Real-time monitoring security and privacy vectors | DoA "Key Exploitable Result (KER)" | ATC |
| 13 | TELEMETRY Integrated Toolkit Framework | DoA "Key Exploitable Result (KER)" | ATC, all |

## 5.1 Register of TELEMETRY Tools

The TELEMETRY Tools are intended to work synergistically to analyse detect security issues and the risks arising from them. Different technologies are provided by different partners, with each being based on their typical methodologies and assumptions. In TELEMETRY these differences are identified and aligned, so that compound statements can be given.

For example, vulnerabilities scanner and Fuzzers, which systematically probe systems operate in a testing setting. While network anomaly and Intrusion Detection Systems (IDS) monitor traffic during operation. The latter assumes swift real-time reactions, while the former may include time-intensive operations and provide results in non-real-time.

Finally, both systems are integrated into the dashboard provided by ATC and the risk assessment system, Spyderisk provided by UoS, in TELEMETRY.

Processes include integration to outside information, such as threat intelligence feeds, anomaly model training and re-evaluation or re-starting testing phase insights. Interfaces, which include timing, protocol and semantics need to be defined, which may lead to standards for a good and open security toolset framework.

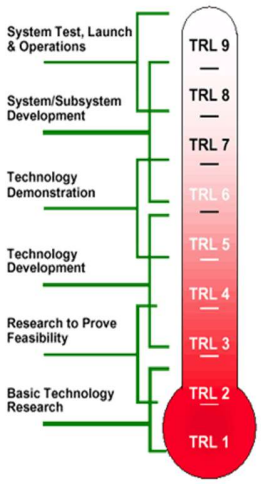
TELEMETRY tools, each and in joint effort aim to play a pivotal role in safeguarding digital assets, such as those defined in use cases, and mitigate security risks in a dynamic and evolving environment.

In the following subsections, the list of TELEMETRY tools is presented, followed by a tabular detailed description of each of them. The Table 7 includes the following information:

**Table 7: Tool Template with Description**

| | |
|---|---|
| **Tool Name:** | Name |
| **Purpose:** | Purpose of the tool – what it is intended to do |
| **Features and Benefits:** | Highlighting the tools unique value proposition |
| **Uniqueness:** | What is unique about this tool |
| **Provided Upstream Interfaces (API):** | Interfaces that are provided toward other TELEMETRY TOOL |
| **Required Downstream Services (API):** | Information about interfaces provided by the tool |
| **Input:** | What kind of input digests the tool |

| | |
|---|---|
| **Semantic Description of Input:** | About the semantics, so to understand consequences and requirements |
| **Output:** | What kind of output does the tool produce |
| **Semantic Description of Output:** | Semantic |
| **How can the functionality be evaluated:** | Information to facilitate the evaluation of the tool |

| | |
|---|---|
| **Current TRL:** | The Technology Readiness Level[3] (TRL), gives an indication of how much research component is included and how close to market that tool is.<br>  <br><br>Figure 13: Representation of TRL Levels<br>See also:<br>https://en.wikipedia.org/wiki/Technology_readiness_level |

### 5.1.1 SIEA Pipeline

The SIEA Pipeline consists of three applications: the Aggregator, the Distributor and the SIEA (Security Information and Event Acquisition). This inseparable chain of applications ingests events only at the Aggregator. The final aggregated, prioritized and filtered events are forwarded via the SIEA to the SSM. The SIEA is aware of the status of the SSM and controls the

[3] Figure by NASA/Airspace Systems (AS) - http://as.nasa.gov/aboutus/trl-introduction.html at the Wayback Machine, Public Domain, https://commons.wikimedia.org/w/index.php?curid=65946549

downstream message flow by polling this status. If the message flow is stalled, new incoming events are queued.

For testing purposes, the SIEA Pipeline listens to a specific Kafka message which triggers resetting the SSM and itself. It also uses representational state transfer REST interface to access e.g. the SSM.

**Table 8: SIEA Pipeline**

| | |
|---|---|
| **Tool Name:** | Security Information & Event Acquisition Pipeline |
| **Purpose:** | Collects, aggregates, filters, and prioritizes security-related events and vulnerabilities. Connects to the SSM via RESTful southbound interface for further processing. |
| **Features and Benefits:** | The SIEA pipeline subscribes to a Kafka broker which publishes events from upstream tools without changing their order. These events are pushed into three queues of different priorities, i.e. high, medium and low. Less prior events can be 'overtaken' by higher prior events. Example: Low-prior events are only forwarded downstream, if the high and medium queues are empty.<br>In order not to flood the downstream tools, the SIEA pipeline groups, aggregates and, in some cases, suppresses recurring events.<br>The SIEA pipeline controls the flow of events to the SSM via RESTful handshake. |
| **Uniqueness:** | The SIEA Pipeline operates in real-time and offers other upstream tools a controlled connection to the SSM. |
| **Provided Upstream Interfaces (API):** | **Name:** Aggregator-Distributor-SIEA<br>**Description:** single point of entry @Aggregator for security-related events and vulnerabilities (e.g.: detected threats, anomalies, and ML inference insights).<br>**Parameters:** JSON Package delivered via Kafka or RESTful message which contains a description of the event with information on where and when it was detected, and which node is impacted. Potentially also additional information on the severity of the threat.<br>**Return value:** 'OK' or 'ERROR'<br>**Used by:** Nokia's Anomaly Detection, the DTL from MTU, and other TELEMETRY tools |
| **Required Downstream Services (API):** | **Name:** Aggregator-Distributor-SIEA<br>**Description:** single point of exit @SIEA. Pushes the aggregated results of the SIEA to the SSM |

| | |
|---|---|
| | **Parameters:** JSON Package transmitted via RESTful message which contains a description of the event with information on where and when it was detected, and which node is impacted. Potentially also additional information on the severity of the threat.<br>**Return value:** 'FINISHED' or 'FAILED'<br>**Provided by:** Threat Diagnosis (SSM) |
| **Input:** | Security related events (from e.g.: Wazuh Server, Snort, BACON, …) |
| **Semantic Description of Input:** | The Aggregator listens to a set of Kafka topics, each of which is dedicated to an upstream tool. The semantic descriptions of the JSON messages which are injected by the upstream tools must be provided by the tools themselves. |
| **Output:** | Prioritized, aggregated and filtered events (to e.g.: the SSM and DLT based Data Sharing) |
| **Semantic Description of Output:** | The semantic output of the SIEA Pipeline is dependent on the upstream tools. The pipeline only adds additional information to the JSON e.g. its priority, a potential repetition count and additional fields which the SSM might require. |
| **How can the functionality be evaluated:** | In near real-time an anomaly, reported by an upstream tool, must be correctly ingested by the SSM. |
| **Current TRL:** | 4 |

### 5.1.2 Network Fuzzer

The network fuzzer facilitates security testing of network interfaces, by assisting with the detection of unknown vulnerabilities. The tool achieves this by sending a large amount of specifically crafted requests to the interface under test, and observing whether it responds or behaves in an unexpected way. Such unexpected behaviour can indicate the presence of a vulnerability, which an analyst in turn can investigate further.

Table 9: Network Fuzzer

| | |
|---|---|
| **Tool Name:** | Network fuzzer |
| **Purpose:** | The network fuzzer facilitates security testing of network interfaces, by assisting with the detection of unknown vulnerabilities. |

| | |
|---|---|
| **Features and Benefits:** | The main benefit of the tool is that allows for an automatic detection of unexpected behaviour. By using this as a starting point, security analysts can greatly improve the efficiency of searching for new vulnerabilities. |
| **Uniqueness:** | The tool builds on the open source boofuzz, which provides a well-documented framework for network fuzzing. We intent to expand on this primarily by<br>- Enabling the tool to generate test cases based on network captures from the interfaces of interest, reducing the need for manual configuration and set up.<br>- Implementing a standardized way of delivering/presenting test results. |
| **Provided Upstream Interfaces (API):** | N/A<br><br>Since a fuzz test will be initiated by a human, the output can be collected afterwards. We do not foresee the need for a continuously running API service. |
| **Required Downstream Services (API):** | N/A<br><br>As fuzz testing will likely not run continuously, providing a network capture file manually should be sufficient. |
| **Input:** | Packet capture (.pcap) file |
| **Semantic Description of Input:** | A .pcap file from the device or interface to be tested. This file shows the packets and protocols involved with communication to the interfaces which we would like to test. |
| **Output:** | .pcap files and text files. The output provides a set of tuples consisting of a description of the unexpected behaviour and a packet capture file of the communication needed to reproduce it. |
| **Semantic Description of Output:** | The output of the tool should be a set of tuples, containing a description of the observed behaviour and an associated packet capture file with the packets required for reproducing the reported behaviour. |
| **How can the functionality be evaluated:** | The functionality of the tool can be evaluated by setting up a device with intentionally vulnerable interfaces and testing the fuzzer against those interfaces. |

| | |
|---|---|
| **Current TRL:** | TRL 4 |

### 5.1.3 Digital Twin

A Digital Twin acts as an infrastructure tool designed for conducting varied security tests on IoT devices in an isolated environment, with a primary focus on assessing the security of their software components. It utilizes a firmware file as input to emulate the capabilities of the IoT device, with a specific emphasis on system and configuration files. Its main objective is to assist in identifying vulnerabilities related to firmware, IoT device configuration, and system OS components.

**Table 10: Digital Twin**

| | |
|---|---|
| **Tool Name:** | Digital Twin |
| **Purpose:** | Provides an emulation environment, allowing IoT developers to create an isolated testing platform and test the software stack including the operating system and device drivers. |
| **Features and Benefits:** | The tool offers benefits as a virtualized infrastructure, enabling other security tools and testing (for example, SBOM) to operate without the need for physical hardware. |
| **Uniqueness:** | The tool is constructed using the open-source quick emulator (QEMU) package, which offers a versatile framework for emulating diverse hardware platforms. We aim to enhance this framework by:<br>• Facilitating the tool's ability to effortlessly replicate IoT device configuration files.<br>• Integrating vulnerability scanners to detect and analyse security vulnerabilities effectively. |
| **Provided Upstream Interfaces (API):** | **N/A** |
| **Required Downstream Services (API):** | **N/A** |
| **Input:** | IoT device firmware (unencrypted) |
| **Semantic Description of Input:** | Firmware is a form of software that is integrated directly into hardware devices, allowing them to manage their operations and functionalities. This file can be obtained by downloading it from the IoT vendor's |

| | |
|---|---|
| | servers or by extracting it from the device using reverse engineering methods. |
| **Output:** | N/A as provides infrastructure to test |
| **Semantic Description of Output:** | N/A |
| **How can the functionality be evaluated:** | The capability to emulate IoT device firmware and conduct security assessments to detect vulnerabilities within the environment. |
| **Current TRL:** | 4 |

### 5.1.4 BACON

The solution proposed implements an Intrusion Detection System (IDS) at runtime on network traffic, using the anomaly detection approach. The approach is based on the Federated Learning paradigm. The BACON tool is composed of (i) a Federated Learning framework able to detect anomalies and/or potential zero-day attacks, (ii) an Alert Graphical User Interface (GUI), based on ELK (Elasticsearch, Logstash and Kibana) stack, to visualize the alerts generated on anomalies.

**Table 11: BACON**

| | |
|---|---|
| **Tool Name:** | BACON-server |
| **Purpose:** | To train and execute a set of FL based models to detect anomalies in network traffic data based on behavioral patterns identified in historic usage scenarios such as normal activities of devices and infrastructure. |
| **Feature:** | Server side of the Anomaly detection tool for network traffic data based on federated learning approach, which starts the learning phase and receives the alerts from the BACON-clients deployed on IoT devices/edges |
| **Uniqueness:** | Application of Federated learning approach (i) trains models locally, on client devices, while maintaining user **data privacy**, (ii) creates **robust models** based on data from various resources hence features and patterns, (iii) fosters **collaborative security** on device level |

| | |
|---|---|
| **Provided Upstream Interfaces (API):** | **Name:** BACON-server-u01<br>**Description:** entry point for model weights received by BACON-clients<br>**Parameters:** JSON Package transmitted via KAFKA or RESTful message which contains a description of the event detected<br>**Return value:** 'OK' or 'ERROR'<br>**Used by:** BACON-clients<br><br>**Name:** BACON-server-u02<br>**Description:** entry point for alerts or anomaly data received by BACON-clients, to be displayed in the GUI and/or forwarded to other TELEMETRY tools<br>**Parameters:** JSON Package transmitted via KAFKA or RESTful message which contains a description of the event detected<br>**Return value:** 'OK' or 'ERROR'<br>**Used by:** BACON-clients<br><br>**Name:** BACON-server-u03<br>**Description:** entry point for training network data<br>**Parameters:** pcap or comma separated value (csv) transmitted via REST full<br>**Return value:** 'OK' or 'ERROR'<br>**Used by:** use case owner |
| **Required Downstream Services (API):** | **Name:** BACON-server-d01<br>**Description:** single point of exit for BACON-server, forwarding anormal event data<br>**Parameters:** JSON Package transmitted via KAFKA or RESTful message which contains a description of the event detected<br>**Return value:** 'OK' or 'ERROR'<br>**Provided by:** Security Information & Event Acquisition Pipeline and/or Threat Diagnosis (SSM) |
| **Input:** | pcap or real-time network traffic |
| **Semantic Description of Input:** | The input for the tool will be network traffic data |

| | |
|---|---|
| **Output:** | JSON |
| **Semantic Description of Output:** | The output could be a JSON file collecting alerts of anormal event detected. |
| **How can the functionality be evaluated:** | Training and labelled test datasets are needed to verify the accuracy of the solution: test dataset must include anomalies (in relation to the training dataset) or potential anomalies already classified |
| **Current TRL:** | 2 |

### 5.1.5 Misuse Detection ML Toolkit

A set of runtime libraries with ML models trained to detect the misuse of software components & systems based on baseline behavioural patterns identified in historic usage scenarios such as normal activities of users and/or similar patterns on log files or data storages. The ML will learn user-interaction models and the detection of divergences in user behaviour from the norm, using similar principles to social engineering for capturing user aspects such as user functional footprint, temporal behaviour and statistical data distribution.

**Table 12: Misuse Detection Toolkit**

| | |
|---|---|
| **Tool Name:** | Misuse Detection ML Toolkit |
| **Purpose:** | To train and execute a set of ML models to detect the misuse of software components & systems based on baseline behavioral patterns identified in historic usage scenarios such as normal activities of users and/or similar patterns on log files or data storages. |
| **Feature:** | Toolkit which allows users with some Data Scientist knowledge to train their own AI/ML models and deploy them in a central platform ready to be incorporated into either bigger and wider applications as a submodule or as a standalone callable service.<br>It is composed of three main modules:<br>- *Model trainer*: allows training of AI/ML models<br>- *Model manager*: allows to deployment of trained models<br>- *Execution runtime*: allows the execution and serving of deployed trained models |
| **Uniqueness:** | The Toolkit allows users with little or no analytical knowledge to train their own models by selecting the algorithms present in its library. |

| | |
|---|---|
| **Provided Upstream Interfaces (API):** | **Name:** Misuse-ML-Toolkit-up01<br>**Description:** entry point to receive sensor readings so that these can be analysed<br>**Parameters:** JSON Package transmitted via KAFKA or RESTful message which contains sensor readings<br>**Return value:** 'ACK' or 'ERROR'<br>**Provided by:** IoT sensors at pilot sites<br><br>**Name:** Misuse-ML-Toolkit-up02<br>**Description:** upload external AI/ML models in the form of a ZIP/RAR file containing the AI/ML model in the form of a pickle file<br>**Parameters:**<br>- *modelFile*: ZIP/RAR file with the model<br>**Return value:** JSON file containing the results of the validation performed on the uploaded model<br>**Used by:** Any UC owner that has their own AI/ML model already trained<br><br>**Name:** Misuse-ML-Toolkit- up03<br>**Description:** execute a given model with specific data<br>**Parameters:**<br>- *modelId*: ID of the model to obtain the list of the executions and its execution data<br>- *inputData*: JSON with a collection of predictor-value pairs<br>**Return value:** JSON file with the numeric value returned with the predictions as well as translated texts, plots and unified resource locator (URL)<br>**Used by:** UC owners |
| **Required Downstream Services (API):** | **Name:** Misuse-ML-Toolkit-dw01<br>**Description:** list available trained AI/ML models<br>**Parameters:**<br>- *targetOrganisation*: abbreviation of the target pilot for which we want to list the models. Possible values are: NOKIA, ANT, TIM<br>**Return value:** JSON file with models listed with a series of parameters |

| | |
|---|---|
| | **Used by:** UC owners<br><br>**Name:** Misuse-ML-Toolkit- dw02<br>**Description:** obtain info and predictors of the chosen model<br>**Parameters:**<br>- *modelId*: ID of the chosen model<br>**Return value:** JSON file specifying the predictors for the trained model. In addition to the name of the predictor, in the case of text-based information, it will include the possible textual values as well as their correspondence to a numeric value<br>**Used by:** UC owners<br><br>**Name:** Misuse-ML-Toolkit- dw03<br>**Description:** publish anomalies detected by the AI/ML model executed identified by the execution ID passed as a parameter<br>**Parameters:**<br>- *executionId*: ID of the executing model<br>**Return value:** An array with the anomalies detected for the AI/ML model identified by the execution ID passed<br>**Used by:** Downstream components<br><br>**Name:** Misuse-ML-Toolkit- dw04<br>**Description:** obtain the URL of the generated portable network graphic (PNG) file for the execution passed as a parameter<br>**Parameters:**<br>- *executionId*: ID of the executing model<br>**Return value:** An array with all the names of the PNG files generated for the execution ID passed<br>**Used by:** UC owners<br><br>**Name:** Misuse-ML-Toolkit- dw05<br>**Description:** list the executions performed for the model passed as a parameter<br>**Parameters:**<br>- *modelId*: ID of the model to obtain the list of the executions and its execution data |

| | |
|---|---|
| | **Return value:** JSON file listing the executions performed for the requested model<br>**Used by:** UC owners |
| **Input:** | Depending on the tool being executed, the input can be of three types:<br>- *Model trainer*: JSON-based or CSV-based datasets<br>- *Model manager*: Pickle file<br>- *Execution runtime*: JSON-based or CSV-based data stream |
| **Semantic Description of Input:** | Depending on the tool being executed, the description of input is as follows:<br>- *Model trainer*: The input for training any AI/ML model will be sensor-related data from the use case devices<br>- *Model manager*: Trained model by an external tool<br>- *Execution runtime*: The input for executing any AI/ML model will be sensor-related data from the use case devices |
| **Output:** | Depending on the tool being executed, the output can be of three types:<br>- *Model trainer*: Pickle file<br>- *Model manager*: 'Model uploaded' or 'Validation errors'<br>- *Execution runtime*: JSON-based output |
| **Semantic Description of Output:** | Depending on the tool being executed, the description of output is as follows:<br>- *Model trainer*: The trained model with the Model Trainer<br>- *Model manager*: Whether an external model has been successfully uploaded or the validation routine has encountered errors<br>- *Execution runtime*: Alerts referring to the anomalies detected |
| **How can the functionality be evaluated:** | Since the toolkit offers two types of results, the functionality can be evaluated in two ways:<br>- For the AI/ML models: by training as many models with different datasets as necessary to assess its functionality<br>- For the anomalies alerts: by having a validation dataset that will assess the accuracy of the trained models |
| **Current TRL:** | 5 |

### 5.1.6 Data Space - DLT based Data Sharing

A DLT based data sharing and persistent storage for data assets. The tool is based on an existing platform known as SmartQC which facilitates the definition of context and metadata structures that can be extended and validated prior to being committed to an immutable ledger layer. In the context of TELEMETRY, the tool can be used to create immutable, auditable records of events and reports that are generated by other tools.

**Table 13: Data Space - DLT based Data Sharing**

| | |
|---|---|
| **Tool Name:** | Data Space |
| **Purpose:** | Data sharing platform based on DLT and Smart Contracts, for maintaining an immutable, auditable record of events, alerts and reports generated by the various security and anomaly detection tools. |
| **Features and Benefits:** | The tool offers the benefits of DLT, in particular immutability of records, while abstracting from its complexity by featuring a data sharing API on top that is accessible by other tools in TELEMETRY. |
| **Uniqueness:** | The tool provides a JSON API on top of an underlying DLT platform, offering an abstraction layer that facilitates the interaction with the data-sharing platform through relatively simple JSON messages while making use of DLT features such as immutability in order to create reliable, auditable records. The abstraction layer further allows for flexibility in the choice of underlying DLT based on the features that are required, instead of being limited to one particular platform. |
| **Provided Upstream Interfaces (API):** | **Name:** Data Sharing Client API<br>**Description:** JSON API that provides an abstraction of the underlying DLT platform, facilitates the submission of reports (e.g., events, alerts) from tools to the ledger and read access for applications (e.g., mobile applications, dashboards) and tools to query the data that is stored in the DLT<br>**Parameters:** context, data (e.g. test reports), timestamp<br>**Return value:** confirmation or error message<br>**Used by:** SIEA Pipeline, Security testing tools |
| **Required Downstream Services (API):** | **Name:** Data Sharing Client API<br>**Description:** JSON API that provides an abstraction of the underlying DLT platform, facilitates the submission of reports (e.g., events, alerts) from tools to the ledger and reads access for applications (e.g., mobile applications, dashboards) and tools to query the data that is stored in the DLT |

| | |
|---|---|
| | **Parameters:** context of requested data<br>**Return value:** data, metadata<br>**Provided by:** SSM, Trust and Security Analyser, dashboard<br><br>**Name:** Notification interface<br>**Description:** Interface onto which notifications can be sent to inform about received updates<br>**Parameters:** either a generic indicator that something has been received or the received update itself<br>**Return value:** Acknowledgment<br>**Provided by:** SSM |
| **Input:** | Security events and reports |
| **Semantic Description of Input:** | In the testing stage, events and reports will be received directly from testing tools. In the operation stage, aggregated reports will be received from the SIEA pipeline. |
| **Output:** | Notifications, query access for applications and tools |
| **Semantic Description of Output:** | Through the Client API, tools such as the SSM or the Trust and Security Analyser can query the data-sharing platform for reported data. Notifications will be triggered by incoming reports in the testing stage and will be sent to inform the SSM of new content. |
| **How can the functionality be evaluated:** | Submission of reports to the data sharing platform, querying of content and sending reports that should trigger notifications. |
| **Current TRL:** | 4 |

### 5.1.7 API Trust and Security Analyzer

The Trust and Security Analyser models and assesses the trustworthiness of SUT components based on the outputs of security testing tools.

**Table 14: API Trust and Security Analyzer**

| | |
|---|---|
| **Tool Name:** | API trust and Security Analyzer |

| | |
|---|---|
| **Purpose:** | The Trust and Security Analyser conducts a trustworthiness assessment of the components of the SuT, based on reports from tools such as the anomaly detection tools that are available in the DLT data space. This serves to identify components whose trustworthiness is compromised. |
| **Features and Benefits:** | Create a trustworthiness score of SUT components/devices based on security testing reports. This score can be used to identify problematic devices and can serve as additional input to risk models. The Analyser may include an API scanning component that actively probes the SUT for known vulnerabilities. |
| **Uniqueness:** | The Trust and Security Analyser creates trustworthiness assessments based on reports from security testing tools. It aggregates the reported metrics and alerts into a trust model, and it uses the score obtained through that model to identify if any components or devices of the SUT are acting in a manner that is deemed not trustworthy. |
| **Provided Upstream Interfaces (API):** | **Name:** Data Sharing Client API<br>**Description:** The Trust and Security Analyser pulls test reports from the DLT based Data Sharing<br>**Parameters:** context of requested data<br>**Return value:** data, meta-data<br>**Used by:** DLT based Data Sharing |
| **Required Downstream Services (API):** | **Name:** Data Sharing Client API<br>**Description:** Trust assessments are reported back to the DLT based Data Sharing, API scanner reports test results<br>**Parameters:** context, trust values, test results from API scanner<br>**Return value:** confirmation or error message<br>**Provided by:** DLT based Data Sharing |
| **Input:** | Events and alerts from security testing tools |
| **Semantic Description of Input:** | The Trust and Security Analyser queries the DLT based Data Sharing for events and alerts that were reported, in order to use them as input for trust assessment. |
| **Output:** | Trustworthiness assessments, test results from API scanner |

| | |
|---|---|
| **Semantic Description of Output:** | Trustworthiness assessments that assign trust values to components of the SUT are reported back to the DLT based Data Sharing. |
| **How can the functionality be evaluated:** | Reported anomalies should lead to reduced trustworthiness scores in relation to the affected components. |
| **Current TRL:** | 4 |

### 5.1.8 Cyber Range

The cyber range is a platform for the development, delivery and use of interactive cybersecurity hybrid environments that can include both virtual and physical components. This tool can be used for testing, scenario development, data extraction, attack simulation, etc. In the context of TELEMETRY, it can be used to generate simulated network attacks as test scenarios for the anomaly detection tools in the TELEMETRY use cases.

**Table 15: Cyber Range**

| | |
|---|---|
| **Tool Name:** | Cyber Range |
| **Purpose:** | Cyber range is a platform for the development, delivery and use of interactive cybersecurity hybrid environments |
| **Features and Benefits:** | The Cyber Range is a platform with a variety of features around cyber security testing. One of the features that can contribute to the work in TELEMETRY's use cases is the generation of a variety of simulated traffic and attack patterns that can be applied to an SUT. |
| **Uniqueness:** | The wide range of cyber security testing features makes the Cyber Range a somewhat unique piece of testing infrastructure. Even though it will not be physically moved to one of the Use Case testbeds, it is envisaged that it can connect to them remotely via suitable methods such as VPN and can perform tests through that remote connection. |
| **Provided Upstream Interfaces (API):** | N/A |
| **Required Downstream Services (API):** | **Name:** SUT access<br>**Description:** Network access to the SUT<br>**Parameters:** N/A |

| | |
|---|---|
| | **Return value:** N/A<br>**Provided by:** UC owners |
| **Input:** | Specification of the SUT |
| **Semantic Description of Input:** | A description/specification of the SUT is needed to create traffic patterns that simulate potential attacks. |
| **Output:** | Simulated network attack traffic |
| **Semantic Description of Output:** | The Cyber Range exposes the SUT to simulated attack traffic in order to be able to investigate the SUT's response and the detection by anomaly detection tools. |
| **How can the functionality be evaluated:** | Traffic is being sent to the SUT, with the SUT and anomaly detection tools responding accordingly. |
| **Current TRL:** | 4 |

### 5.1.9 Spyderisk System Modeller - Risk Assessment

The Spyderisk System Modeller (SSM) is a comprehensive automated risk management toolkit designed to enhance a system's security via the assessment of risks and recommendations of controls to lower the likelihood of risks with an unacceptably high level. SSM enables users to construct detailed system models (models of the system under test), identify cybersecurity and compliance risks, and determine the most effective mitigations. Its user interface is shown below in Figure 14. A panel of asset icons is shown at the left, from which a system model can be constructed in the central panel, and at the lower right individual risks (named Consequences) are shown. Each risk can be explored to determine threats that cause it, and controls are recommended by the tool. The controls can be selected and the risk levels re-calculated to see the effect of the controls on the risks.

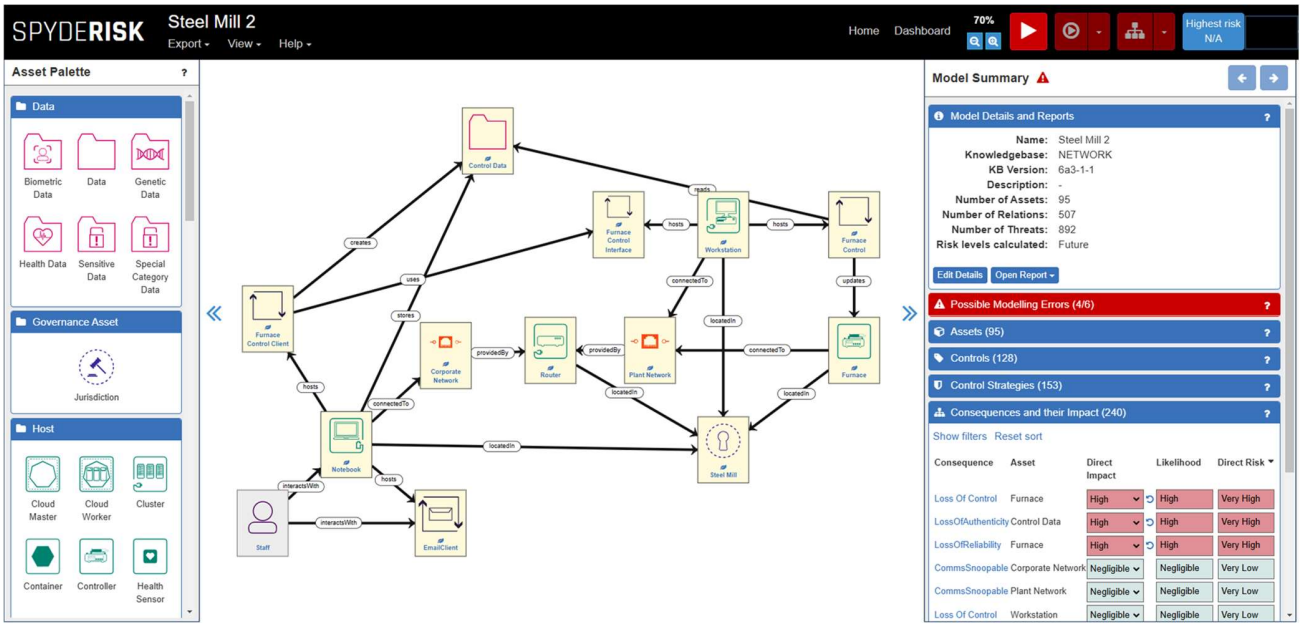



Figure 14: UoS Spyderisk User Interface

The SSM also supports dynamic cybersecurity risk assessments that adapt to new threats as they are detected by tools like vulnerability scanners (e.g., Wazuh, OpenVAS), plus other tools developed in TELEMETRY such as anomaly detection, fuzzing or the event pipeline. This adaptive feature enables the assessment of emerging risks at runtime along with by dynamically recommending and implementing necessary security controls, so they be addressed in a timely manner, enabling the maintenance of a robust defence.

Table 16: Risk Assessment (Spyderisk)

| | |
|---|---|
| **Tool Name:** | Spyderisk System Modeller (SSM) |
| **Purpose:** | The Spyderisk System Modeller (SSM) is a comprehensive automated risk management toolkit designed to enhance a system's security via the assessment of risks and recommendations of controls. |
| **Purpose, Features and Benefits:** | To lower the likelihood of risks with an unacceptably high level. It is a knowledge-based risk assessment tool enabling system-level risk assessment at both design time and runtime. SSM system model concepts are compliant with the standard 27000/27005 of the international standards organization (ISO). SSM enables users to construct detailed system models (models of the system under test), identify cybersecurity and compliance risks, and determine the most effective mitigations. |
| **Uniqueness:** | Spyderisk enables modelling and analysing risk in information systems, based on ISO 27005, part of the ISO 27000 family for information |

| | |
|---|---|
| | security. It is at TRL6 and has been in development for 9+ years. It is knowledge-based, whereby knowledge of vulnerabilities, threats, risks and controls to manage risk are encoded in a knowledge base. Spyderisk Knowledge base ontologies are generic and designed to support automation using a cause-and-effect approach to risk modelling and in TELEMETRY the knowledge base will be enhanced to accommodate additional cyber threats and risks for IoT devices, plus contributions from other work in the project such as SBOMs to determine internal software structure.<br><br>In use, the user constructs a model of their system under test, and the tool uses knowledge in the knowledge base to identify relevant risks and threats and calculate risk levels. The tool also recommends controls to lower the likelihood of risks. It differs from existing methods of risk assessment in that other methods are largely based on checklists and human judgement, whereas Spyderisk is a simulator of the causes and effects of cyber threats in interconnected systems.<br><br>Dynamic context and threat propagation via interdependencies of assets and consequences offer a continuous monitoring and updating of risk assessments as new threats emerge or as changes occur in the system. |
| **Provided Upstream Interfaces (API):** | **Name:** adaptor/notify-immediate-action-event<br><br>**Description:** The SIEA system sends a notification event to SSM, which then adjusts the trustworthiness levels of certain model assets to reflect that event.<br><br>**Parameters:** the webkey of the SSM model corresponding to the live system, and a JSON object representing an event that changes model assets trustworthiness.<br><br>**Return value:** 200 OK<br><br>**Used by:** SIEA<br><br>**Name:** adaptor/notify/report<br><br>**Description:** Notify that one or more scan reports are available for reading and analysis. The SSM Adaptor reads and parses pending reports, converting them to internal state-reports form that will be directly applicable to the system model.<br><br>**Parameters:** the webkey of the SSM model corresponding to the live system.<br><br>**Return value:** is of state report IDs parsed<br><br>**Used by:** Data Space / Dashboard to present to user.<br><br>**Name:** adaptor/state/process |

| | |
|---|---|
| | **Description:** Process state reports and apply changes to the system model<br>**Parameters:** the webkey of the SSM model corresponding to the live system.<br>**Return value:** OK<br>**Used by:** Data Space / Dashboard to present to user.<br><br>**Name:** adaptor/calc-risks<br>**Description:** Calculate system model risks in terms of FUTURE or CURRENT mode.<br><br>**Parameters:** the webkey of the SSM model corresponding to the live system.<br>**Return value:** risk consequences on assets<br>**Used by:** Data Space / Dashboard to present to the user.<br><br>**Name:** adaptor/recommendations<br>**Description:** SSM applies an algorithm to assess risk by identifying consequences and high-likelihood threat paths. Following this analysis, SSM seeks out control strategies and effective controls designed to reduce the likelihood of these paths.<br>These controls form the basics of our recommendations.<br>**Parameters:** the webkey of the SSM model corresponding to the live system.<br>**Return value:** A JSON response containing the ID of the asynchronous recommendation calculation.<br>**Used by:** Data Space / Dashboard to present to the user. |
| **Required Downstream Services (API):** | **Name:** adaptor/recommendations/result<br>**Description:** Download calculated recommendations<br>**Parameters:** the webkey of the SSM model corresponding to the live system.<br>**Return value:** JSON response containing a list of recommendation objects<br>**Used by:** Data Space / Dashboard to present to the user. |
| **Input:** | REST based parameters and JSON data |

| | |
|---|---|
| **Semantic Description of Input:** | Input state reports provide detailed information about system anomalies and vulnerabilities, typically including CVE (Common Vulnerabilities and Exposures) and CVSS (Common Vulnerability Scoring System) metrics. This information should be mapped to corresponding assets in the system model. The result will be an adjusted trustworthiness level for each affected asset, reflecting the potential risk. |
| **Output:** | Recommendation response in a JSON format |
| **Semantic Description of Output:** | The output is a JSON object that presents the current status of the risk model and includes a list of potential recommendations. Each recommendation specifies the controls that need to be implemented and details the expected risk reduction resulting from the implementation of these controls. |
| **How can the functionality be evaluated:** | By experiments around use cases with the end user partners where end users test the realism and accuracy of the risk assessment and the usefulness of the recommendations provided by the tool to lower risks. |
| **Current TRL:** | 6 |

### 5.1.10 SBOM Generator

The SBOM (software bill of materials) generator gives an overview of the software components and libraries included in a software product. This will in turn allow the tool to list known vulnerabilities present in the software product and the vulnerabilities' severity.

**Table 17: SBOM Generator**

| | |
|---|---|
| **Tool Name:** | SBOM generator |
| **Purpose:** | Provides an overview of the software components and libraries included in a software product. |
| **Features and Benefits:** | The purpose of constructing an SBOM is to have an overview of the software components and libraries which are included in a given software product. Having such an overview is a prerequisite for determining which vulnerabilities affect your system, which in turn is essential when performing risk assessments or vulnerability management. |
| **Uniqueness:** | While an SBOM is normally generated during the build process of a software product, this tool seeks to perform the same activity after the software has been released only using the software package or binary file. |

| | |
|---|---|
| **Provided Upstream Interfaces (API):** | **Name:** get components<br>**Description:** Provides a list of identified software components and libraries in a software product.<br>**Parameters:** Software product<br>**Return value:** List of components and libraries<br>**Used by:** SSM<br><br>**Name:** get vulnerabilities<br>**Description:** Provides a list of identified vulnerabilities and severity scores found in a software product.<br>**Parameters:** Software product<br>**Return value:** List of CVEs and CVSS<br>**Used by:** SSM |
| **Required Downstream Services (API):** | **Name:** CVE database<br>**Description:** Service providing an overview of vulnerabilities for a given software product<br>**Parameters:** software product<br>**Return value:** List of CVEs<br>**Provided by:** project external<br><br>**Name:** CVSS database<br>**Description:** Service providing an overview of CVSS scores for CVEs<br>**Parameters:** list of CVEs<br>**Return value:** Associated CVSS scores<br>**Provided by:** project external |
| **Input:** | Binary file/software package |
| **Semantic Description of Input:** | The input to the tool is a binary file or a software package, in the same format as it is delivered by the software development company. |
| **Output:** | SBOM file |

| | |
|---|---|
| **Semantic Description of Output:** | The output of the tool is an SBOM file, although it is likely not complete. Since the tool effectively performs reverse engineering, 100% coverage should not be expected. |
| **How can the functionality be evaluated:** | By building a known project, for instance, an open-source project, we can obtain a ground truth for the SBOM and compare it to the one produced by the tool. |
| **Current TRL:** | TRL 1 |

### 5.1.11 r-Monitoring Tool

The tool aims to enhance system security by providing comprehensive monitoring and analysis of system processes, monitoring metrics, and network traffic.

Table 18: r-Monitoring Tool

| | |
|---|---|
| **Tool Name:** | r-Monitoring Tool |
| **Purpose:** | Comprehensive monitoring and analysis of system processes, monitoring metrics, and network traffic |
| **Features and Benefits:** | Resource Monitoring (Overall Central processing unit (CPU), Rapid Access Memory (RAM) etc. consumption)<br>System Monitoring (Detailed process consumption etc.)<br>File Monitoring (Monitor sensitive files for changes)<br>Hash Monitoring (check hashes for altered processes)<br>Network capturing and Monitoring (Signature monitor)<br>Anomaly Detection on Network Data<br>Provide Historical Data<br>Dashboard Monitoring |
| **Uniqueness:** | The agent designed for collecting metrics, system parameters, and network traffic boasts a lightweight architecture that is compatible with various CPU architectures. Its efficient design makes it especially well-suited for devices with constrained resources, ensuring broad applicability without compromising performance.<br>While several tools on the market provide some of the features listed above, it is rare to find a single tool that encompasses these capabilities comprehensively and even more for devices with constrained resources. |

| | |
|---|---|
| **Provided Upstream Interfaces (API):** | **Name:** r-Monitoring External Service<br>**Description:** External service will be responsible for analysis and provide outcomes.<br>**Parameters:** Service<br>**Return value:** Historical Data, Monitoring, malware analysis outcome, known CVEs of process running, state of sensitive files<br>**Provided by:** ATC |
| **Required Downstream Services (API):** | **Name:** SIEA Pipeline<br>**Description:** Collects, aggregates, filters, and prioritizes security-related events and vulnerabilities.<br>**Parameters:** JSON<br>**Return value:** -<br>**Provided by:** NOKIA<br><br>**Name**: Data Sharing Client API<br>**Description**: JSON API that provides an abstraction of the underlying DLT platform, facilitates the submission of reports (for testing tools)<br>**Parameters:** JSON<br>**Return value:** -<br>**Provided by:** MTU<br><br>**Name**: r-Monitoring Agent<br>**Description**: Agent will provide system metrics<br>**Parameters:** JSON<br>**Return value:** -<br>**Provided by:** ATC |
| **Input:** | System Metrics |
| **Semantic Description of Input:** | Input will be the system metrics provided by the agent |
| **Output:** | alerts, system metrics |

| | |
|---|---|
| **Semantic Description of Output:** | r-Monitoring Tool will provide all system metrics, assessments and alerts if something is wrong |
| **How can the functionality be evaluated:** | r-Monitoring tool can be tested by stressing and testing the monitored device and checking if outcomes metrics are correct |
| **Current TRL:** | 3 |

### 5.1.12 SNORT + Ruleset

The tool utilizes the standard rule-based network intrusion detection software SNORT with a set of advanced rules that are derived from the input of the other TELEMETRY components, namely SBOM and Fuzzer, as well as AI-based internet crawling. The rules consist of both white and blacklist rules and different Ruleset configurations will be tested. The alarm events will be evaluated in conjunction with other TELEMETRY network anomaly detection tools.

**Table 19: SNORT + Ruleset**

| | |
|---|---|
| **Tool Name:** | SNORT + Ruleset |
| **Purpose:** | Generate additional alarm events that are originated in well-grounded root cause. That is SNORT rules are derived from information and aim to detect specific network events. |
| **Features and Benefits:** | Alarm events that are in relation to known malicious network traffic, as well as with the system under tests known open-source components (SBOM) are detected. Alarm events are tagged with a description of how the rule had been derived and why the alarm event is fired. Most of them can be linked to specific known threats. Events of other monitoring tools can become more importance if SNORT rules fire in same timeslot. |
| **Uniqueness:** | The alarm events can be annotated with specific reasons, because the rulesets are not fully AI-generated, but artificial Intelligence (AI) is used to support rule finding.<br>In the whitelist configuration a very strict, setup is achieved, while in the blacklist configuration, each rule is linked to a CVE or reasoning for its creation. |
| **Provided Upstream Interfaces (API):** | **Name:** SBOM Insight<br>**Description:** SBOM reports are taken in from a testing phase and utilized for ruleset generation |

| | |
|---|---|
| | **Name:** Fuzzing Reports<br>**Description:** Fuzzing reports are taken in from a testing phase and utilized for ruleset generation |
| **Required Downstream Services (API):** | **Name:** Alarm Events<br>**Description:** each rule firing will generate an alarm event. The alarm events can in the upstream be used to emphasize or correlate other alarm firings. Vague alarms can become strong alarms. Forensic investigation can be facilitated<br>**Parameters:** Alarm ID, Timestamp, Rule ID<br>**Used by:** SIEA Pipeline |
| **Input:** | Reports of SBOM, Report of fuzzing, and General SNORT rulesets derived by security researchers |
| **Semantic Description of Input:** | The inputs will be digested and used by human experts to generate the rules. |
| **Output:** | Security Events with their root cause |
| **Semantic Description of Output:** | The security events will have information associated that gives **origin**, classification, reasoning and information for forensic experts. |
| **How can the functionality be evaluated:** | By way of comparing the firings to the network anomaly detections, by way of comments of the administrator that sees the event firings and uses the description for forensic. |
| **Current TRL:** | 3 (while SNORT itself has TRL 9, the Ruleset generation is research state) |

### 5.1.13 NOKIA Anomaly Detection Pipeline

The Nokia Anomaly Detection Pipeline (NAD Pipeline) consists of a string of applications which connect to sensor readings reported by devices under test (DUT). The NAD Pipeline consists of the following applications: the Importer Service, the Time Series Processor (TSP), the Automated Model Builder (AMB) and finally the Security Assessment (SA). The task of this pipeline is to report anomalies in near real-time during the operational phase of the DUT.

Table 20: NOKIA Anomaly Detection Pipeline

| | |
|---|---|
| **Tool Name:** | Nokia Anomaly Detection Pipeline |
| **Purpose:** | The Nokia Anomaly Detection Pipeline (NAD) is able to predict sensor measurements of a DuT during live operation. It compares the prediction (target) with the measurement (actual) and makes a decision to report significant deviations in near-real-time to downstream components. |
| **Features and Benefits:** | This tool takes a fingerprint of recorded sensor readings from the DUT operated during normal operation and can detect abnormal situations during live operation. |
| **Uniqueness:** | The Nokia Anomaly Detection Pipeline contains a machine learning application, i.e.: the AMB, which is tuned to predict the values of a chosen key performance indicator (cKPI) based on values of other key performance indicators (KPIs). In this way, the tool is capable of detecting deviations not only from the cKPI but also from the other KPIs which are used to predict the cKPI. |
| **Provided Upstream Interfaces (API):** | **Name:** DUT via Kafka<br>**Description:** The sensor readings of the DUT will be reported to a Kafka broker. An Importer Service subscribes to a Kafka topic which is mapped to reported sensor readings of a DUT.<br>**Parameters:** a Kafka topic<br>**Return value:** see Kafka specification<br>**Used by:** Importer Service |
| **Required Downstream Services (API):** | **Name:** Aggregator via Kafka<br>**Description:** To reduce a flood of reported anomalies, the Security Assessment application triggers only a report to the Aggregator of the SIEA Pipeline if the abnormal condition prevails for a configurable amount of time. The reports are alarms which can escalate from yellow to orange and finally to red. Vice versa, the alarms can de-escalate when the DUT's readings go back to the expected values.<br>**Parameters:** Alarm condition with timestamp<br>**Return value:** see Kafka specification<br>**Provided by:** SIEA Pipeline |
| **Input:** | Sensor readings from the DUT |

| | |
|---|---|
| **Semantic Description of Input:** | Numeric values of measurements from the DUT reported in regular intervals |
| **Output:** | Alarms |
| **Semantic Description of Output:** | A JSON containing information on when an anomaly was detected mapped to an alarm condition (off-yellow-orange-red) which points to how long the anomaly prevails. As soon as the DUT operates normally, the alarm condition is de-escalated (red-orange-yellow-off). The output information also contains timestamps when transitions in alarm conditions occur. |
| **How can the functionality be evaluated:** | Either manipulate the DUT itself or manipulate the sensor readings of the DUT |
| **Current TRL:** | 3 |

### 5.1.14 Service-Level Cybersecurity Testing

A universal and intuitive tool for Service-Level Cybersecurity testing will allow network management (administrator) to improve the quality of decisions made and reduce response time to incidents.

**Table 21: Service-Level Cybersecurity Testing**

| | |
|---|---|
| **Tool Name:** | Service-Level Cybersecurity Testing |
| **Purpose:** | Determining the level of effectiveness of the access control system. |
| **Features and Benefits:** | Testing the cybersecurity of IT systems will allow the network administrator to identify vulnerabilities in restricting access to system components, and will also allow monitoring typical or atypical user behaviour, ensuring response to incidents at the HW, and SW levels (closing user rights, ensuring the safety of logs, sending clusters to quarantine, notifying responsible employees, setting timings, issuing recommendations in case of violation of the conditions or rules for resolving an incident, logging actions when resolving an incident, etc.). |
| **Uniqueness:** | The methodology involves a combination of modern tools and methods of data analysis and information system states in order to determine the risk level of the access control system. The information system is considered from the point of view of system analysis as an interaction of subjects and objects, the relations between which are described by access control |

| | |
|---|---|
| | policies. In the methodology, for the first time, the use of indicators of the vulnerability of objects and monitoring of anomalies in the system is proposed for the assessment of the level of risks of the existing access control system. This approach allows administrators to take into account the real state of objects, based on the system architecture and its vulnerabilities, the change in the system state over time, and adjust access policies based on the level of risks that are assessed using the specified data. The mathematical model is built with the use of fuzzy logic approaches, taking into account the great ambiguity in the description of the state of the system and assessments of its vulnerabilities. Existing tools don't take into account factors and parameters mentioned in WRCVE (T3.3.) methodology. |
| **Provided Upstream Interfaces (API):** | **Name:** SBOM Insight<br>**Description:** SBOM reports are taken in from a testing phase and utilized for CVE code generation for using Common Platform Enumeration (CPE) Dictionary -> Common Vulnerabilities and Exposures (CVEs) - > Common Vulnerability Scoring System, CVSS.<br>**Parameters:** CVE codes, CVSS description<br>**Return value:** list of software component<br>**Provided by:** SBOM<br><br>**Name:** User Activity Monitoring (UAM).<br>User and Entity Behaviour Analytics (UEBA).<br>**Description:** The systems are designed to monitor user activity, and their output data: frequency of access to various system objects, anomalous user behaviour, and analysis of data access, are used as input data to the methodology.<br>**Parameters:** object access frequency, anomaly user's behaviour.<br>**Return value:** FLOAT<br>**Provided by:** no in any UC (external software)<br><br>**Name:** ML<br>**Description:** possible use if the tools allow us to solve the issues solved by the User Activity Monitoring systems.<br>**Parameters:** object access frequency, anomaly user's behaviour.<br>**Return value:** degree of anomaly<br>**Provided by:** ML |

| | |
|---|---|
| **Required Downstream Services (API):** | **Name:** Alarm Events<br>**Description:** The methodology is intended to assess the risk level of an access control system to information objects, taking into account the real architecture of the system and the relationships between objects. The methodology takes system state data obtained from partner tools as input and returns an assessment of the risk level of the access control system.<br>**Parameters:** Alarm ID, Timestamp (for dynamic evaluation), Rule ID<br>**Return value:** Linguistic variables, FLOAT risk assessment<br>**Used by:** UoS |
| **Input:** | Reports of SBOM, Report of UAM system, system administrator checklist |
| **Semantic Description of Input:** | The input data will be used as facts to activate the corresponding rules of the expert system. If appropriate APIs are available and implemented with existing systems, input data can be processed automatically. Some of the input data is filled in manually by the system administrator in accordance with the actual system architecture. |
| **Output:** | Security Events related to their root cause |
| **Semantic Description of Output:** | The security events will be associated with risks of access control policy levels, reasoning risks and recommendations for the system's administrator. |
| **How can the functionality be evaluated:** | The following indicators can be used to evaluate the functionality of this tool:<br>- KPI 1 – accuracy of issuing appropriate rights to system users (%);<br>- KPI 2 – the ability to adapt the access control system to changes in the user profile or other factors (vulnerability levels) (%). |
| **Current TRL:** | TRL-2. The concept of the methodology is formulated, the rules of the expert system are being developed, the input tools and integration possibilities are being studied, the data of the UC are being studied and certain indicators are being modelled |

### 5.1.15 Security Platform (Wazuh Server)

Wazuh is an open-source security platform. We intend to use Wazuh in TELEMETRY for monitoring file systems of devices under test (DUTs) and report tampering attempts to the SIEA Pipeline.

**Table 22: Security Scanner (Wazuh Server)**

| | |
|---|---|
| **Tool Name:** | Wazuh Security Platform |
| **Purpose:** | We will configure a Wazuh Server to report changes in specific directories of a remote host containing software libraries and/or programs. If such changes occur, we will report these events for further processing. |
| **Features and Benefits:** | Wazuh has multiple endpoint security and threat intelligence features. Since this tool is open source, we intend to use it in the project to report security-related events in near real-time. |
| **Uniqueness:** | This tool is capable of monitoring a remote host without having to install an agent on this remote host (Agentless Monitoring). This makes the tool especially interesting in cases where the operating system of the remote host is customized in a way that does not allow any installation of foreign software. |
| **Provided Upstream Interfaces (API):** | **Name:** An SSH connection to the DUT<br>**Description:** A Wazuh Server monitors the file system of a remote host and reports changes in binaries launched by updates and potentially even malware.<br>**Parameters:** see Wazuh documentation<br>**Return value:** see Wazuh documentation<br>**Used by:** Wazuh Server |
| **Required Downstream Services (API):** | **Name:** SIEA Pipeline<br>**Description:** the Wazuh Server does not support Kafka for reporting events. In this project, we will implement a Wazuh plugin that reports to a RESTful Server. This Server must then push these reports into a Kafka channel which can be picked up by the Aggregator of the SIEA Pipeline.<br>**Parameters:** The target Kafka topic<br>**Return value:** 200 OK<br>**Provided by:** Wazuh plugin |
| **Input:** | The Wazuh Server monitors the directories of the DUT via SSH connections. There is therefore no specific input. |
| **Semantic Description of Input:** | --- |

| | |
|---|---|
| **Output:** | A security report |
| **Semantic Description of Output:** | An indication of which file changed in which way (size, hash, length, ...) together with a timestamp when the change occurred. |
| **How can the functionality be evaluated:** | Modify a binary on the DUT. |
| **Current TRL:** | > 4 |

### 5.1.16 Secure Software Updates

Perform efficient and secure software updates of the system, utilizing lightweight cryptographic algorithms and novel protocol solutions to develop a new framework for secure patches.

**Table 23: Secure Software Updates**

| | |
|---|---|
| **Tool Name:** | Secure Software Updates |
| **Purpose:** | Perform efficient and secure software updates |
| **Features and Benefits:** | More of an enabler rather than a tool, our solution would enhance and enable the overall architecture of the system by providing fast and efficient software updates to the TELEMETRY architecture, while keeping it secure. |
| **Uniqueness:** | We would develop a novel cryptographic protocol in the scope of this project, to perform secure updates in a very lightweight setting, without a heavy computational and communication overhead. This can easily be patched on to existing systems by a simple firmware update. |
| **Provided Upstream Interfaces (API):** | N/A |
| **Required Downstream Services (API):** | N/A |
| **Input:** | Data generating tools |

| | |
|---|---|
| **Semantic Description of Input:** | Inputs from the various different components and tools which generate data, and parse that data to encrypt it |
| **Output:** | Cryptographic libraries to perform secure software patches and updates |
| **Semantic Description of Output:** | Develop a cryptographic library, to perform the secure patches by just calling a function, and make it easily deployable by firmware updates |
| **How can the functionality be evaluated:** | By either performing implementation benchmarks or testing the formal security of the protocol via formal methods or proofs |
| **Current TRL:** | 3 |

### 5.1.17 Robot Anomaly Detection Tool

Anomaly detection is applied to monitor and analyse robot activity to identify unusual or deviant behaviours. This process involves continuously assessing the robots' operations against established patterns of normal activity. By using explainable methods, it is possible to determine which specific parameter is causing the anomaly.

**Table 24: Robot Anomaly Detection Tool**

| | |
|---|---|
| **Tool Name:** | Robot Anomaly Detection |
| **Purpose:** | Anomaly detection in robotic systems serves to enhance both safety and efficiency in automated and manufacturing environments. By identifying deviations from normal operational patterns, this tool allows for early detection of potential failures or malfunctions. |
| **Features and Benefits:** | • Identify anomalies on robot using data from sensors<br>• Root cause analysis to determine which specific parameters are causing the anomaly |
| **Uniqueness:** | Taking into account the domain-specific robot TELEMETRY data. Taking advantage of this domain-specific TELEMETRY data is a currently not leveraged security event generator |

| | |
|---|---|
| **Provided Upstream Interfaces (API):** | **Name:** Outcome endpoint<br>**Name:** Robot's sensor Data<br>**Description:** API that provides robot's sensor data<br>**Parameters:** -<br>**Return value:** sensor data<br>**Provided by:** NOKIA |
| **Required Downstream Services (API):** | **Name:** Entry point for robot's sensor data<br>**Description:** Receive the robot's sensor data so can be analysed<br>**Parameters:** JSON<br>**Return value:** Anomaly – Root cause<br>**Used by:** -<br><br>**Name:** Outcome endpoint<br>**Description:** Endpoint that will be used to get the outcomes of the model<br>**Parameters:** JSON<br>**Return value:** Anomaly – Root cause<br>**Used by:** SIEA pipeline (NOKIA) |
| **Input:** | Robot's Sensors Data |
| **Semantic Description of Input:** | A JSON package which contains the current robot's sensor data |
| **Output:** | Anomaly Detection Outcome |
| **Semantic Description of Output:** | The outcome of the anomaly detection model + the parameter that caused the anomaly. |
| **How can the functionality be evaluated:** | Functionality can be tested by creating "controlled" anomalies on the robot or using synthetic data |
| **Current TRL:** | 3 |

### 5.1.18 r-Anomaly Detection

This tool is designed to monitor network traffic and identify unusual patterns that deviate from established norms. It utilizes a sophisticated algorithm to analyze a dataset representing healthy or typical network activity. By continuously comparing incoming traffic data against this baseline, the tool efficiently flags anomalies, which could indicate potential security threats or system failures.

**Table 25: r-Anomaly Detection**

| | |
|---|---|
| **Tool Name:** | r-Anomaly Detection |
| **Purpose:** | This tool is designed to monitor network traffic and detect anomalies by analysing incoming data against a predefined baseline of typical activity. It employs machine learning algorithms to identify deviations that may suggest security threats or system malfunctions. The purpose is to enhance network security and reliability by promptly flagging potential issues. |
| **Features and Benefits:** | • Identify patterns that deviate from the norm (As established by a provided dataset of typical activity)<br>• Pinpointing the specific features that contribute to each detected anomaly |
| **Uniqueness:** | Anomaly detection tools are inherently unique, as they are tailored to meet the specific requirements of each use case. |
| **Provided Upstream Interfaces (API):** | **Name**: r-Monitoring Agent<br>**Description**: Agent will provide system metrics<br>**Parameters:** JSON<br>**Return value:** -<br>**Provided by:** ATC |
| **Required Downstream Services (API):** | **Name:** SIEA Pipeline<br>**Description:** Collects, aggregates, filters, and prioritizes security-related events and vulnerabilities.<br>**Parameters:** JSON<br>**Return value:** -<br>**Provided by:** NOKIA<br><br>**Name**: Data Sharing Client API<br>**Description**: JSON API that provides an abstraction of the underlying DLT platform, facilitates the submission of reports (for testing tools)<br>**Parameters:** JSON<br>**Return value:** -<br>**Provided by:** MTU |
| **Input:** | Network traffic |

| | |
|---|---|
| **Semantic Description of Input:** | Network traffic consists of data packets, each containing parameters from various layers |
| **Output:** | Outcome of model (Anomaly detected or not), severity and parameters that caused the anomaly. |
| **Semantic Description of Output:** | The output of the model will be a JSON object that contains three values: one indicating whether an anomaly was detected, one the other describing the severity of the deviation from the defined normal and a third defining the list of parameters that are considered abnormal. |
| **How can the functionality be evaluated:** | Functionality can be testes using other tools to generate traffic that is considered abnormal. |
| **Current TRL:** | 3 |

## 5.2 Utilization of the TELEMETRY Tools in Use Cases

Table 26 lists the tools developed by TELEMETRY project partners and indicates their usage in the three use cases based on Table 5. Each tool should be applied at least in one use case. Multiple usage is possible.

The legend applies as follows:

* possible

** likely

*** definite

### Table 26: TELEMETRY Requirements

| Nr | Tool Name | Project Partner | UC1 Aviation | UC2 Smart Manufacturing | UC3 Telecom-munication |
|---|---|---|---|---|---|
| 1 | SIEA Pipeline | NOKIA | | *** | *** |
| 2 | Network Fuzzer | SINTEF | ** | * | *** |
| 3 | Digital Twin | SINTEF | * | * | *** |
| 4 | BACON | ENG | | *** | *** |
| 5 | Misuse detection ML Toolkit | i4RI | ** | *** | |

| | | | | | |
|---|---|---|---|---|---|
| 6 | Secure configuration of deployment systems | i4RI | * | * | * |
| 7 | Data Space - DLT based Data Sharing | MTU | * | *** | *** |
| 8 | API Trust and Security Analyzer | MTU | | *** | |
| 9 | Mobile Cyber Range | MTU | | * | * |
| 10 | Spyderisk System Modeller (SSM) | UoS | ** | *** | *** |
| 11 | SBOM Generator | SINTEF | ** | *** | *** |
| 12 | SNORT + Ruleset | NOKIA | | * | *** |
| 13 | NOKIA Anomaly Detection Pipeline | NOKIA | ** | *** | |
| 14 | Service-Level Cybersecurity Testing (Access control system risk assessment) | WRCVE | ** | *** | ** |
| 15 | Security Scanner (Wazuh Server) | NOKIA | | * | ** |
| 16 | Secure Software Updates | KUL | *** | *** | * |
| 17 | Robot Anomaly Detection Tool | ATC | | *** | |
| 18 | r-Monitoring | ATC | | *** | *** |
| 19 | r-Anomaly Detection | ATC | | *** | *** |

## 5.3 Considerations for Deploying TELEMETRY Tools

The deployment of TELEMETRY tools is central to enhancing the security, efficiency, and reliability of IoT ecosystems across various domains, including aerospace, smart manufacturing, and telecommunications. As these tools are designed to facilitate the continuous assessment and improvement of interconnected IoT devices and systems, several key considerations must be addressed to ensure successful implementation and optimal performance. These are described below:

- The deployment of technology such as applications and components developed within TELEMETRY in these different environments can be difficult and UC providers require advanced solutions to do this. In this sense, containerization technology built on top of Kubernetes (K8s) strives for, as an integrated solution, combining and deploying multiple components developed to make this deployment efficient, automated, and secure.
- The heterogeneity of ecosystems arising from the various domains requires that TELEMETRY tools be highly adaptable and compatible with diverse technologies. Uninterrupted coexistence is essential to avoid disruptions within the ecosystem, thus, tools must be compatible with the diverse hardware profiles, software environments, and communication protocols prevalent in these sectors.
- As TELEMETRY addresses all aspects of the IoT components' lifecycle it is imperative that deployment strategies consider lifecycle stages meticulously. This involves establishing clear protocols for testing during development, integration, and operational phases, ensuring that each stage is supported by appropriate tools and methodologies from the TELEMETRY toolkit.
- Given the sensitive nature of the data handled by components of the TELEMETRY toolkit, such as network anomaly detection, it is crucial to prioritize data privacy during deployment and operation. Ensuring that all data processing complies with applicable data protection laws and adopting robust encryption methods to safeguard data both in transit and at rest are essential steps.
- A secure configuration of the deployment systems is ensured by a component which encapsulates all functionality regarded to deploy, configure and execute applications and programs developed under TELEMETRY in a secure fashion. It is composed of two main tools, namely the Kubernetes Cluster Provisioning Tool and the DevSecOps tool:
    - The *Kubernetes Cluster Provisioning Tool* aim is to provide a user-friendly way to provision fully featured production-ready Kubernetes clusters with some enhancements related to security best practices on Kubernetes clusters out of the box, using a very simple web application like an installer, to deploy clusters from a single server or even a developer laptop. It allows DevSecOps teams to dynamically provision Kubernetes clusters on-demand, ensuring scalability and agility for development and testing environments. This dynamic provisioning minimizes the overhead of manual cluster setup and maintenance.
    - The *DevSecOps Tool* will provide developers with a modern Kubernetes native continues integration / continuous delivery (CI/CD) tool to control the development and deployment of applications with an emphasis on security and using GitOps best practices for deployment. It enhances the security of applications deployed within Kubernetes clusters. It is tailored for DevSecOps

teams, emphasizing the integration of security practices throughout the application development, deployment, and operational lifecycle.

The conclusion from the analysis of these considerations is that the TELEMETRY "base" deployment and all its containing modules should support Docker Containerisation technology because these requirements are addressed using existing mature technology that is easy to integrate with and deploy. Thus, modules will be "dockerised" and tested before M18 within the UC premises and a Helm Chart can be created to allow their deployment within the K8s environment during the second phase of the project from M18 until M36. Thus, the toolkit/platform will initially be installed at UC premises using its dockerised form with any default modules plus Rancher (including UI) and K8s core features. After testing is performed, the user selects new modules. They will then be visible/installable via their local "base" platform, which will ensure and ease portability, inter-module operation, maintenance updates and other classic K8s features - all of which will be achieved from M18 until M36.

# 6 Evaluation

This chapter presents the initial validation plan for each use case. For this purpose, the chapter is divided into three subchapters, each relating to one of the use cases. We defined the same structure for all these three subchapters:

- Description of the use case
- Threats associated with the current implementation
- TELEMETRY solution
- Evaluation environment description
- Evaluation scenarios

First, the use case is described, followed by possible threats and the solution TELEMETRY will offer. Finally, the evaluation environment and the evaluation scenarios are illustrated.

## 6.1 Aviation

There will be different steps of implementation during the project run-time

The phases of the implementations are:

1. Wired approach (cargo monitoring) without satellite
2. Wired approach (cargo monitoring) with satellite
3. Wireless approach (cargo monitoring) with satellite
4. Wireless approach (flight data monitoring) with satellite

### 6.1.1 Threats Associated with the Current Implementation

- Supply chain attack
- Anomalies in operation
- Process interruption
- Rogue device
- Realtime alarm system

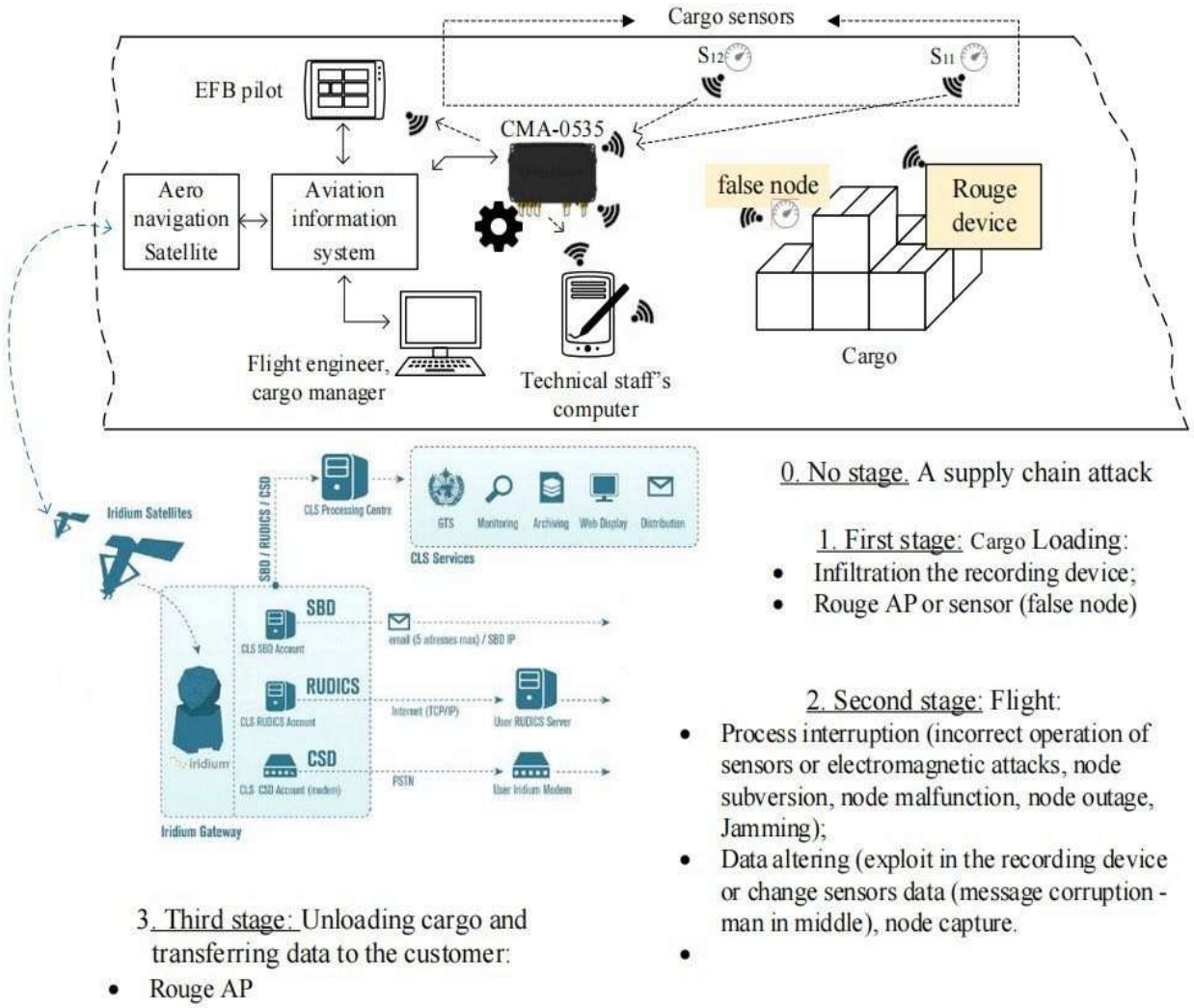



Figure 15: Overview of Threats in the Aviation Use Case

Table 27: Violations in the Aviation Use Case

| Data Protection Violation | Description |
|---|---|
| Supply chain attack | Unmaintained, unpatched (or un-patchable) firmware could be vulnerable due to exploits |
| Anomalies in operation | Detection of anomalies in network traffic and assignment of risk profiles to each device. |
| Process Interruption | The connection of the sensors to the recording unit is interrupted. |

| | |
|---|---|
| Rogue device | The security system has been manipulated, so that it sends out strong signals, which can be picked up and make the plane with the precious cargo easy to locate (and could be physically attacked). |
| Realtime alarm system | The cargo is exposed to too high/too low temperatures for a prohibitively long time. It could be prevented during the flight. |

### 6.1.2 TELEMETRY Solution

TELEMETRY tools can help improve the security of an air cargo monitoring system based on on-board indicators, information traffic analysis tools and anomaly detection.

Machine learning tools developed by TELEMETRY partners will help detect atypical behaviour of onboard information sensors (temperature, pressure, humidity), and suspicious network activity and track requests to connect to abnormal network access points.

In turn, testing the cybersecurity of on-board and ground-based IT systems will help identify vulnerabilities in restricting access to system components, and will also allow monitoring their typical or atypical behaviour, ensuring response to incidents associated with vulnerabilities at the HW, SW level (closing user rights, ensuring security logging, sending clusters to quarantine, notifying responsible employees, setting timings, issuing recommendations in case of violation of the conditions or rules for resolving an incident, logging actions when resolving an incident, etc.).

### 6.1.3 Evaluation Environment Description

Specific tools will be tested in order to address the defined UC1 storylines. The tools can be used standalone, or in combination with other tools to address a specific storyline.

The TELEMETRY tools will be installed on a separate laptop residing in the airplane. This laptop is connected to the sensor network in the airplane. It will host all required tools for the wired and wireless approach, as well as for evaluation purposes record the raw data. This allows to evaluate the tool events with respect to real events.

### 6.1.4 Evaluation Scenarios

**Rogue Device**

The security system has been manipulated, so that it sends out strong signals, which can be picked up and make the airplane with the precious cargo easy to locate (and could be physically attacked).

**Device operation anomaly**

TELEMETRY detects anomalies in network traffic and assigns a risk profile to each device. If a threshold value is exceeded, the operator is informed.

### Access control risk

Testing the cybersecurity of IT systems will allow to identify vulnerabilities in access control to system components and regulate the creation of temporary or permanent users with different access levels and sets of rights, and also allow to control their typical or atypical behaviour. TELEMETRY will also provide testing of response to incidents at the HW and SW level (closing user rights, ensuring the safety of logs, sending clusters to quarantine, notifying responsible employees, setting timings, issuing recommendations in case of violation of the conditions or rules for resolving an incident, logging actions when resolving an incident, etc.).

### Scenario 1: Rogue device

Secretly placed devices in the company network can independently spy on the network by exploiting known vulnerabilities. TELEMETRY detects anomalies in network traffic and assigns a risk profile to each device. If a threshold value is exceeded, the operator is informed.

**Table 28: Aviation Validation Plan Scenario 1**

| Tool 1 - Misuse Detection ML Tool | Tool 2 – DLT based Data Sharing | Tool 3– TELEMETRY Platform |
|---|---|---|
| 1. Detect anomaly | 2. Record the detected activity | 3.Display message for operator |

### Scenario 2: Device operation anomaly

TELEMETRY anomaly detection algorithm identifies operational change and notifies the operator about anomalies in network traffic and assigns a risk profile to each device. If a threshold value is exceeded, the operator is informed.

**Table 29: Aviation Validation Plan Scenario 2**

| Tool 1 - Anomaly Detection Pipeline | Tool 2 – DLT based Data Sharing | Tool 3 – TELEMETRY Platform |
|---|---|---|
| 1. Recognize hazardous conditions at component and system level | 2. Provide events and alerts reporting, in order to use them as input for trust assessment | 3. Provide information for the operator to validate the security level of the data |

### Scenario 3: Access control risk

The multiple accesses to information give the user mechanisms to obtain permissions from multiple policies, leading to an accumulation of effective permissions and collectively presenting a certain level of risk. A universal tool for testing access control systems will allow the network administrator to improve the quality of decisions made and reduce response time to incidents.

### Table 30: Aviation Validation Plan Scenario 3

| Tool 1 – SBOM | Tool 2 – ML Tool | Tool 3– Access control risk |
|---|---|---|
| 1. Detect list of SW, formation CVE list | 2. Subject anomaly behaviour | 3. Calculate access control risk level and send message for network administrator |

## 6.2 Smart Manufacturing

A Smart Manufacturing environment consists of multiple equipment from multiple providers with manifold divers soft- and hardware. Multiple threats are associated in such environments. Thus, evaluation need to be planned accurate.

### 6.2.1 Threats Associated with the Current Implementation

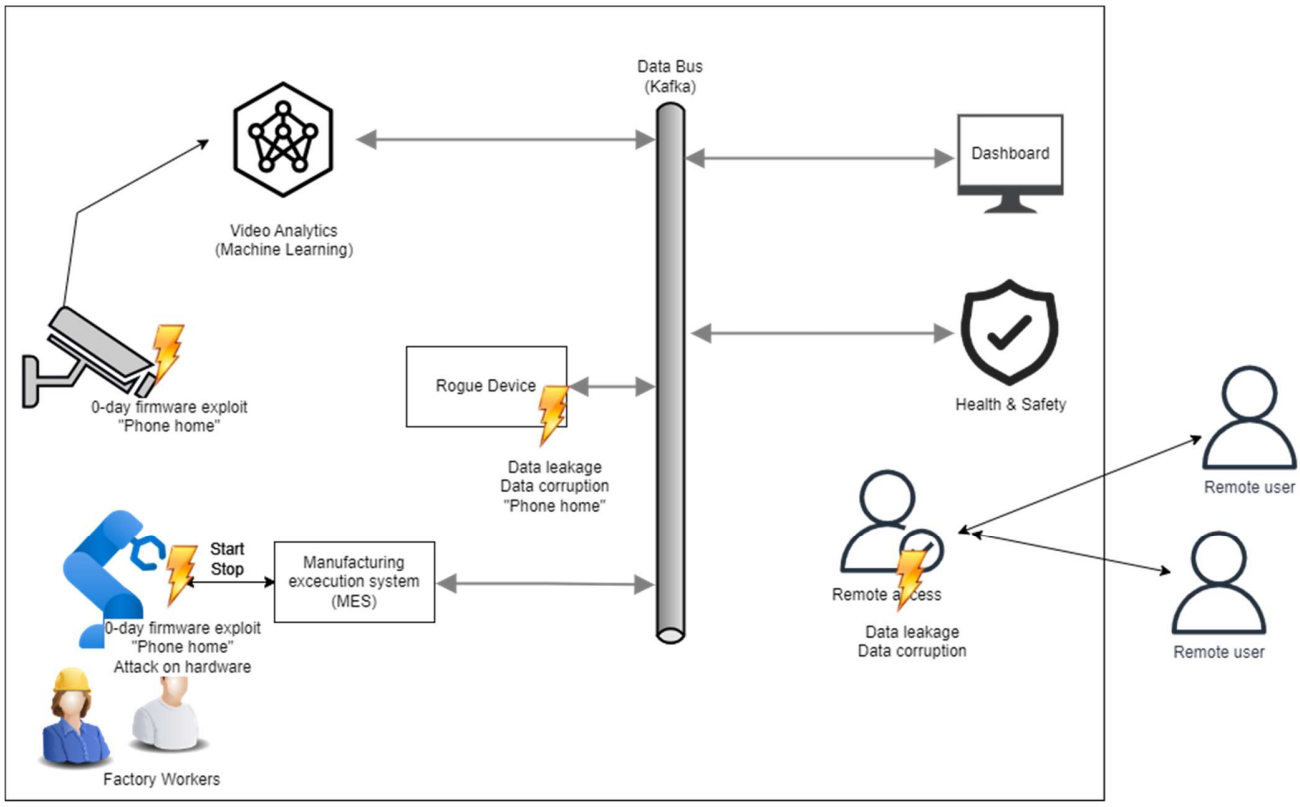



Figure 16: Overview of Threats in the Smart Manufacturing Use Case

Typical threats in a Smart Manufacturing environment may include:

- Network traffic anomaly
- Device operation anomaly
- Rogue device, device behaviour anomaly (“Supply chain risk”)
- 0-day exploit handling

Figure 14 extends the model shown in Figure 4 by threats that can occur in a smart factory.

Table 31: Violations in the Smart Manufacturing Use Case

| Data Protection Violation | Description |
|---|---|
| 0-day exploit | Unmaintained, unpatched (or un-patchable) firmware could be vulnerable due to exploits |
| Phone home | Devices could collect data and send it to a third party from within the factory network, unknown to, and unauthorised by, the factory. |
| Attack on hardware | Operational parameters of production equipment could be modified by an attacker, which could lead to damage to the production machine or degraded quality of the manufactured product |
| Rogue device | A hidden device placed on the network could collect data or attack the factory network |
| Remote users | A compromised remote user account could be used to collect data or attack the factory network |

### 6.2.2 TELEMETRY Solution

From the various ways TELEMETRY can help to improve security for the FiaB solution, 5 aspects seem to be an immediate benefit.

- Machine Learning based anomaly detection will support finding anormal network activity, such as increased traffic from a device, connection requests to not-seen-before endpoints, and untypical user behaviour (e.g. camera access by a remote user, direct device access by a user).
- Furthermore, the same mechanisms will analyse operational parameters from production devices to spot changes in their operation.
- SBOM monitoring against a CVE database will alert the administrator about new exploits, triggering an update or replacement of the device.
- The alerts generated by those entities will be aggregated and assessed by a risk management system and presented to the administrator along with remedy proposals.
- All events will be logged in a distributed ledger system for future audit purposes.

### 6.2.3 Evaluation Environment Description

For a proof of concept, it is planned to integrate the TELEMETRY solutions in the Nokia “Factory-in-a-Box" (FiaB) lab. The FiaB is a setup where product and research developments take place and as such represents a real test environment.

The TELEMETRY solutions will be hosted on Ubuntu virtual machines, which are connected to the FiaB network and have access to all resources they need for their operations. Access to the network will be provided by means of VPN connection. Additionally, to the data a log of real live events (e.g. robot finished a routine) will be recorded. Verification data will be kept under closed access, that will be used later for objective evaluation.

### 6.2.4 Evaluation Scenarios

The evaluation scenarios depend on the data protection capabilities of the TELEMETRY Software artefacts at the time of the demo implementation. Nonetheless, the envisioned validation of the TELEMETRY concept and architecture uses the following scenarios.

**Scenario 1: Network anomaly**

An external user connected via VPN connects to the IP camera. This is unusual user behaviour, as the camera stream should only be used from within the factory and not from the outside.

This activity should be detected by the Network Anomaly Machine Learning tool and reported to the administrator.

**Table 32: Smart Manufacturing Validation Plan Scenario 1**

| **Tool 1 - ML** | **Tool 2 – Data space – DLT Based Data Sharing** | **Tool 3 – Spyderisk System Modeller (SSM)** | **Tool 4 – TELEMETRY Platform** |
|---|---|---|---|
| 1. Detect activity and create a report | | | |
| | 2. Record the detected activity | | |
| | | 3. Calculate risk level and send to dashboard | |
| | | | 4.Display message from SSM |

**Scenario 2: Rogue device**

The Robot starts to send data to an unknown IP address, which will be detected by Network Anomaly detection and reported to the administrator.

**Table 33: Smart Manufacturing Validation Plan Scenario 2**

| Tool 1 - ML | Tool 2 – Data space – DLT Based Data Sharing | Tool 3 – Spyderisk System Modeller (SSM) | Tool 4 – TELEMETRY Platform |
|---|---|---|---|
| 1 Detect the anomaly | | | |
| | 2. Record the detected activity | | |
| | | 3. Calculate risk level and send to dashboard | |
| | | | 4.Display message from SSM |

**Scenario 3: Device operation anomaly**

The control messages which define how the robot is working are being modified, resulting in a slightly changed way on how the robot operates. The anomaly detection algorithm recognizes this change in the operation and alerts the administrator.

**Table 34: Smart Manufacturing Validation Plan Scenario 3**

| Tool 1 - ML | Tool 2 – Data space – DLT Based Data Sharing | Tool 3 – Spyderisk System Modeller (SSM) | Tool 4 – TELEMETRY Platform |
|---|---|---|---|
| 1 Detect the anomaly | | | |
| | 2. Record the detected activity | | |
| | | 3. Calculate risk level and send to dashboard | |
| | | | 4.Display message from SSM |

**Scenario 4: 0-day exploit**

BACON finds a vulnerability with the help of anomaly detection based on network traffic data. This is being reported to the administrator.

Table 35: Smart Manufacturing Validation Plan Scenario 4

| **Tool 1 – ML** | **Tool 2 – Data space – DLT Based Data Sharing** | **Tool 3 – Spyderisk System Modeller (SSM)** | **Tool 4 – TELEMETRY Platform** |
|---|---|---|---|
| 1 Find the vulnerability in CVE database | | | |
| | 2. Record the detected activity | | |
| | | 3. Calculate risk level and send to dashboard | |
| | | | 4.Display message from SSM |

## 6.3 Telecommunication

### 6.3.1 Threats Associated with the Current Implementation

The European Standards Institute (ETSI)[4] published the technical report: TR 103 743 CYBER: Home Gateway Security Threats Analysis. The ETSI document provides an analysis of cyber security threats specific to Home Gateways (HGs) and an introduction to measures for risk mitigation posed by these threats.

The Home Gateway (the HG of the ETSI document corresponds to the RGW in the UC3 context of the TELEMETRY project) is defined as a physical device that lies between the home network (LAN, Local Area Network) and the public network (WAN, Wide Area Network), with a primary purpose of dividing and isolating home network traffic from external network traffic. It can be provided for retail purchase by the user or can be supplied as part of a service contract with the Internet Service Provider (ISP), Telecom Italia in our case.

There is assumed to be no restriction on the availability of the HG and thus attackers are considered to have freedom of access to the HG. More specifically, the attacker may have unrestricted access to an instance of the HG in order to develop different attack strategies and to maximize each system knowledge (i.e. of the HG), time (i.e. to optimize the time required to be able to launch an attack), expertise (i.e. time to develop knowledge of the HG's operation, weaknesses and vulnerabilities), and each of opportunity and equipment (i.e. develop means of access and any equipment in addition to the HG in order to launch an attack).

Best practice suggests that the HG should be provisioned in such a way that any sensitive configuration data is not accessible to normal user accounts, but rather a privileged administrator account should be required to update configuration or to analyse administrative data (e.g. log files).

Points of attack on the HG include the open interfaces of the home network side of the HG, interfaces open on the ISP side of the HG, and the supply chain of the HG, as shown in Figure 17.

The owner/user of the HG can act as an attacker either deliberately or by accident, or act as a vector in some forms of attack. The HG is considered user accessible, i.e. the device can be opened, and a user can examine the components internal to the device.

The employed Threat Model is based on ETSI's Threat Vulnerability and Risk Analysis (TVRA) as defined in ETSI TS 102 165-1, combined with Microsoft's STRIDE[5] (Spoofing, Tampering, Repudiation, Information disclosure, Denial of services, Elevation of privilege) model.

4 European Telecommunication Standard Institute: https://www.etsi.org

5https://learn.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)?redirectedfrom=MSDN

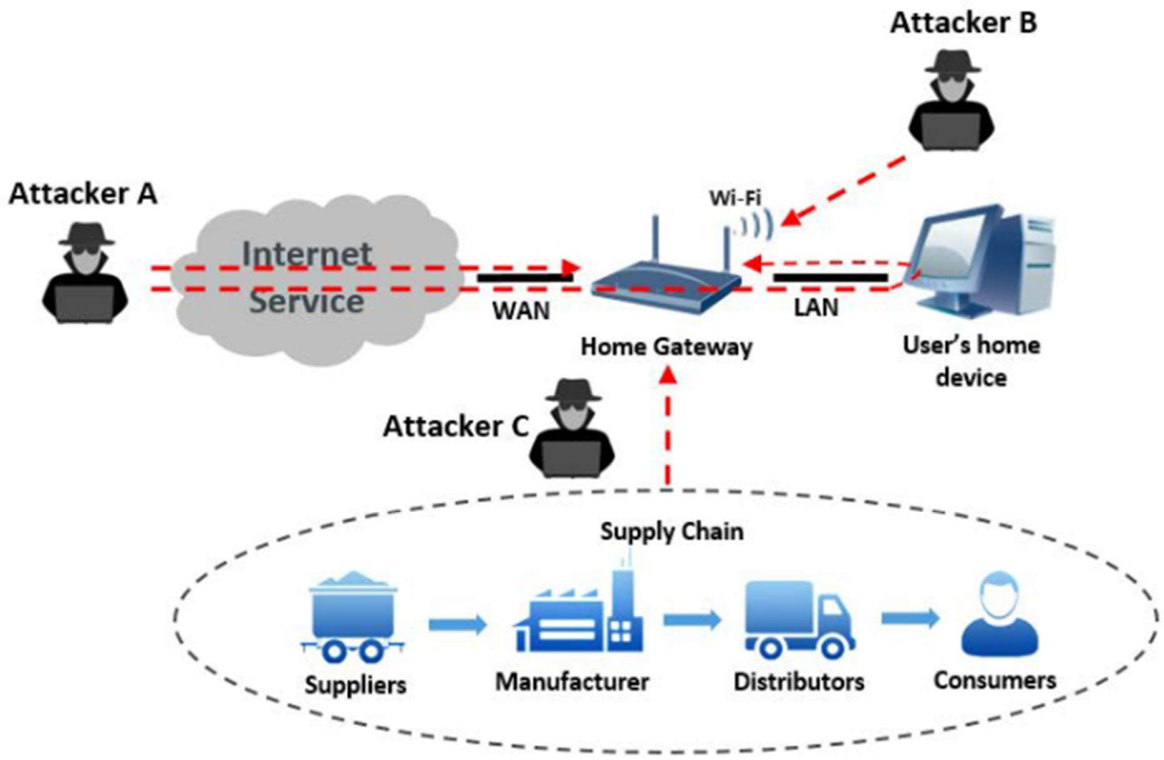



Figure 17: Overview of Threats in the Telecommunication Use Case (from the ETSI TR 103 743 V1.1.1)

Table 36 below reports the most relevant threats identified for the UC3 scope.

Table 36: Violations in the Telecommunication Use Case

| Scenarios vs Threats | Scenario A<br>Attacks via the WAN interface | Scenario B<br>Attacks via the LAN interface | Scenario C<br>Attacks across the supply chain | Note |
|---|---|---|---|---|
| Inject and execute malware | yes | yes | yes | Many kinds of attacks are foreseen here, such as structured query language (SQL) injection, cross-site scripting (XSS), and command injection. Malware can use known software vulnerabilities. |

| | | | | |
|---|---|---|---|---|
| Obtain access to HG | yes | yes | n.a. | By exploiting flaws in the system's code, brute force attacks to guess a weak password, or social engineering. |
| Disrupt or disable the services | yes | n.a. | n.a. | Many kinds of DDoS attacks (e.g. DNS, amplification, etc.) are possible from the public WAN |
| Erasure of evidence of attacks | yes | n.a. | n.a. | Once the attacker controls the HG, he can also hide/delete his traces. Hence it is important to prevent possible device hijacking |

### 6.3.2 TELEMETRY Solution

The Telco UC is focused on the security testing of the RGW related to (or at most at the final stage of) the design time. The main goal is the verification (and if needed the patching) of several issues, such as the presence of possible vulnerabilities, possible misconfigurations, to malware injections due to supply chain attacks. In order to effectively protect customers, it is extremely important that these problems are detected and resolved during the so-called "testing phase" and before any vulnerable device is supplied to the final customers as part of the service contract and deployed in production ("production phase"). The runtime/operation time is not in the scope of the UC3.

Moreover, by defining several storylines related to the enhancement of the testing process, UC3 will be used to validate also possible solutions able to support the operations of the security testers, enabling the continuous improvement of the process, according to the plan-do-check-act (PDCA)[6] model.

### 6.3.3 Evaluation Environment Description

During the testing phase, we will be focused on specific tools able to address the defined UC3 storylines. Each tool can be used in a standalone fashion, or in combination with other tools to address a specific storyline. The tools will be hosted in the Telecom Italia laboratories. Access will be provided by means of a VPN connection.

6https://en.wikipedia.org/wiki/PDCA

In this UC traffic generators will be utilized to emulate subscriber and malicious traffic. Home gateways from different vendors will be used to proof the applicability of the TELEMETRY tools.

### 6.3.4 Evaluation Scenarios

The evaluation scenarios defined in the following chapters below aim to select proper TELEMETRY tools to be used in the UC3 for each defined storyline and then validate the achieved results. The rationale is that by identifying the vulnerabilities (and patching them), possible misconfigurations or compromised components in the DUT by enhancing the testing tools and the security testing process, the threats and risks identified can be better mitigated.

#### *6.3.4.1 Evaluation in the Vulnerability Prioritization Scenario*

In the case of vulnerability prioritization, the aim is to enable the prioritisation of CVEs via risk assessment. UoS proposes to use the tool Spyderisk (also called SSM) and extend it to be able to model the software/hardware structure of a router so that different consequences (each with severity and likelihood that make up the risk associated with the consequence) can be prioritised and assessed. The idea is to map these consequences to vulnerabilities mentioned in CVEs via modelling of the software structure – e.g. the libraries installed, their dependencies, their version and linking this information to CVE and vulnerability databases. The list of identified CVEs can come from the Telecom Italia standard tools (e.g. vulnerability scanners) or from the TELEMETRY tools, in particular the SBOM provided by SINTEF. Additionally, the SSM can use also additional information, such as the list of the actual software libraries used by the DUT (still provided by the SBOM) or other information about the DUT area that could contain possible issues, as detected by the Network Fuzzer.

SSM's scope is to understand the software/hardware architecture of the router and update the knowledge base of its risk management tool with new assets, threats, consequences, vulnerabilities and controls that enable the modelling of these elements specifically targeted at a software library/dependency/firmware level. The knowledge is likely to come from experimentation/desk research/discussion with and expertise of project partners.

The evaluation of the results will be based on the TIM internal Threat Intelligence processes, manually performed by expert security analysts.

#### *6.3.4.2 Evaluation in the Backporting Scenario*

In order to detect a “backporting”, a non-present CVE identified by a vulnerability scanner because of the library/software version and not because actually detected/present, it is needed to trace the actual modification of the libraries installed on the DUT.

Two approaches will be used, based on the usage of the following tools:

- SBOM tool
- Wazuh tool, in an agent-less mode, i.e. installed on an external system (e.g. on a VM adjacent to the DUT directly in the testbed)

Both, Wazuh and the SBOM tool, will be able to scan the software installed on the DUT and compute a “digest” for each installed library (e.g. a hash value computed by means of the MD5

message-digest algorithm[7]). Subsequent scans can reveal modifications of the installed software, also when the official version remains the same, hence revealing the backporting.

Similar to the validation of the Supply Chain Risk scenario, two different firmware releases will be used in the backporting scenario as input for the involved tools, whereas in the most recent release, the binary content of one or more software components will be changed, keeping the original software version, in order to simulate a backporting event.

#### *6.3.4.3 Evaluation in the 0-day Scenario*

In this scenario, the SINTEF Network Fuzzer will be used. The Fuzzer will be able to generate specific (malformed) kinds of input to be sent to the DUT and then evaluate the possible answers of the DUT itself and then identify the areas that can contain an issue. In order to speed up the testing procedures, it would be useful to emulate the DUT in a virtual environment, although it is actually difficult to virtualize the full DUT given the proprietary nature of the firmware and the presence of specific physical interfaces. Anyway, during the evaluation period, both approaches will be tested: the actual physical DUT and a virtualized one by using the open-source version of the firmware.

To be noted that the Fuzzer can be used to focus on specific libraries and not the full firmware. For example, the Fuzzer can be integrated with the SSM, which can suggest the most critical functional block, based on the security assessment and the result of an SBOM analysis (actual libraries), to emulate the digital twin to speed up the testing processes.

#### *6.3.4.4 Evaluation of the Virtualization Scenario*

The virtualization of firmware allows testers to test the device without consequences. For instance, they can perform attacks, trying to exploit zero-day and known vulnerabilities. Testing activities can also be fully parallelized, running several test cases simultaneously on different virtualized devices.

For evaluation, a comparison will be made between the test results obtained on the virtual model and those obtained on the physical device. The comparison will focus on the components made available by the virtualized model and will concern both the consistency of the results and the time required to obtain them. Anyway, it should be noted that the availability of a virtualized environment for devices is in itself a great advantage, as it avoids the scarcity of physical devices and enables additional time savings through parallelization of several test case execution.

#### *6.3.4.5 Evaluation in the Access control Risks Scenario*

The WRVCE methodology is expected to produce reasonable recommendations for reducing the possible risks related to the access control system. Such results will be validated by cyber security analysts, supported by the internal TIM Cyber Security and Threat Intelligence processes.

---

7 https://en.wikipedia.org/wiki/MD5

A common problem with complex access control systems is that they provide the user with a mechanism to obtain permissions from multiple policies, which leads to an accumulation of effective permissions and collectively introduces a certain level of risk. Determining whether an access level poses a potential security risk is not a trivial task.

A tool that allows to evaluate the effectiveness of an enterprise's access control system will help detect hidden vulnerabilities and improve cybersecurity in the following areas:
- reduce incident response time;
- improve the quality of decisions made;
- minimizing damage from cybersecurity breaches;
- increase the efficiency of the entire information system.

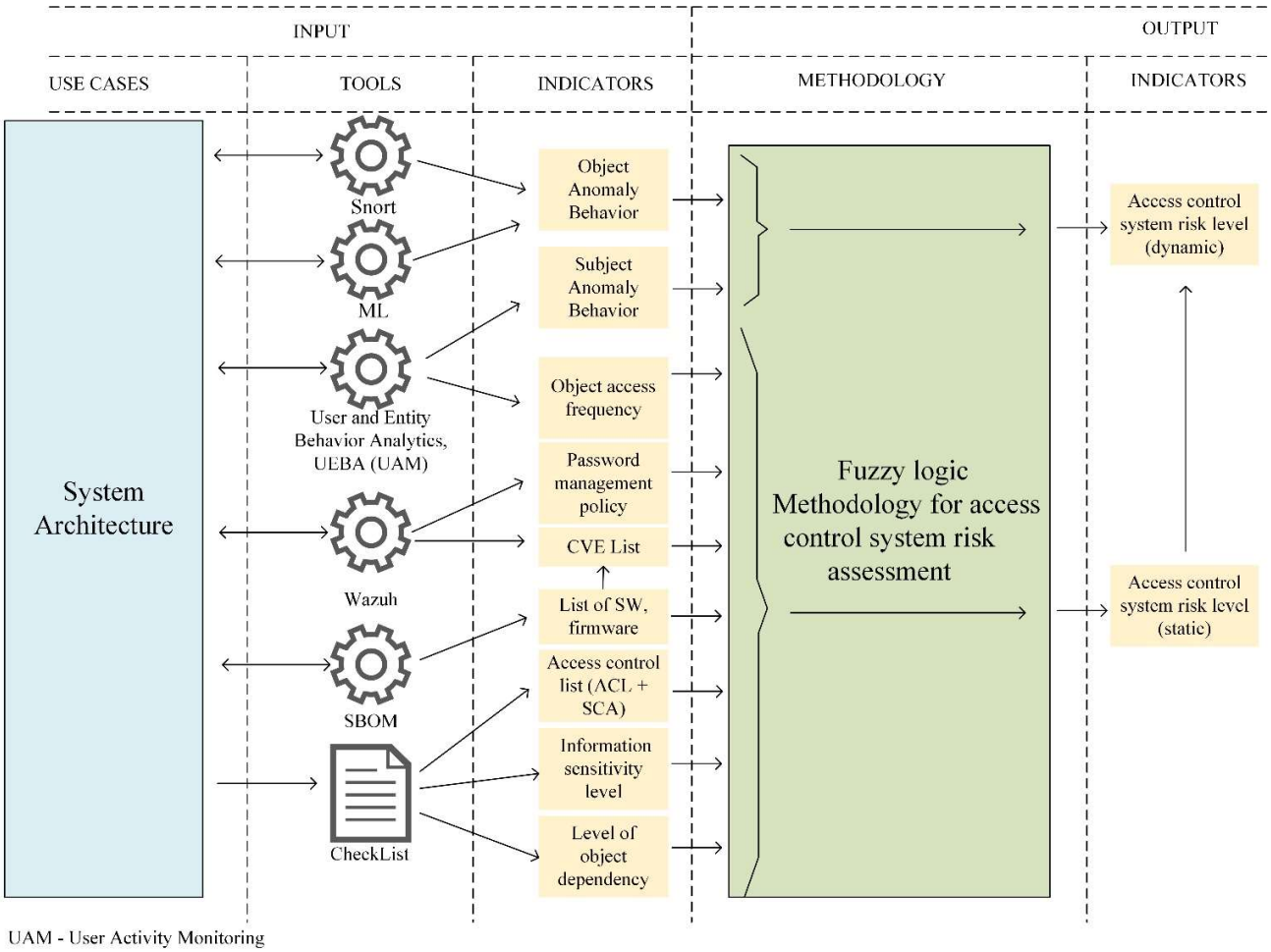



**Figure 18: Methodology's Architecture for Access Control System Risk Assessment (all necessary instruments)**

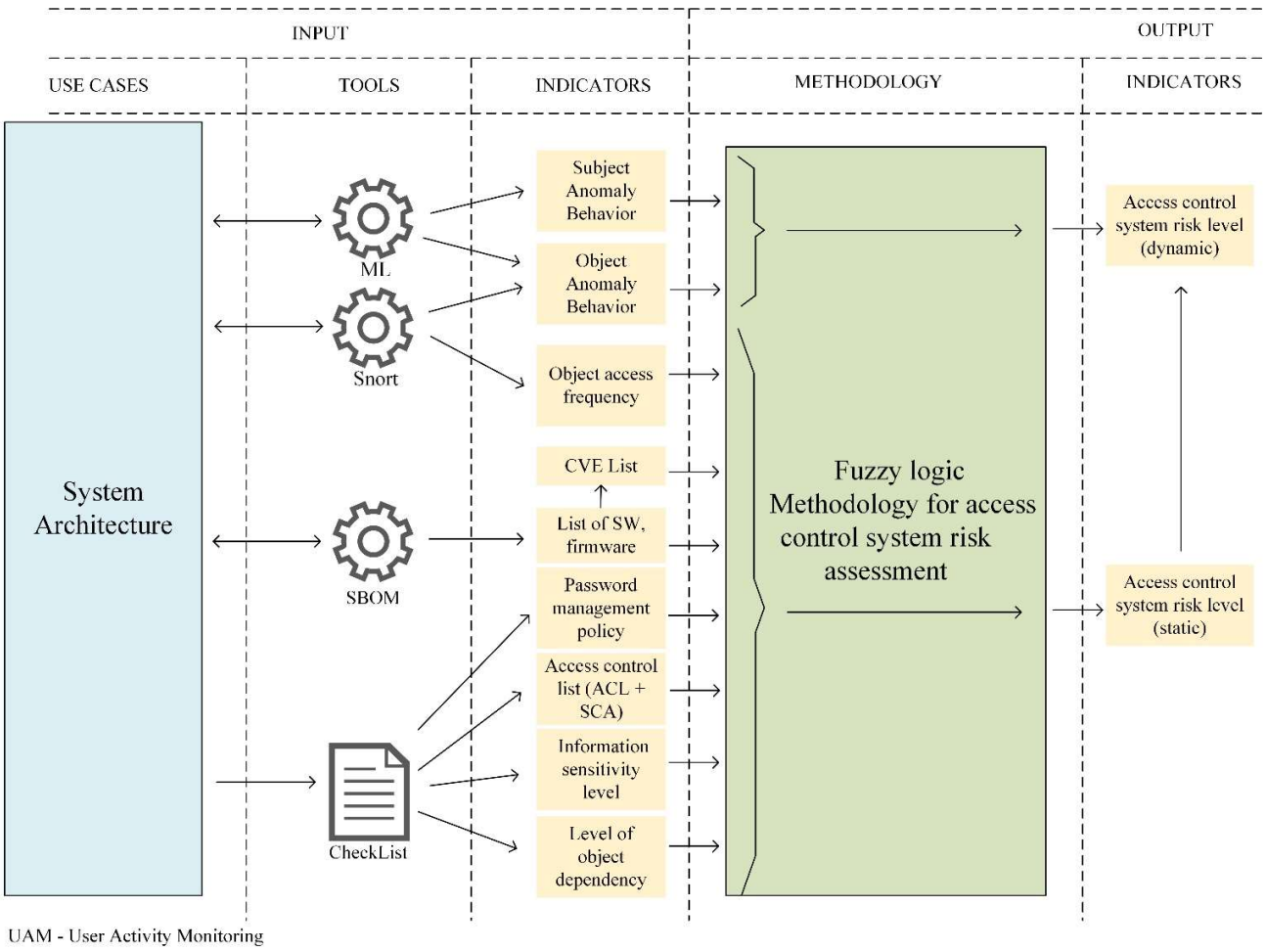



**Figure 19: Methodology's Architecture for Access Control System Risk Assessment (TELEMETRY instruments only)**

Assessing the effectiveness of access control to IT system components will include the following KPIs:

- KPI 1 – accuracy of issuing appropriate rights to system users (%);
- KPI 2 – control system operating speed (operations/time);
- KPI 3 – the ability to adapt the access control system to changes in the user profile or other factors (%).

To validate the methodology, two approaches will be considered:

1. Static analysis.
2. Dynamic analysis of the system (promising direction of development).

As part of the static analysis, input data about the state of the system (access policies, access control system, presence of vulnerabilities) obtained using the TELEMETRY tools (SBOM, Snort) is used (Figure 18).

User and entity behaviour analytics (UEBA) or User Activity Monitoring (UAM) is a cybersecurity solution that uses algorithms and machine learning to detect anomalies in the behaviour of not only the users in a corporate network but also the routers, servers, and endpoints in that network. UEBA seeks to recognize any peculiar or suspicious behaviour—instances where there are irregularities from normal everyday patterns or usage. The UEBA solution then goes "silent" as it starts collecting data on device and network usage. In learning mode, the UEBA solution's algorithms will determine and further define what is considered normal or even optimal. UEBA system monitors not only human activity on devices but also the devices themselves, including servers, routers, endpoints, and Internet-of-Things (IoT) devices.

With the help of algorithms constructed using the fuzzy set apparatus, the risks of the access control system are calculated taking into account the real state of the system.

As part of dynamic analysis, the system is monitored for abnormal user behaviour (ML tools), which is taken into account to recalculate the degree of current risk.

#### *6.3.4.6 Evaluation in the Supply Chain Risks Scenario*

A supply chain attack[8] is a combination of at least two attacks. The first attack is on a supplier that is then used to attack the final target to gain access to its assets. The target can be the end customer or another supplier. Therefore, for an attack to be classified as a supply chain one, both the supplier and the customer have to be targeted.

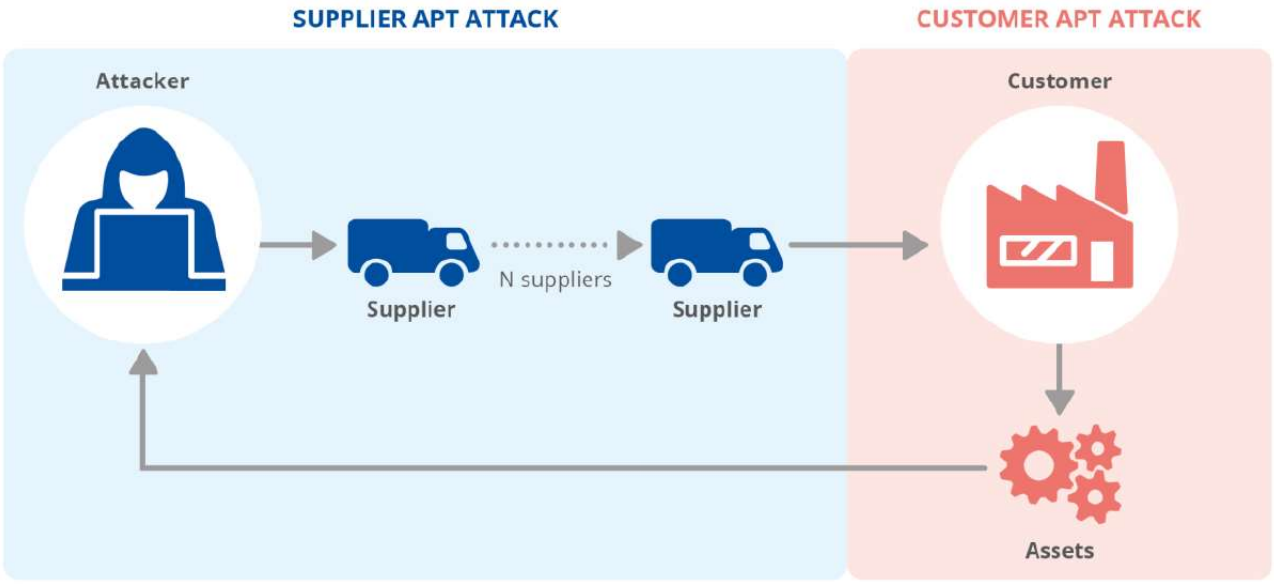



**Figure 20: Illustration of Supply Chain Risk (from ENISA THREAT LANDSCAPE FOR SUPPLY CHAIN ATTACKS, July 2021)**

8 Reference: ENISA THREAT LANDSCAPE FOR SUPPLY CHAIN ATTACKS, JULY 2021

The SolarWinds hack is one of the first attack scenarios used to refer to the supply chain risks. The attackers compromised SolarWinds Orion and modified the code of ORION software (first attack). As a consequence, The ORION instances in the customers were updated with malware, which allowed the attackers to access the data of customers (final attacks). Many other examples are available, although the model is the same or very similar to the SolarWinds case. Also recently, a maliciously introduced backdoor in the Linux utility xz within the *liblzma* library has been discovered[9]. In fact, public software repositories are nowadays the preferred target for supply chain attacks.

In order to validate the TELEMETRY tools in the Supply Chain Risks storylines two different approaches will be used:

- Static analysis approach
- Dynamic analysis approach

In the Static Analysis approach, only the SINTEF SBOM tool will be used.

Considering the Supply Chain risk definition given before, the UC3 scenario is temporarily placed after a hypothetical attack against the Telecom Italia suppliers and before the final attack against the end users (the residential customers). The aim is to detect any possible malware already present on the device, e.g. within the libraries installed on the DUT.

The following steps describe the process:

- The DUT in the TIM network has been updated with malware in order to simulate an infected device, which could allow the attackers to access the DUT.
- SBOM tool is used to identify all the SW installed on the DUT (static analysis)
- For each piece of identified SW, the SBOM tool identifies also its version and, when available, the list of the (already known) CVEs.
- The risk can be prevented whenever the SW used by the DUT is known to be compromised with malware or contains known CVEs (this kind of information is available via several Threat Information sources).

For the validation process, a sample analysis will be carried out, manually and/or supported by other tools, of the modules and corresponding versions reported by the SINTEF SBOM tool. The ability to identify installed but not running modules will also be investigated.

The SINTEF SBOM tool also provides a database containing, for each identified software component, a corresponding fingerprint (based on a hash computation of the binary content). In this way, it will be possible to compare different software releases identifying anomalous situations corresponding to a change in the binary but not in the version of installed components.

9 (see https://nvd.nist.gov/vuln/detail/CVE-2024-3094)

In this case, the process is described by the following steps:

- The DUT in the TIM network has been updated with malware, which could allow the attackers to access the DUT. More specifically, the malware has been injected into a software component present on the firmware and not updated by the manufacturer (typically only a part of the firmware is updated in a new firmware release).
- SBOM tool is used to identify all the SW installed on the DUT (static analysis).
- For each piece of identified SW, the SBOM tool identifies its version and the corresponding fingerprint (based on the binary content). It also compares these values with those corresponding to the previous releases.
- The risk can be prevented whenever a software component presents the same version but a different binary content with respect to those contained in previous firmware releases. It should be noted that in this case, the injected malicious component can be totally unknown and undetectable by traditional tools, such as anti-malware or other signature-based tools.

For the validation of this attack case, two different firmware releases will be used as input for the tool, wherein the most recent release the binary content of a software component (or several components) will be modified, keeping the same software version, in order to simulate a malware injection.

In the Dynamic Analysis approach, it is supposed no known vulnerabilities are present at starting time on the DUT. The presence of a possible malware can be detected by triggering its activation by simulating a real environment with data traffic flowing in and out of the RGW (for instance traffic generated by a traffic simulator). The hidden malware could start generating its own traffic toward a Command and Control (C2) server (e.g. specific IP addresses) inserted in a blacklist because included in a botnet. Anomalous behaviours like that will be detected by the anomaly detection tools and reported to the security tester as evidence of a device compromise. Other kinds of misbehaviour can be related to the irregular consumption of resources, such as CPU cycles, transmission control protocol (TCP)/user datagram protocol (UDP) ports opening, memory consumption, and so on.

For evaluation, two different types of procedures will be taken into consideration, depending on the technology used by the tools to detect the anomalies:

- Machine Learning (ML)-based tools (e.g. BACON, r-Monitoring[10])
- Not ML-based tools (e.g. Snort)

Whereas no-ML-Based tools can be deployed immediately in the testbed to detect anomalies based on e.g. specific attacks signatures or a white/blacklist of IP addresses, ML-based tools

10 Some features of r-Monitoring will not require specific learning phases, whereas the network anomaly detector will do.

require a preliminary learning phase during which the tools can “learn” the features of the assumed clean baseline of traffic behaviour.

The learning phase will be performed as follows: synthetic, but realistic, traffic for a typical residential customer (e.g. HTTPs web browsing, DNS, email, streaming traffic, and so on) will be generated by a traffic generator and captured by means of a sniffer in the TIM testbed. The collected traffic (e.g. PCAP files) will be transferred to the tool owner's premises for their learning phases and generation of the model. The same traffic files can also be used by the others for their internal tests (e.g. for Snort testing).

The same approach will be adopted also for the subsequent “detection phase”, at least for the first iteration of the validation phase. Again, in the TIM testbed will be generated the same/similar synthetic traffic but with the insertion of some “anomalies”, such as:

- The attempt to start a communication with a specific IP address is part of a potential C2 of a botnet.
- A connection of abnormal duration.
- A connection with an abnormal amount of data exchanged.
- Simulation of data exfiltration attempts by tunnelling different protocols (such as DNS).

The newly generated traffic will be captured by a sniffer and transferred to the tool owner’s premises for testing the detection phase.

Hence there will be two kinds of validations:

- In the tool owner premises, to be performed during the first iteration.
- In the TIM testbed, during the final iteration where the tool, already tuned and configured, will be inserted in the actual testbed.

The anomalies detected by monitoring the DUT resources (e.g. file system, CPU, RAM) such as the case of the r-Monitoring agent, will be verified directly in the TIM testbed[11].

Only one ML-based tool will be selected to be used in the testbed, depending on the performance and results achieved.

11 It depends on the final architecture of the r-Monitoring tool. The agent will be installed on the DUT but the central collector could be installed also in the tool owner premises and communicate via the public interface of the testbed.

# 7 Conclusions

The work presented in this deliverable can be summarized as follows:

- we have collected and structured requirements from various sources
- intensive discussions between project partners have led to a common understanding of the respective use cases and their storylines
- similarly, we gathered the TELEMETRY tools and their functionality
- we derived an evaluation plan based on the functionality of the TELEMETRY tools and the storylines of the use cases
- finally, by taking all this input into account, we compiled a first version of the TELEMETRY architecture, which will be further detailed in D1.2

Given the holistic approach, this deliverable builds the basis for further work in several directions. First, towards solutions for satisfying the goals and requirements enlisted in this document. Second, the use case partner and tool owners will work on implementing and extending the use cases according to the descriptions provided here. Finally, the work in the use cases will converge in the validation activities, according to the validation plans described here.