



Horizon 2020 Program
Dynamic countering of cyber-attacks
SU-ICT-2018



Cyber security 4.0: Protecting the Industrial Internet of Things

D1.3: Architecture definition^{1†}

Abstract: The purpose of this deliverable is to provide a refined conceptual specification of the C4IIoT platform architecture. First, it presents a detailed description of each of the different modules provided by the technology providers, and then explains how they will be integrated into the C4IIoT platform. It also discusses the development efforts that will be carried out during the project for each module.

The modules are grouped into three architecture layers, from those closer to the production and the machines (i.e., the edge), continuing through the field gateway layer, up to the highest-level modules that reside in the cloud. In addition, D1.3 also provides a description of the user interface. From a security point of view, the architecture is divided into three levels – hardware-based security, security of horizontal device-to-device communications and machine learning based behavioural analysis and cognitive security capabilities. The results of this deliverable will serve as the major guidelines of the C4IIoT Minimum Viable Product (MVP).

¹ † The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 833828.

Contractual Date of Delivery	30/11/2019
Actual Date of Delivery	30/11/2019
Deliverable Security Class	Public
Editor	<i>Srdjan Skrbic, Zarko Bodroski, Dusan Jakovetic (UNSPMF)</i>
Contributors	All C4IIoT partners
Quality Assurance	<i>Giorgos Tsirantonakis (FORTH) Giorgos Vasiliadis (FORTH) Jacques Robin (UPIPS) Jlenia Puma (CRF)</i>

The *C4IIoT* Consortium

FOUNDATION FOR RESEARCH AND TECHNOLOGY HELLAS	Coordinator	FORTH	EL
CENTRO RICERCA FIAT SCPA	Principal Contractor	CRF	IT
INFINEON TECHNOLOGIES AG	Principal Contractor	IFAG	DE
THALES SIX GTS FRANCE SAS	Principal Contractor	TSG	FR
HEWLETT PACKARD ITALIANA SRL	Principal Contractor	HPE	IT
COMMISSARIAT A L ENERGIE ATOMIQUE ET AUX ENERGIES ALTERNATIVES	Principal Contractor	CEA	FR
IBM ISRAEL - SCIENCE AND TECHNOLOGY LTD	Principal Contractor	IBM	IL
AEGIS IT RESEARCH UG	Principal Contractor	AEGIS	DE
UNIVERSITE PARIS I PANTHEON-SORBONNE	Principal Contractor	UP1PS	FR
INFORMATION TECHNOLOGY FOR MARKET LEADERSHIP	Principal Contractor	ITML	EL
SPHYNX TECHNOLOGY SOLUTIONS AG	Principal Contractor	STS	CH
UNIVERSITY OF NOVI SAD FACULTY OF SCIENCES	Principal Contractor	UNSPMF	SRB
UNIVERSITY OF GREENWICH	Principal Contractor	UOG	UK
VIP MOBILE D.O.O.	Principal Contractor	VIP	SRB

Document Revisions & Quality Assurance

Internal Reviewers

1. *Giorgos Tsirantonakis (FORTH)*
2. *Giorgos Vasiliadis (FORTH)*
3. *Jacques Robin (UP1PS)*
4. *Jlenia Puma (CRF)*

Revisions

Version	Date	By	Overview
1.0.0	29/11/2019	Srdjan Skrbic, Zarko Bodrski (UNSPMF)	Final
0.3.0	18/11/2019	Srdjan Skrbic, Zarko Bodrski (UNSPMF)	Third phase, final
0.2.0	4/11/2019	Srdjan Skrbic, Zarko Bodrski (UNSPMF)	Second phase, final
0.1.0	3/10/2019	Zarko Bodrski, Srdjan Skrbic, Dragana Bajovic, Milos Savic (UNSPMF)	First phase, final
0.0.4	23/09/2019	Zarko Bodrski, (UNSPMF)	First phase, draft
0.0.3	21/08/2019	Srdjan Skrbic, Zarko Bodroski, Dusan Jakovetic (UNSPMF)	ToC Second draft
0.0.2	14/08/2019	Zarko Bodroski (UNSPMF)	ToC First draft
0.0.1	12/08/2019	Zarko Bodroski (UNSPMF)	ToC

Table of Contents

LIST OF FIGURES.....	7
LIST OF ABBREVIATIONS.....	8
EXECUTIVE SUMMARY.....	9
1 INTRODUCTION.....	10
2 C4IIOT ARCHITECTURE OVERVIEW.....	11
2.1 OVERVIEW OF THE ARCHITECTURE	11
2.2 CLOUD LAYER MODULES DESCRIPTION	14
2.2.1 Mitigation engine – Reconfiguration Search (<i>VariaMos</i>)	14
2.2.2 Mitigation engine – SDN controller.....	15
2.2.3 Mitigation engine – Binary code analyzer (<i>BINSEC</i>).....	15
2.2.4 Security Assurance Module.....	16
2.2.5 Risk assessment	18
2.2.6 Cloud gateway.....	19
2.2.7 Cloud Management & Orchestration.....	20
2.2.8 Privacy aware, Trustworthy Data & Analytics	21
2.2.9 Advanced visualisation, Private/secure data analytics.....	22
2.3 FIELD GATEWAY LAYER MODULES DESCRIPTION	23
2.3.1 SDN-enabled switch/router.....	23
2.3.2 Security-aware dynamic offloading decision mechanism	24
2.3.3 Field gateway.....	24
2.4 EDGE NODE LAYER MODULES DESCRIPTION	25
2.4.1 Trucks/Warehouses and Smart Factory Edge Devices	25
2.5 LEVEL 3 – SECURITY BY ML-BASED BEHAVIOURAL ANALYSIS & COGNITIVE SECURITY CAPABILITIES – MODULES DESCRIPTION.....	27
2.5.1 Level 3 – behavioural analysis & cognitive security	27
2.5.2 Level 3 – Traffic analysis.....	27
2.6 LEVEL 2 – SECURITY BY HORIZONTAL DEVICE-TO-DEVICE COMMUNICATION – MODULES DESCRIPTION	28
2.6.1 Level 2 – Identity management.....	28
2.6.2 Level 2 – Decentralized access control.....	29
2.6.3 Level 2 – Decentralized access control with attribute-based encryption (<i>ABE</i>).....	30
2.7 LEVEL 1 – HARDWARE ENABLED SECURITY	31
2.7.1 Level 1 – Secure execution environment.....	31
2.8 SIMULATED ENVIRONMENT	32
2.8.1 Fabricated devices.....	33
3 EDGE NODE LAYER.....	35
3.1 EDGE NODE LAYER ARCHITECTURE REFINEMENT	35
3.1.1 Logistics 4.0 use case.....	35
3.1.2 Smart Factory use case.....	37
4 FIELD GATEWAY LAYER.....	38
4.1 FIELD GATEWAY LAYER ARCHITECTURE REFINEMENT	38
4.1.1 Logistics 4.0 use case	38
4.1.2 Smart Factory use case.....	41
5 CLOUD LAYER.....	42
5.1 CLOUD LAYER ARCHITECTURE REFINEMENT	42
6 USER INTERFACE.....	45
6.1 FVT INTEGRATION REGARDING THE ARCHITECTURE REFINEMENT	45

7	INTEGRATION OF SECURITY ASPECTS.....	47
7.1	LEVEL 1 – HARDWARE-ENABLED SECURITY	47
7.2	LEVEL 2 – SECURITY BY HORIZONTAL DEVICE-TO-DEVICE COMMUNICATION	48
7.2.1	<i>Blockchain.....</i>	<i>49</i>
7.3	LEVEL 3 – SECURITY BY ML-BASED BEHAVIOURAL ANALYSIS & COGNITIVE SECURITY CAPABILITIES..	51
7.4	PRIVACY-AWARE ANALYTICS AND ACCOUNTABLE DATA PROCESSING	53
7.5	SECURE COMMUNICATION PROTOCOLS	54
8	CONCLUDING REMARKS AND NEXT STEPS.....	55
9	REFERENCES.....	56

List of Figures

Figure 1: C4IIoT high-level architecture.....	12
Figure 2: Cloud Layer	14
Figure 3: Field Gateway Layer	23
Figure 4: Edge Node Layer.....	25
Figure 5: Level 3 – Security by ML-based Behavioural Analysis & Cognitive Security Capabilities	27
Figure 6: Level 2 – Security by Horizontal device-to-device Communication	28
Figure 7: Level 1 – Hardware enabled security	31
Figure 8: C4IIoT edge node layer	35
Figure 9: Logistics 4.0 use case – microcontroller residing inside a container	36
Figure 10: Smart Factory use case – edge node structure	37
Figure 11: Smart Factory use case – passive personal card NFC token for personal identification	37
Figure 12: Field Gateway - Logistics 4.0 user case	39
Figure 13: Mobile operator infrastructure	39
Figure 14: Security functional areas.....	40
Figure 15: Field Gateway - Smart Factory use case.....	41
Figure 16: Cloud layer.....	42
Figure 17: Containerization	43
Figure 18: Securing C4IIoT access to Cloud Layer	44
Figure 19: C4IIoT Cybersecurity 4.0 framework	47
Figure 20: Blockchain network.....	49
Figure 21: Attribute-based Encryption (ABE).....	50
Figure 22: Data-flow	50
Figure 23: 3ACEs & Security Infusion Architecture	53

List of Abbreviations

ABE	Attribute-based Encryption
AES	Advanced-Encryption Standard
AGV	Automated Guided Vehicle
BTS	Base Station Receiver
CSA	Cloud Security Alliance
C&C	Command and Control
DISCO	Distributed SDN Controllers
EC	European Commission
GA	Grant Agreement
H2M	Human to Machine
IIoT	Industrial Internet of Things
IoT	Internet of Things
NVD	National Vulnerability Database
MCU	Micro-Controller Unit
MME	Mobility Management Entity
ML	Machine Learning
MNO	Mobile Network Operator
MVP	Minimum Sustainable Product
NFC	Near-field Communication
NPM	Node.js Package Manager
PKI	Public Key Infrastructure
PLC	Programmable Logic Controller
SDN	Software-Defined Networking
SGW	Serving Gateway
SGX	Software Guard Extensions
VM	Virtual Machine
WP	Work Package

Executive Summary

This deliverable provides a specification of the conceptual architecture of the C4IIoT platform. The C4IIoT project aims at building and demonstrating a new and integrated IIoT cyber-security framework for the detection and prediction of anomalies and malicious behaviour, as well as mitigation of detected threats and notification of end-users. The framework provides a security solution for minimizing the attack surfaces of IIoT systems using (i) modern security software and hardware protection mechanisms, (ii) state-of-the-art behavioral analysis based on machine- and deep-learning techniques, (iii) privacy-aware analytics, (iv) a novel encrypted network flow framework, (v) secure-by-design IIoT device fabrication and (vi) blockchain technology. These technologies will provide a viable scheme for enabling security and accountability, maintaining privacy, enabling reliability, and providing trustworthiness within IIoT applications.

The specification of the architecture is based on a detailed analysis of reference architectures, state-of-the-art literature review, end-user requirement analysis, as well as general non-functional requirements and best practices. These review activities were performed within deliverable 1.2 - positioning of C4IIoT, deliverable 2.1 (analysis of edge-node assets), as well as deliverable 7.2 - data management plan. C4IIoT architecture is built on top of the latest version of Reference Architectures proposed by ECSO (RAMI4.0 and IIRA) for Industry 4.0 [1]. This document can be updated and refined during the project lifetime, as consortium members experience becomes bigger during the process of its implementation and integration.

The deliverable contains information that includes a detailed description of all C4IIoT platform modules provided by technology providers, describes the interaction and integration of those modules, and describes actions that must be taken during the project lifetime to develop and integrate individual modules. The deliverable brings discussion related to the specificity of two cases of the C4IIoT platform (i.e., *Logistics 4.0* and *Smart Factory* use case) and draws implications to the architecture. Information from this deliverable provides initial integration guidelines for the C4IIoT Minimum Sustainable Product (MVP). Further architectural aspects of the C4IIoT platform may be revisited during the project resulting in corresponding updates - for instance, in task 5.2 - framework deployment and execution of real-life industrial demonstrations.

1 Introduction

This deliverable is part of Work Package 1 (namely, “Setting the Scene: Project Set Up”). Work Package 1 consists of four tasks that examine and discuss the critical roles of C4IIoT in protecting Industrial IoT systems and technologies (Task 1.1), adapting C4IIoT security components to real-life industrial manufacturing environments (Task 1.2), demonstration protocol and real life industrial pilots (Task 1.4), while Task 1.3 connects directly to this deliverable through analysis of convergence technology - specifications and C4IIoTC4IIoT architecture.

The specification is based on functional and non-functional requirements as well as the reference architecture, which build up from software engineering good practices [2] and the identified state of the art cyber-security tools and technologies. Within the scope of this deliverable, we give a detailed definition of C4IIoT platform modules provided by technology providers and describe the interaction between those modules, as well as the options for their integration. The modules have been grouped into three layers - the edge node layer consisting of devices close to the production lines and transport containers, the field gateway layer that aggregates and analyses data from the edge node layer and the cloud layer that represents the central component of the system in which advanced anomaly detection and mitigation take place. From a different point of view, orthogonal to described layers, we defined three levels of security - hardware based security, horizontal device-to-device communication security, and machine learning based behavioral analysis and cognitive security capabilities. The first and second security levels extend through the first two layers, while the third security level extends through all three.

Actions that must be taken during the project lifetime to develop individual modules and integrate them to the final version of the platform are discussed and defined within the deliverable. Moreover, data flow between the individual components are thoroughly described. The specification described in this deliverable will serve as an input for further development within which the implementation of the C4IIoT platform modules and their integration will be carried out. It is planned to develop three releases of the C4IIoT platform in the iterative process – a proof of concept - minimum viable product (MVP), a first prototype and a second prototype of the system.

The structure of the deliverable is as follows. Chapter 2 gives the overview of the C4IIoT architecture and provides an overall description of the architecture. Modules of the system are thoroughly described and positioned within the three layers of the architecture and the three levels of security. At the end of the chapter, we give a description of the simulated environment that will be used to better understand events occurring at the edge node layer. Description of individual modules includes information about their interaction with other modules. Next, Chapters 3-5 detail each of the architecture layers (respectively, Edge node layer, Field gateway layer and Cloud layer). User interface details are discussed in Chapter 6. Chapter 7 describes the three levels of security that C4IIoT offers, namely, hardware-based security, horizontal device-to-device communications security, and machine learning based behavioral analysis and cognitive security capabilities. Finally, Chapter 8 is the conclusion that gives insights into next steps in platform implementation and integration.

2 C4IIoT Architecture Overview

2.1 Overview of the Architecture

C4IIoT introduces a new comprehensive framework that aims to enable cybersecurity assurance in IIoT systems, referred to as the C4IIoT Cybersecurity 4.0 framework. The C4IIoT cybersecurity framework will be built upon techniques and methodologies to enable strong authentication and authorization models, code signature and verification models, secure execution, compliance, accountability, behavioural analysis, and privacy-preserving multi-party analytics. C4IIoT framework is organized into three layers: the edge nodes layer, the field gateway layer, and the cloud layer. Framework's architecture with the three levels and layers is presented in Figure 1.

The architecture is divided into three logical layers (Figure 1 right side) and three levels of security (Figure 1 left side). This stratification describes the same system from different aspects. Even though they are created according to different criteria, layers and levels have a relationship with each other. Levels 1 and 2 refer to the edge node layer and the field gateway layer, while level 3 extends through all three layers. The data is being fed through the edge node layer that contains two types of devices. Microcontrollers attached to containers that can be loaded to trucks or stored in warehouses. Microcontrollers use 2G/3G/4G mobile networks to communicate directly with the upper layer – field gateway residing within the mobile network operator server infrastructure. Although microcontrollers do not have much of a processing power, even at this level it is possible to have lightweight security anomaly detection. These devices cover the *Logistics 4.0* use case. The other type of edge node layer devices covers the *Smart Factory* use-case. This type of device includes single board computers (such as the Raspberry Pi [3]) that are directly connected to the devices in the factory. Their processing power is higher than of the previous ones, so at this level it is possible to execute more complex algorithms related to machine learning at the edge. Moreover, in this scenario, they connect to (possibly multiple) field gateways that reside in factories using WiFi or wired connection.

In the *Logistics 4.0* use case, where the microcontrollers connect to a field gateway that resides in a mobile network operator infrastructure using mobile network, there is just one field gateway. In the *Smart Factory* use-case though, there might be multiple gateways in one factory, as well as there might be multiple factories. In both cases, field gateways are strong server computers with much more processing power than devices at the edge. This is where intermediate anomaly detection takes place. In cases where advanced level processing is necessary, the data is offloaded from field gateways to the cloud layer through the cloud gateway, using SDN enabled switch/router and offloading mechanism – MEDICI tool.

The cloud layer is where advanced anomaly detection and privacy aware data analytics takes place using data aggregated and offloaded from the field gateways. In case an anomaly is detected, data is being fed to the mitigation engine. Mitigation engine decides if an action needs to be taken and search for a new configuration of the system, that is either resistant to the detected anomaly or at least minimizes the damage that it can cause. The system is then reconfigured through the SDN controller. At the same time, continuous verification tool is doing verification of IoT device firmwares and acts if a change is detected in that segment.

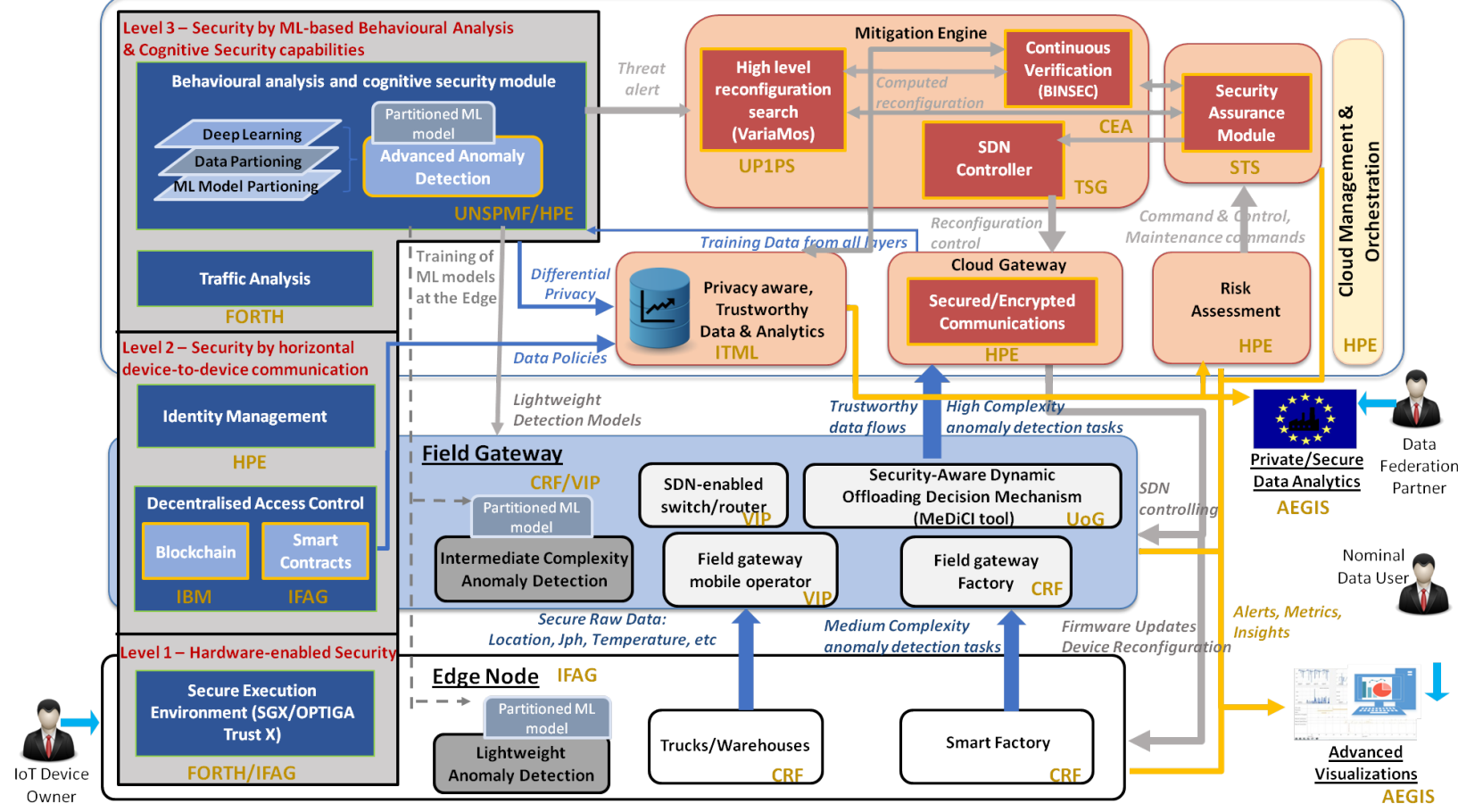


Figure 1: C4IIoT high-level architecture

The initial architecture of C4IIoT (as described in GA), has been modified in some aspects in order to represent better the corresponding technologies and the interactions with each other. First, the “Security Assurance” module that is provided by STS has been moved outside the “Mitigation Engine”, since it provides security assurance to the whole C4IIoT framework. Second, we have redefined the execution flows and interactions between the modules within the “Mitigation Engine”: in particular, UNSPMF’s anomaly detection module will provide input to VariaMos, which will conduct an extensive anomaly mitigation reconfiguration search. These reconfigurations will be used to setup the SDN controller and the BINSEC tool [4]. We note that the primary goal of CEA’s BINSEC tool is to perform anomaly detection in binary code, however it also provides mitigation capabilities by proposing patches to detected anomaly. Also, since VariaMos reasons at a high-level of abstraction to perform efficient search, it cannot entirely avoid the possible introduction of new vulnerabilities at the lowest level of binary code for the reconfiguration that it proposes. Whenever possible, it thus calls BINSEC to verify this new configuration at the binary code level, before calling the SDN controller to carry it out. Finally, the “Field Gateway” component is splitted in two cases: (i) in the *Smart Factory* scenario, it will be located within CRF’s premises and will communicate with the edge nodes that reside inside the factory. (ii) in the *Logistics 4.0* scenario, it will be located within VIP’s data center and will keep track of the sensors that are integrated inside the trucks and warehouses. The communications in the former case will be over wireless, while in the latter case they will use GSM (provided by VIP) to provide better covering, since these sensors are prone to location changes.

The effectiveness of the proposed C4IIoT architecture shall be demonstrated through two fundamentally different, yet intertwined use cases. In the first use case (*Smart Factory*), the focus is on monitoring parameters related to the assets such as welding cells on car manufacturing line, and Automated Guided Vehicles (AGVs). The data of interest for the welding cells is acquired from temperature, pressure and capacity flow sensors (for the cooling liquid of the robot's clamps), as well as from the accelerometers on the robot's slide. For the AGVs, the data of interest includes vibration, acceleration and velocity. The edge devices use short range Wi-Fi communication to deliver data to field gateways located inside the factory. The edge nodes used in this case have no hard constraints when it comes to the energy efficiency, and the small form factor. Also, in such an environment the likelihood of an unauthorized access to the hardware by malicious parties is significantly lower in comparison with the other use case, described below.

In this use case (Logistic 4.0), the logistic loggers are used to track, trace, and monitor the quality of components transported on containers from the supplier plant to the vehicle production plant. The edge devices are externally attached to the containers, and acquire data such as GPS coordinates, temperature, humidity, and data related to the shock/vibrations. The containers are usually handled by a logistic partner, and thereby spend significant time out of the reach of the car manufacturer. Having that in mind, the edge devices must be designed in such way that they utilize communication technologies such as cellular 3G/4G network, which enable connectivity all along the product trajectory. In addition, the battery powered devices must be power efficient in order to reduce maintenance costs. Finally, as the devices might be physically accessible to the potential attackers, they must utilize strong security mechanisms to prevent extracting confidential data, as well as sending false data to the field gateway/cloud.

2.2 Cloud Layer modules description

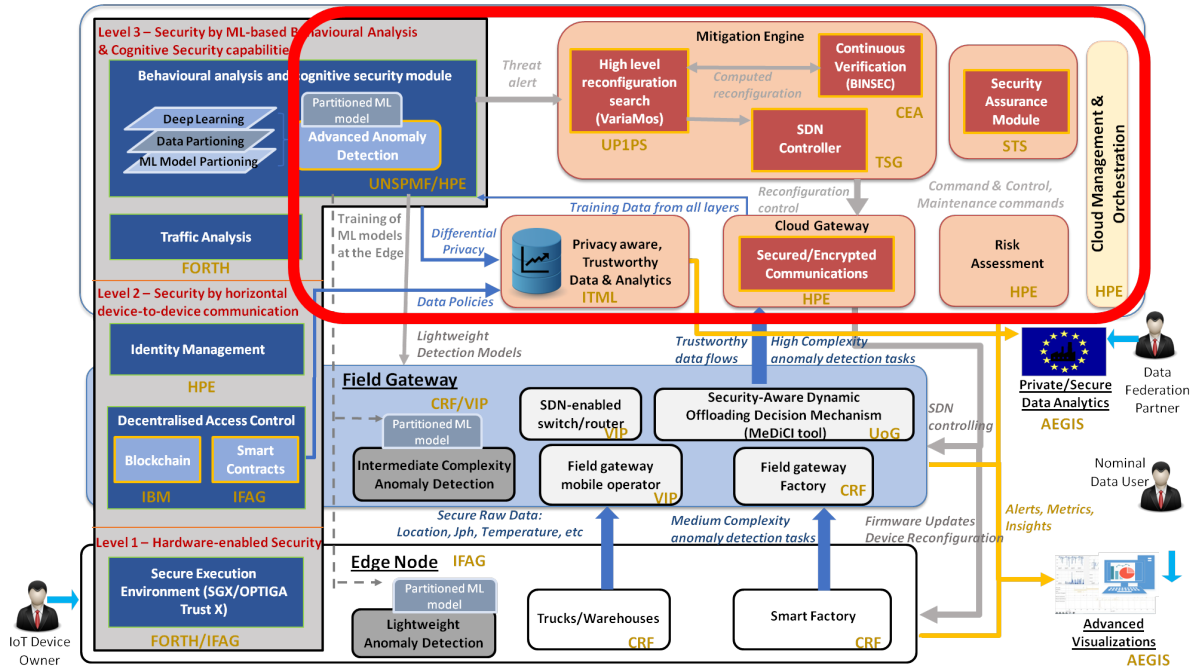


Figure 2: Cloud Layer

2.2.1. Mitigation engine – Reconfiguration Search (VariaMos)

Objectives / Brief description

This component is a tool to model:

- the operational context of a Self-Adaptive System (SAS) which should include instances of security concepts such as vulnerabilities, threats, attack methods and alerts, current deployment of logical components/resources/assets instances onto physical nodes/resources/assets, current communications among these instances;
- its context-aware requirements, which should include security and other dependability (safety, availability, integrity, confidentiality, etc.) hard goals/constraints and soft goals/constraints;
- its configuration space in terms of assemblies and parameter choices of implementation assets reusable to satisfy the requirements which should include concepts such as logical component/asset classes, dependencies and constraints among them, how they can be assembled, how they can individually or collaboratively contribute to the hard and soft functionalities, dependability and security goals, physical deployment nodes, the hardware, software, sensing, actuating, computing resources/assets available at each network node, the communication/networking resources/assets available at each node and the ongoing communication/connections resources

It provides services to automatically verify whether the current configuration satisfies the requirements in the current context and if it does not search for an alternative configuration that does (if it exists).

Technical pre-requisites and requirements

The desktop version requires Java and SWI-Prolog [5] to be installed locally.

The back-end of the web application under development requires the installation of SWI-Prolog with the extension pack Logtalk [6] to support *object-oriented* constraint logic programming. The front-end, developed in Vue.js [7] and Axios.js [8] is only needed to visually edit the context, requirements, assets and configuration models.

Interactions

The needed input to the component consists of a context model, a context-aware requirements model, a reusable asset model and a current configuration model, in JSON format for the web version or in XML format for the desktop version.

Having the specified input, the component performs either a verification of an input configuration or a search for an alternative configuration.

The output for a verification task will be a Boolean.

The output for an alternative configuration search will be either a configuration model or fail if there is no configuration of the input reusable asset model that satisfies the input requirements model in the input context model.

2.2.2 Mitigation engine – SDN controller

Objectives / Brief description

The SDN controller takes mitigation orders and translates them to policies pushed on the SDN-enabled switch/routers in the Field Getaway. It can also monitor the switch/routers to ensure the policies are applied and gather some metrics on the data plane (e.g. throughput, latency, etc.).

Technical pre-requisites and requirements

The SDN controller is an OS-agnostic binary. It must have network access to the SDN-enabled switch/routers deployed in the Field Gateway with minimal latency.

Interactions

Inputs are provided via a REST API; they include the SDN-enabled switch/router to update as well as the decision to apply (e.g. block flows, redirect flow, etc.). The southbound interface will use the OpenFlow protocol.

2.2.3 Mitigation engine – Binary code analyzer (BINSEC)

Objectives / Brief description

The binary code analyser will support a continuous verification based on events happening on the IIoT system, such as firmware and software update, recording of an attack targeting a specific binary, etc. This continuous verification can be done by exploiting the results of an initial verification effort of the sensitive binaries of the system.

Technical pre-requisites and requirements

For the module to run, the BINSEC tool will need to have access to the binary executables files that should be executed on the edge node, in a standard format (e.g. ELF). The executable should be in one of the instruction set supported by BINSEC (e.g. Intel X86-32 bit, ARMv7, RISC-V).

In addition, the BINSEC tool can be fed with an input, modeling the interactions with the binary (e.g. commands or data sent to the binary, or sequence of instructions executed by the

binary), that will be used to target the analysis to specific traces in the binary. This can be used to have a more thorough analysis of a specific event (such as a detected potential threat).

BINSEC can benefit from debugging information contained in the executable, so it is better if applications compiled to be analyzed leave these debugging information (e.g. gcc compilation with the -g option).

For continuous verification, the BINSEC tool will also need access to the history of binary, so that it can find the difference between different versions of a binary and concentrate its analysis on these parts.

Interactions

The module will need to:

- receive the binaries executed on the edge node, preferably with debugging information;
- have access to a history of the different versions of a binary executable;
- may use a specific input of a binary to target the analysis.

It can output:

- different analyses about a binary, such as control flow graph, invariants, certified absence of error in parts of the code
- specific input traces that trigger a bug or vulnerability, with the location of the bug.

2.2.4 Security Assurance Module

Objectives / Brief description

The Security Assurance Module is an integrated framework of models, processes, and tools to enable the certification of security properties of services. It uses different types of evidence to demonstrate the support for the required properties and award the corresponding certificate. The types of evidence, which have been envisaged for our framework, include monitoring and testing data. The Security Assurance Module will be used for monitoring, testing and assessing the C4IIoT framework. It will also be responsible for monitoring the execution times and accuracy of each and every component of the platform, ensuring and assuring the proper functioning of the whole C4IIoT framework. The Security Assurance Module:

- combines runtime monitoring and dynamic runtime testing to ensure correct and effective operation of security controls
- can be hooked to different systems programmatically through appropriate probes (e.g., event captors, test tools) in order to obtain the monitoring and/or test evidence required for assurance and/or certification assessments
- operates based on models that determine the operational evidence that should be captured from systems and how it should be assessed (e.g., what conditions it should satisfy) in order to assess the correctness and effectiveness of implemented system security controls
- enables the runtime assessment of temporal event patterns and rules that can express signature or anomaly-based patterns.

Technical pre-requisites and requirements

The main components of the Assurance Tool are the following:

- **Security Assurance Loader** is the component responsible for receiving the security assurance model for the target organization. This model includes the assets of the organisation, security properties for these assets, threats that may violate these properties and the security controls that protect the assets.
- **Monitoring Manager** is the component responsible for initiating, coordinating and reporting the results of the monitoring process.
- **Certification Generator** is the core of the assurance tool and provides management capabilities for its operation, such as sufficiency conditions, anomalies, conflicts and life cycle management.
- **Certification Manager's** main functionality is to begin the certification process, receiving requests from the application GUI/dashboard and selecting the appropriate underlying tools to carry out the certification.
- **Certification Communicator** offers the means for getting the generated certification, via the Retrieval API, for both the service consumer and the CA/Cloud Service Provider.
- **Vulnerability Loader** loads the known vulnerabilities and updates the Security Assurance Module depending on the organization's assets included in the assurance model.

Databases:

- Monitoring Database, that holds the monitoring results and the monitoring certificates.
- Evidence DB, that holds evidence and detailed evidence aggregated by the monitoring Manager and used to generate the Certification.
- Security Assurance Database that holds the security assurance model and its components.
- Vulnerabilities database that holds the known vulnerabilities from NVD.

External components utilized by the Security Assurance Module are the following:

- **Monitoring Module (EVEREST)**, is a runtime monitoring engine built in Java that offers an API for establishing monitoring rules to be checked. The module is made of three submodules: a monitor manager, a monitor and an event collector. The role of the module is to forward the runtime events from application's monitored properties and finally obtain the monitoring results.
- **Virtual Machines (VM)** and other cyber infrastructures, to enable the continuous assessment of the security of cyber systems through the combination of runtime monitoring and dynamic testing, in order to provide information about the status of the actual cyber systems and support the range of simulations required by CTP models and the interaction with the emulated mechanisms of the THREAT-ARREST platform.
- **Message Broker (RabbitMQ)**, a messaging bus that allows communication between the external components and the Security Assurance Module.

Interactions

- **addAssuranceModel:** This method allows the end-user to load the assurance model to the Security Assurance Module.
- **addCertificationModel:** This method allows the end-user to load the certification model to the Security Assurance Module.
- **DeleteCm_Monitoring:** This method deletes the CM.
- **updateAssuranceMode:** This method updates the assurance mode or part of it.
- **startMonitoring:** This method allows to start the monitoring process accordingly to the given assertions. The results of the process are going to be provided asynchronously to the caller.
- **subscribeToEventBus:** This method subscribes the monitor to a predefined channel to listen to the results from the event captor.
- **getKnownVulnerabilities:** This method loads the known vulnerabilities for a specific asset described in the assurance model.
- **Access Control API:** It will allow the authentication and authorization of users.
- **Retrieval API:** It will enable sending requests for certified components, checking the validity of certificates, and getting runtime-related information.
- **sendResults:** This method distributes the results within the mitigation engine.
- **Risk Assessment API:** This method will allow the communication between the Risk Assessment tool and the Assurance one.

2.2.5 Risk assessment

Objectives / Brief description

This module is in charge of evaluating and assessing the risks detected by the C4IIoT platform, through a methodology-based structured approach able to measure the business impact of the risk. It will decide if a given risk can be tolerated by the system, if mitigation actions are needed, or if deeper reconfigurations must be considered to keep the system secure. The risk assessment methodology will be based on the recognized ISO/IEC 27001 standard.

Technical pre-requisites and requirements

-

Interactions

The risk assessment module has an inbound interface towards the Trustworthy Data & Analytics module, through which it fetches the data (metadata) to elaborate for producing its output.

It has an outbound interface towards the Mitigation Engine module, namely to the Security Assurance component. Through this interface, the risk assessment passes through the results of its evaluation, so that the Security Assurance can elaborate it and trigger further mitigation actions if needed.

Additional information

All the above description is tentative, to be discussed and validated by WP4.

2.2.6 Cloud gateway

Objectives / Brief description

The cloud gateway is the module in charge of ensuring effective and secure data exchange between the cloud layer modules and the field gateways located in the Layer 2. It has two main jobs to perform, tied to the upstream and downstream data flows moving across the C4IIoT framework:

- On the upstream side, it must ensure that the trustworthy data flows coming from the field gateways are properly passed into the Cloud layer, stored and made available to the high complexity anomaly detection modules, which process them to generate the behavioural models.
- On the downstream side, the gateway forwards the C&C and maintenance outputs generated by the Mitigation Engine towards the field gateways and/or directly to the field devices, to trigger the proper reconfiguration and upgrade actions.

Technical pre-requisites and requirements

The Cloud Gateway main component is an API gateway, exposing to both its internal (i.e., inside the cloud layer) and external counterparts the needed functionalities to send/receive secured data flows, and have them dispatched to the proper final destination. It also might need to embed an encryption/decryption component, needed to ensure the gateway functionality even in presence of encrypted data flows at its southbound interface.

In compliance with all the other architectural constraints and requirements, it is highly wanted that the gateway interfaces, or the most part of them, is implemented by REST API, for the sake of flexibility and maintainability. In terms of latency, the functions provided by the cloud layer don't require a real-time response. It's needed nevertheless that the layer 2 downstream interface assures to handle the throughput of the data flows incoming from the field gateways.

Interactions

The Cloud Gateway has the following inbound interfaces:

- interface towards Mitigation Engine: through this interface, the Cloud Gateway receives C&C and Maintenance commands, to be properly encrypted and dispatched to the right destination (Field Gateway or Edge Node)
- interface towards Field Gateways: through this interface, the Cloud Gateway receives the data flows to be encrypted and forwarded into the Cloud layer, for the Behavioural Analysis module to process them

and the following outbound interfaces:

- interface towards Field Gateways: through this interface, the Cloud Gateway sends the C&C and Maintenance commands addressed to Field Gateways (e.g., commands to configure the SDN control plane). It can be expected that most commands crossing this interface are of C&C type.
- interface towards Edge Nodes: through this interface, the Cloud Gateway sends the C&C and Maintenance commands addressed to Edge Nodes (e.g., firmware patches). It can be expected that most commands crossing this interface are of Maintenance type.

- interface towards Behavioural Analysis: through this interface, the Cloud Gateway informs the Behavioural Analysis module of the sends the C&C and Maintenance commands addressed to Edge Nodes (e.g., firmware patches). It can be expected that most commands crossing this interface are of Maintenance type.

Additional information

All the above description is tentative, to be discussed and validated by WP3.

2.2.7 Cloud Management & Orchestration

Objectives / Brief description

This module must perform the proper handling and orchestration of physical and virtual resources available in the C4IIoT cloud infrastructure, in order to ensure the reliable provisioning of C4IIoT services executed in the cloud layer. The module looks after first instantiation of the components running in the cloud layer (Mitigation Engine, Behavioural Analysis, Data & Analytics).

The resource configuration and orchestration functions must be provided by proper automation tools, part of the specific cloud and container platforms that will be selected in the design and implementation phase. The expected deployment environment, for portability and flexibility reasons, will be based on containers.

Technical pre-requisites and requirements

This module must be able to provide resource management and orchestration functions agnostically with respects to different cloud platforms (e.g., AWS [9], Azure [10], OpenStack [11]) and container deployment/orchestration engines (e.g., Docker Swarm [12], Kubernetes [13]). This module aims at abstracting such functionalities, to avoid locking the C4IIoT platform to specific cloud and/or containerization environments.

This module must rely on one or more repositories, storing and keeping up to date the software images of the components to be provisioned in the cloud layer.

Interactions

This module is fully internal to the cloud layer, and is interconnected to the different cloud and containerization platforms making up the actual C4IIoT instance. As said, the connection goes through an abstracted interface layer, that will provide access to the standard functions requested to manage and orchestrate the environment, like e.g.:

- configure deployment nodes
- instantiate/de-instantiate containers
- orchestrate containers into clusters
- infrastructure monitoring
- scale in/out virtual resources.

Additional information

All the above description is tentative, to be discussed and validated by WP3.

2.2.8 Privacy aware, Trustworthy Data & Analytics

Objectives / Brief description

This module provides privacy-preserving storage and analytics to end-users by allowing seamless integration and injection of heterogeneous data, and facilitate the adoption of collaborative analytics to enterprises, without exposing private or sensitive information, achieving at the same time cybersecurity threats' identification and mitigation. The module:

- allows the user to select which data (or portion of the data) will be used, via a policy language that will protect data and selectively allow access according to user defined criteria.
- allows on demand analytics acquisition for whole datasets using Machine Learning algorithms in batches or real-time
- provides encapsulated functionality for analyzing heterogeneous data, from any technical source, providing insights with short or no training period allowing for an iterative approach that constantly improves results, utilizing all available data sets, and applying a plethora of algorithms for providing data analytics.
- provides privacy preservation, threat's identification and vulnerability assessment with the performance of machine learning algorithms in question.

Technical pre-requisites and requirements

Current ITML's modules that will be extended within C4IIoT framework:

Analytics Aspect:

- Can accommodate any modern data feed, through an open, extensible and scalable adapter factory that hosts any kind of adapter, from traditional relational databases, No-SQL databases or even Kafka™ streams.
- Can manage all the available data streams, through the use of opensource tools like Apache Web Application server [14], Laravel [15] and a custom adapter factory accommodating all aforementioned supported data sources pre-process the data.
- Modules are built in a Docker™ [16] container allowing for easy installation either on the client's premises or any available cloud infrastructure.

Security Aspect:

- The service can be accessed either on-site or via a VPN to a cloud infrastructure.
- The agents that are responsible for collecting the data are for network nodes running Windows, Linux and MacOS.
- Data is gathered to market's most well-established security and data analytics tools, including Kibana [17], syslog, Nagios [18] and Elastic-search [19], as well as OpenVAS [20] and Greenbone [21], in order to extract all the valuable information and deliver it to the central Security Center User Interface.
- Laravel™ [15] plays the role of the orchestrator between different streams of information, providing a comprehensive way to depict numerous screens / functionalities, encapsulating features from the aforementioned frameworks.

Interactions

The **Privacy aware, Trustworthy Data & Analytics module** will extend ITML's existing mechanisms and tools to be integrated to the C4IIoT framework and will have an interface towards the:

- **Advanced Visualization Toolkit module** [22] regarding interactive visualizations in a business-style dashboard of the privacy-enabled and security-related situational analytics and awareness for the decision makers, information that comes from the runtime operation of the platform.
- **Behavioural Analysis & cognitive security module:** Data flows to be processed and analyzed from the behavioral analysis module into the privacy aware module.
- **Risk Assessment module:** Privacy aware, Trustworthy Data & Analytics module's output will be evaluated and assessed through a methodology-based structured approach able to measure the business impact of the risk.

Additional information

All the above description is tentative, to be discussed and validated by WP3.

2.2.9 Advanced visualisation, Private/secure data analytics

Objectives / Brief description

The Advance visualization module offers interconnected, interactive visualizations of identified events and indicator values. It enables situational awareness of end users and helps them identify patterns or correlations of indicators which can foster the decision-making process. This way, end users can get a situational awareness of the system (where original data come from) at any given time. They can check to see if any abnormal situation seems to be about to happen (or already took place) and generally search for patterns in events and indicator values that can reveal correlations and help them decide on appropriate actions.

Technical pre-requisites and requirements

Java/PHP applications used in the business layer. APIs/other methods (e.g., direct access) are required to get the Indicator values and the generated events. A JavaScript-based web application offers the frontend of the Advance visualization module. End-users only need a browser to access Advance visualization module.

Interactions

Input of the component must be provided as an API or via some other method, e.g., direct access to the relevant data. Main inputs are (TBD, domain-specific):

- Indicators (and relevant values);
- Events;
- The tool can be also used by users to trigger actions. These actions will mostly include selection of data sources and time periods. The data retrieval will take place using one of the aforementioned methods that will be available in the project.

2.3 Field Gateway Layer modules description

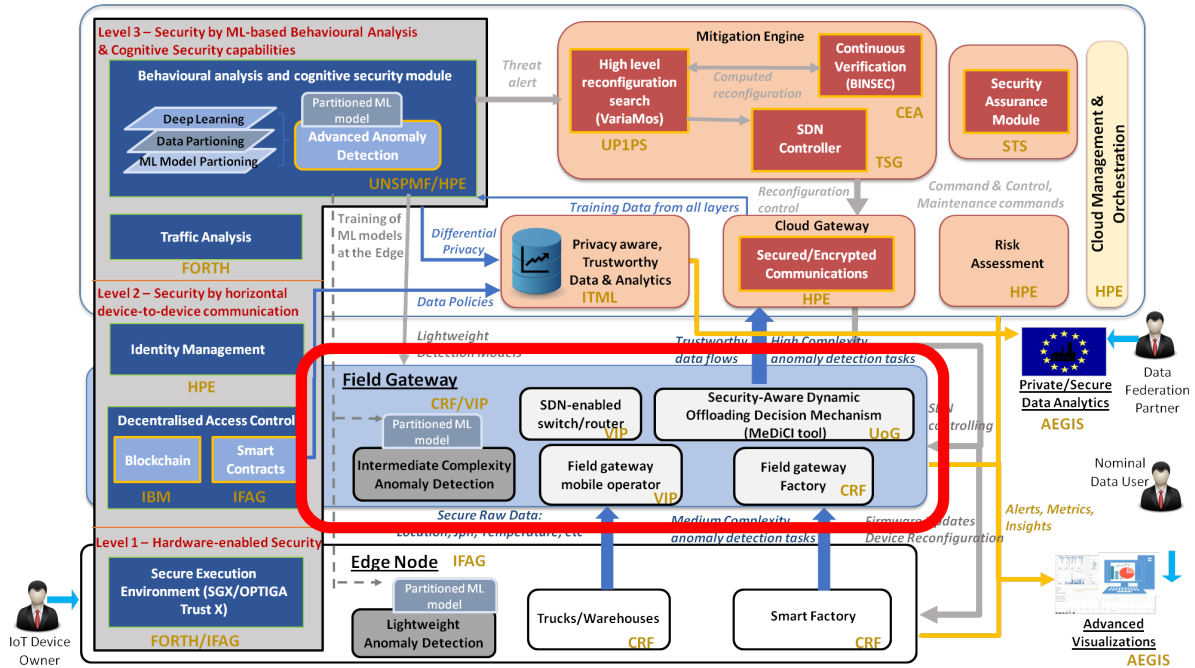


Figure 3: Field Gateway Layer

2.3.1 SDN-enabled switch/router

Objectives / Brief description

SDN-enabled switch/router is a component that is a part of the field gateway and is used for supporting DISCO [23] (an implementation of TSG' multi-controller SDN architecture).

The SDN-enabled switch/router offers data plane connectivity to the network while being connected to the controller, which configure the data path. Contrary to a legacy switch/router, it offers an API to dynamically alter the switching/routing tables depending on the policies pushed by the controller.

Technical pre-requisites and requirements

SDN-enabled switch/router has to be software based, implemented as routing module within the field gateway. In order to establish communication with the SDN controller, a Flow Control Protocol has to be implemented and it could be OpenFlow, BGP-LS, PCEP or any other protocol which is supported by DISCO solution that is part of the mitigation engine. Traffic routing will be implemented on IP level.

It also has to be integrated with other field gateway functionalities, especially with the management module. SDN-enabled switch/router should be capable of standalone deployment for testing purposes (simulating multiple nodes, i.e.)

The SDN-enabled switch/router needs a Linux platform for installation. The controller must be accessible from the network with minimal latency.

Interactions

There are three types of interfaces for this element:

- interface to SDN controller (DISCO), hosted in the cloud, in order to receive commands to perform traffic switching;

- multiple traffic interfaces – sources and destinations for traffic being switched
- operations management interface for alarms, provisioning (could be both internal – towards field gateway and external – towards cloud and mitigation engine)

Additional information

The SDN switch can only be operated via the field gateway. However, synchronization between the main field gateway logic and this component is required in each case.

The input for the SDN-enabled switch/router are the rules/policies pushed by the SDN controller via a northbound API (e.g. OpenFlow, NETCONF [24]). The switch/router can also provide indicators and events through this interface. The SDN-enabled switch/router will also handle the traffic from users on the data plane.

2.3.2 Security-aware dynamic offloading decision mechanism

Objectives / Brief description

The security-aware dynamic offloading decision mechanism will provide a dynamic way to decide whether to offload the computation of anomaly detection in a more powerful machine than the edge node or running it locally. It will sit on the C4IIoT field gateway and interact with the deep learning trained models that will perform detection of complex anomalous and malicious behavior. It will make dynamic offloading decisions based on the trade-off between confidence in anomaly detection (obtained by the output of the reasoning mechanism running deep learning models at the edge) and the cost of offloading to the cloud (considering a heterogeneous set of multiple criteria, such as network latency, computation/response time and energy consumption).

Technical pre-requisites and requirements

The security-aware dynamic offloading decision mechanism will extend UOG's existing energy-aware decision support mechanism (MEDICI tool) to become also security-aware and specifically adapted for the requirements of IIoT and the C4IIoT use cases. Each device running the MEDICI tool needs to have Linux OS.

Interactions

It will interact with the deep learning trained models that will perform detection of complex anomalous and malicious behavior and provide a confidence detection value as an input to the decision mechanism.

2.3.3 Field gateway

Objectives / Brief description

Field gateway is an intermediate layer of our architecture, acting as an element which communicates with edge nodes on the one side and the cloud on the other. This is the element with increased computational power to perform security-related (e.g. anomaly detection) actions. Significantly lower latencies are expected compared to the cloud since it resides closer to the edge nodes. It will support both edge nodes that utilize wired network connectivity (local or internet) as well as those based on cellular protocols.

The role of the field gateway in C4IIoT is dual. Besides acting as a (SDN)-enabled network node it will also include a local offloading/outsourcing decision mechanism.

Technical pre-requisites and requirements

Field gateways will be located in mobile operator premises for the logistic case and in CRF premises for the smart factory case, protected in both cases by a strong security network infrastructure. The field gateway possesses intermediate computational power compared between the low power edge layer and the high power cloud layer.

Minimum HW requirements for Field Gateway in both cases should be:

- CPU: Intel CPU with 4 Core, 3 GHz
- RAM: 4 GB
- Operating system: Ubuntu 16.04.6 LTS Xenial Xerus or any other Linux distribution

Interactions

Edge node will be directly accessible only from the Field Gateways. This way, no data exchange between devices will leave the trusted mobile network infrastructure. Only data sent during offloading to the cloud procedure will reach the Internet.

Edge node will send data upwards directly to the Field gateways where data will be analysed and checked for anomalies using centralised rules and black lists. Only traffic that does not contain malicious signatures or suspicious patterns will be passed downward to other devices/systems. The Field Gateway will also filter traffic in order to protect vulnerable or unpatched devices firmware, to keep devices functional and safe. In order to make secure connection with cloud infrastructure other security mechanism will be applied, like VPN techniques and firewall rules.

2.4 Edge Node Layer modules description

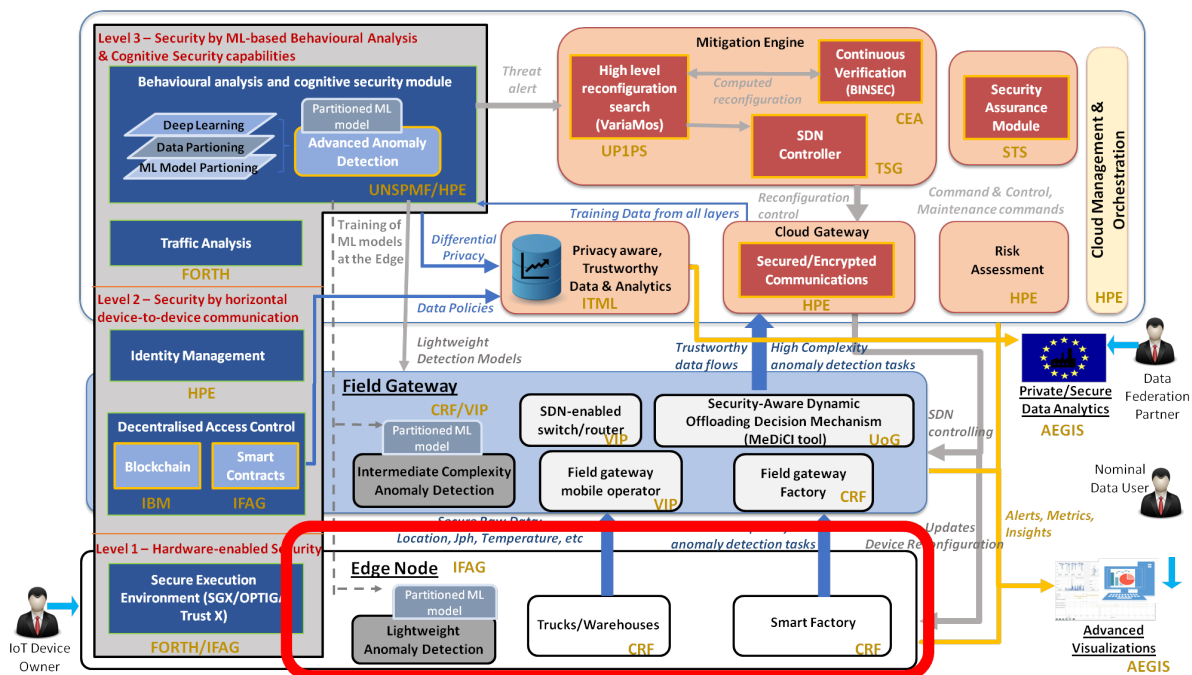


Figure 4: Edge Node Layer

2.4.1 Trucks/Warehouses and Smart Factory Edge Devices

Objectives / Brief description

These include IIoT devices such as sensors, which measure and/or detect events (e.g. temperature, motion, etc.) and transmit this information to be processed elsewhere, actuators,

which allow the control of mechanisms or systems and Safety Instrumented Systems (SIS) consisting of special sensors, safety PLC with a safety analysis hardware logic solver² and actuators which bring a process to a safe state if there is any violation of some predetermined conditions.

The edge devices include so-called smart robots, sophisticated industrial robots designed to perform complex tasks with smart capabilities, such as learning from errors and improve their performance. An example is the AGV, portable robots that navigate following marked lines or wires on the floor, and are equipped with vision cameras, lasers or radio waves, etc.

Technical pre-requisites and requirements

The information produced by the IIoT devices and robots needs to be gathered and safely stored in specialised systems/hardware.

Dedicated machines, called servers, provide functionality for other programs or devices, called clients. Those functionalities might include sharing data or resources between multiple clients, or performing computation for a client using specialised software, or simply gather and safely store information produced by the clients (in the Industry 5.0 content, these might be the IIoT devices and robots).

The following are some examples of the servers and systems used in industry 4.0:

Historians are software systems that gather data from industrial devices and store them in specialised databases.

Application Servers are computer machines that host specialised applications, such as user workstation applications, IoT web applications, etc.

Database Servers are servers used as repositories to store information provided by sensors, agents and management servers.

Enterprise operations systems are systems that integrate information from various parts of an organisation (i.e. manufacturing, distribution, finance, human resources, etc.). They can also provide connections with customers or suppliers. Examples include the Enterprise Resource Planning (ERP), Customer Relationship Management (CRM) and Supply Chain Management (SCM) systems.

Manufacturing operations systems are systems used to manage end-to-end manufacturing processes to optimise efficiency and improve the production output. An example is Manufacturing Execution Systems (MES) which can track in real-time and document the transformation of raw materials to finished goods.

² Note that here “logic solver” refers to an electronic circuit and not to a *software* constraint logic programming solver such as the one used inside VariaMos at the cloud layer.

2.5 Level 3 – Security by ML-based behavioural analysis & cognitive security capabilities – modules description

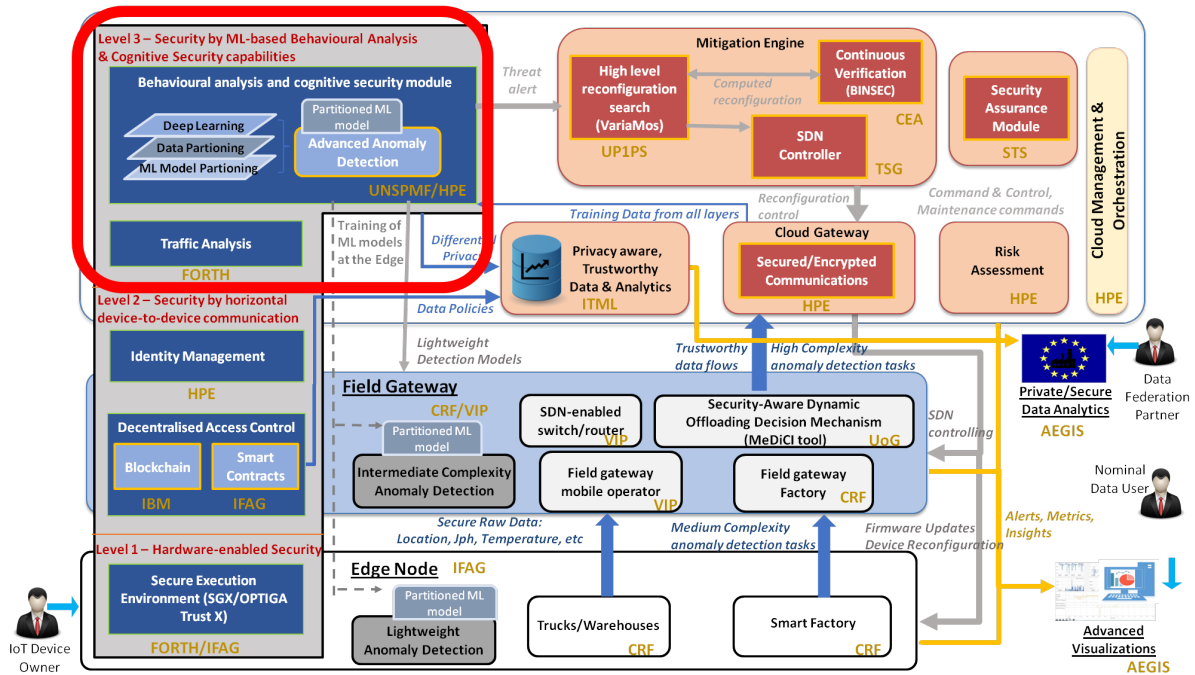


Figure 5: Level 3 – Security by ML-based Behavioural Analysis & Cognitive Security Capabilities

2.5.1 Level 3 – behavioural analysis & cognitive security

Objectives / Brief description

This module will implement the behavioural models within SGX [25] in order to preserve privacy and data protection even when using a third-party, not necessarily trusted, service provider. The behavioural models will be stored in private regions of memory called secure enclaves. This type of memory is protected even from processes running at administrative-privilege levels. Being inside an enclave, both code and data are encrypted in protected areas of execution memory.

Technical pre-requisites and requirements

Intel SGX must be provided in the machine that will run this module and UNSPMF's behavioural models must be written in C, with an option to use Python as well.

Interactions

The same as the module with the models of UNSPMF which will be implemented on SGX.

2.5.2 Level 3 – Traffic analysis

Objectives / Brief description

FORTH aims to generate signatures for intrusion detection using strictly packet metadata extracted from packet headers, making the intrusion detection engine able to handle encrypted network traffic. More specifically, the generated signatures are formulated properly to locate sequences of packet payload lengths inside network flows. The core operation of the intrusion detection engine is pattern matching; FORTH utilizes the Aho-Corasick [26] string searching

algorithm to provide an extended version of the pattern matching technique able to match integer values (packet sizes). Our module will analyze encrypted traffic and report anomalies or attacks to the anomaly detection modules.

Technical pre-requisites and requirements

Linux and OpenCL [27] are needed for the machine that will conduct signature matching. Also a dataset with malicious traffic will be used as ground truth.

Interactions

Our module will receive all upstream and downstream data flows (edge to field gateway, field gateway to cloud, command and control channel). Its output will be network signatures based on packet metadata. By conducting signature matching it will report attacks to anomaly detection modules.

2.6 Level 2 – Security by horizontal device-to-device communication – modules description

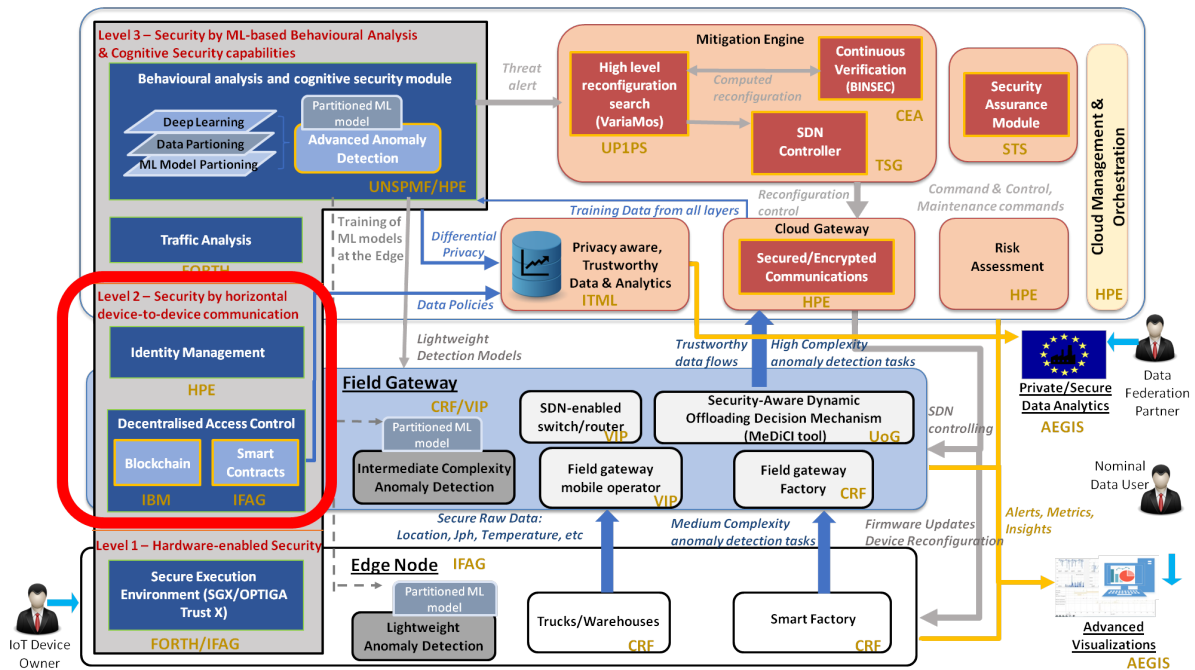


Figure 6: Level 2 – Security by Horizontal device-to-device Communication

2.6.1 Level 2 – Identity management

Objectives / Brief description

This module must provide smart identity management services to be pervasively used by the entities making up the C4IIoT platform, to ensure flawless and intelligent access control at all the existing system layers. It must also provide an integrated PKI infrastructure for C4IIoT, able to seamlessly cope with the different devices, certification schemes and to support advanced authentication policies, to fully exploit the risk detection potential inside the C4IIoT platform.

Technical pre-requisites and requirements

The identity management module must be able to scale, for handling a significant volume of digital certificates. It must be able to manage different typologies of credentials and authentication methods (digital tokens, smart cards, etc.).

A key pre-requisite of the identity management module is GDPR compliance: the identities' privacy and protection level must be granted, and fully regulation compliant.

A credential repository, storing all the identity data, keys, certificates, and so forth must be deployed and its access strictly controlled.

Interactions

The identity management module has a unique interface, internal at Level 2, towards the Decentralized Access Control module. Through this interface, it receives requests to create, delete or modify new identities, and returns confirmation of any requested action. The access to the credential store is considered as part of this interface.

Additional information

All the above description is tentative, to be discussed and validated by WP4.

2.6.2 Level 2 – Decentralized access control

Objectives / Brief description

- **Data integrity:** By using the Secure Element for message digest, signature and seal data with a digital stamp. In this way, the involved entities in data communication have the proven status that the data is not forged, especially when transmitted through a public network.
- **Confidentiality:** This is to protect data against unauthorized access and grant permission to the authorized entities or users. Secure Element in this case usually does not directly encrypt big data but secure keys for encryption, e.g. acting like a safe locker. Typical implementation is to utilize PKI [28], e.g. encrypt the data with the public keys of these users who shall have the access permission. This encryption is individual, meaning to include new users encryption has to be performed again. Simpler approach is to use symmetric keys to encrypt data. Afterwards, all with the right key can access. To manage user level individual access is then much more difficult.

Technical pre-requisites and requirements

Two main frameworks should be considered in this context.

- The industrial environment from CRF. These can be generally categorized in two cases. The first case stationed means that machine-to-machine communication happens in an automated way. The second case is mainly characterized by the involvement of users like technicians who travel with technical devices and transfer work data through such mobile devices.
- IBM's proposed data model for Decentralized Access Management, that specifies which kind of data to distribute, how to track them, where to store the critical keys, how can they be used and managed safely.

These interactions and relationships are illustrated in the system models

- “System Landscape and Use Cases - M2M [29] Communication/Embedded Security”

- “System Landscape and Use Cases - H2M Communication / Mobile Security”.

Interactions

In case of machine-to-machine communication, the common interface is I²C [30]. Secure Element acts usually passive, meaning the control and processing unit from the industrial robot should “talk” to Secure Element first and only after positive response should it then give security relevant tasks.

In case of human-to-machine communication, the Secure Element is usually built into a mobile token, which have interfaces like BLE, USB, NFC to connect to mobile devices. Depending on the complexity of token design, e.g. in terms of additional button, biometric sensor, display etc. The Secure Element shall support I²C, SPI [31], GPIO [32], ISO14443/NFC [33] etc. Eventually the Secure Element shall act as master for example in SPI communication.

2.6.3 Level 2 – Decentralized access control with attribute-based encryption (ABE)

Objectives / Brief description

The decentralized access control solution will consist of a few elements. Data generators (e.g. Edge Nodes) will encrypt data items containing data they wish to share, in accordance with privacy-aware policies, using attribute-based encryption (ABE). The encrypted data items will be uploaded to the cloud, either directly or by other entities (data distributors, e.g. Field Gateway, Cloud Gateway). HyperLedger Fabric (Blockchain) [34] network will be used by all data generators, distributors and consumers, as clients, in order to distribute links to newly uploaded data files on the cloud, allowing decentralized communication, transactions audit and verification of data integrity. Hyperledger Fabric peers, each storing a copy of the ledger, will be installed on various entities throughout the network. Data consumers (e.g. analytics) with the appropriate attributes will use the links stored on the Hyperledger Fabric network to approach the files on the cloud, download the data and decrypt it using their personal secret key issued by the ABE key-issuing component.

Technical pre-requisites and requirements

Each entity running the HyperLedger Fabric peer node (data distributors and consumers) should have Linux OS and support the prerequisites specified at [35], including docker and docker compose, cURL, Go [36], Node.js [37] and Python 2.7.

Additional platforms are required in order to run HyperLedger Fabric’s ordering service nodes, additional peers and organizational administrations and certificate authorities (with the same prerequisites as above), as well as to run the ABE key-issuing component.

Interactions

Data generators, distributors and consumers will communicate with each other through a HyperLedger Fabric channel. Data generators and distributors will initiate transactions to peers in order to store or update records on the Fabric and data consumers will query the Fabric peers and retrieve records. Ordering service nodes will interact with the Fabric peers as well.

Data generators and consumers will interact with ABE library to encrypt and decrypt data, and with ABE key-issuing component to be granted with ABE keys.

Data generators, distributors and consumers will interact with certificate authority to be granted with credentials to the Fabric.

Additional information

Usage of ABE as part of IBM's solution relies on an external library (OpenABE) released under AGPL-3.0 license. The OpenABE library [38] will be used as a non-linked “black box”, it will not bound any other C4IIoT code to AGPL-3.0 license, and is approved to use in C4IIoT by IBM legal under the condition of obtaining consent from all C4IIoT partners.

The question of whether we will deploy Hyperledger Fabric peers on the edge nodes is TBD. This question depends on whether the edge nodes are resources-capable of running Hyperledger Fabric peer and whether running Hyperledger Fabric peers in the field gateways layer and not in the edge nodes is sufficient with respect to C4IIoT requirement of device to device communication.

2.7 Level 1 – Hardware enabled security

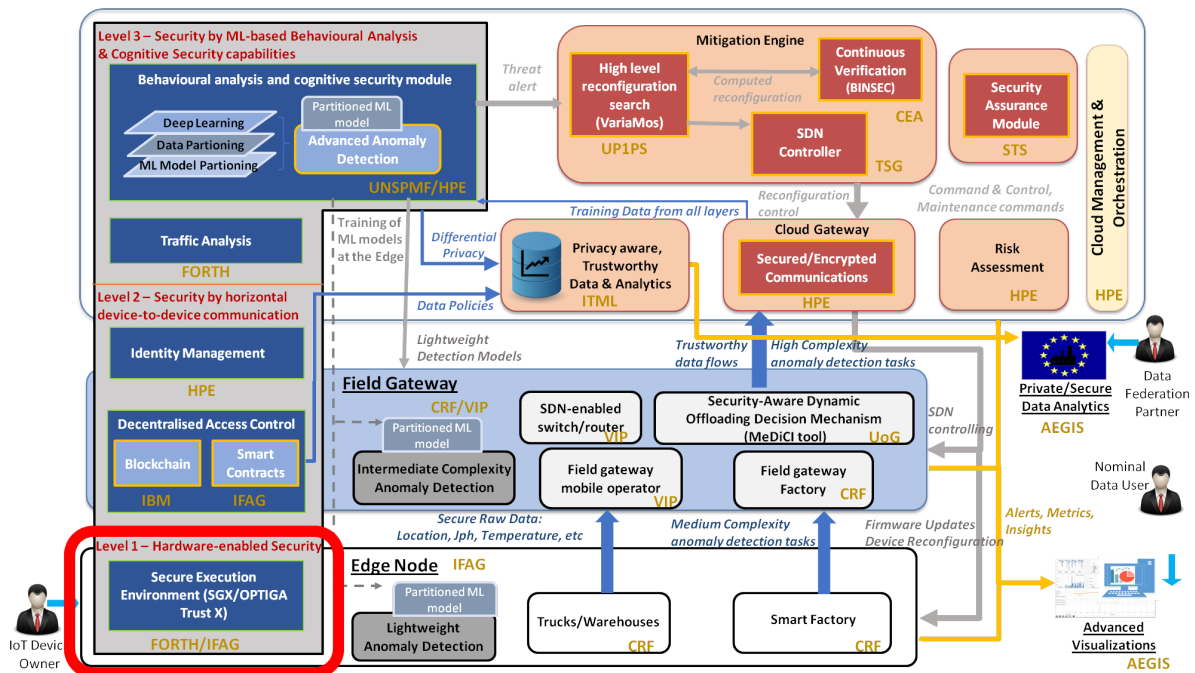


Figure 7: Level 1 – Hardware enabled security

2.7.1 Level 1 – Secure execution environment

Objectives / Brief description

- **Authenticity:** which starts with proof of the genuine entities in the dedicated computing network. Typically, Secure Element is used for device-to-device authentication. If users are involved, then Secure Element shall be used to prove users' genuine identity.
- **Physical isolation.** Comparing to the software-based TEE [39], the biggest advantage is physical separation of Secure Element so that remote attacks are immensely hardened. Via predefined interfaces like I²C, ISO14443, NFC etc. Secure Element are still able to be connected but only after schemes like mutual authentication, etc.

In general, we can consider the Secure Element as the root of the trust before any communication starts. How can Secure Element take this role? The most important principle is to assign critical security decisions to the Secure Element, e.g. deciding whether the entity

is authentic and whether the user is the right one? If these turn out to be true, then we can trust further execution from the entity and its computing environment.

Here we focus on the edge devices security, but the principle is universally applicable also for any critical infrastructure like router/gateway, server/cloud.

This module will also provide secure enclaves for the Android OS, targeting x86-based IoT Gateways and Hubs, in order to mitigate attacks that target sensitive or critical data and communication channels. The module will offer the first SGX interface for Android OS (to the best of our knowledge), as well as services that enhance its security and offer protection schemes for several applications that deal with sensitive or secret data.

Technical pre-requisites and requirements

The most common security decision is based on mutual authentication. Secure Element and any other devices can establish a challenge-response procedure. Mostly used is the symmetric cryptographic function, like AES. Both entities keep a master key securely. Using this shared secret to encryption random numbers as challenge, the signed challenge can be verified by other entity based on the same key.

After mutual authentication, the entities usually derive keys for further communication. The essential rule is never exposing master key, but to use derived keys for communication. These derived keys are usually called session keys, because they exist temporarily only in a certain communication session.

Intel SGX must be provided in the machines that will run this module.

Interactions

- In case of machine-to-machine communication, the common interface is I²C. Secure Element acts usually passive, meaning the control and processing unit from the industrial robot shall “talk” to Secure Element first and after positive response then give security relevant tasks.
- In case of human-to-machine communication, the Secure Element is usually built into a mobile token which via interfaces like BLE, USB, NFC to connect to mobile devices. Within the token depending on the complexity in terms of additional button, biometric sensor, display etc. the Secure Element shall support I²C, SPI, GPIO, ISO14443/NFC etc. Eventually the Secure Element shall act as Master for example in SPI connection.

2.8 Simulated Environment

C4IIoT will offer simulated environment to test the proposed security methods and integrate them in the overall demonstration framework in a controlled environment. To allow testing and evaluation in realistic simulation setup, C4IIoT will use real-world components integrated in a controlled and isolated simulation environment, in which different types of security attacks and breaches will be synthetically created through appropriate procedures, and relevant data will be logged and collected for further processing.

The proposed simulation framework will rely of specific hardware equipment and software elements which will be developed, used and/or defined in order to allow for simulated security threat tests and collection of relevant data. For example, novel edge IoT devices based on 4G NB-IoT technology will be fabricated (UNSPMF) that enable detailed logging of all relevant parameters related to operation of edge devices, such as sensor data, protocol data, radio-environment data, location and movement data, energy consumption data,

etc. Mobile network operator network elements and infrastructure (VIP) will be used to collect data such as protocol data or various data flows statistics at the locations of field gateway or cloud servers. Finally, security threat scenarios and data collection procedures will be defined in detail to simulate and log security threats, log all the relevant data, and subsequently apply, e.g., mitigation measures, machine learning algorithms, learn behavioural models and evaluate anomaly detection methods.

2.8.1 Fabricated devices

Objectives / Brief description

Fabricated devices are edge nodes in the architecture. Based on the current proposal, those devices will have to support the following:

- operations of functional elements (such as sensors and actuators) that enable purpose of device in CRF use cases;
- security-by-design with embedded security elements (SGX, Optiga Trust X [40], Trusted Booting and Remote attestation);
- running Hyperledger Fabric peer;
- ML at edge where first tier of anomaly detection is performed, based on lightweight statistical and machine learning techniques;
- secure wireless communications protocol;
- reasonable power consumption that allows batteries as power source (at least for mobility case);
- size and manufacturing price.

Technical pre-requisites and requirements

Requirements analysis on the C4IIoT edge nodes and the gap analysis on the CRF's existing devices, including sensors mounted on trucks and the smart factory is prerequisite for device design.

In order to fulfil requirements, appropriate hardware platform have to be considered. Due to such diverse and demanding objectives it's clear that either small industrial PC device has to be used or some of the features have to be moved to other layers or removed altogether.

The devices have to support provisioning via field gateway and it could consist of firmware, application and/or configuration changes.

Communication technology that should preferably be used is NB-IoT or LTE, both offering carrier grade SIM based security, wide coverage and usage in different countries (roaming). Using standard commercially available IoT devices might limit connectivity options to low-power Wi-Fi based solutions.

Interactions

For the purpose of exchanging data need for both CRF application purpose and security aspects, fabricated devices will be communicating primarily with field gateway or even directly with the cloud. They will authenticate using AA4CE (Authentication and Authorization for Constrained Environments) mechanism. Authentication might be needed for CRF application as well.

Additional information

It is not clear whether devices will be fabricated for both use cases. Depending on the type, frequency and amount of data being transferred appropriate communication technology should be used. Additional info is needed from CRF regarding current technical solutions for *Logistics 4.0* and *Smart Factory*,

3 Edge Node Layer

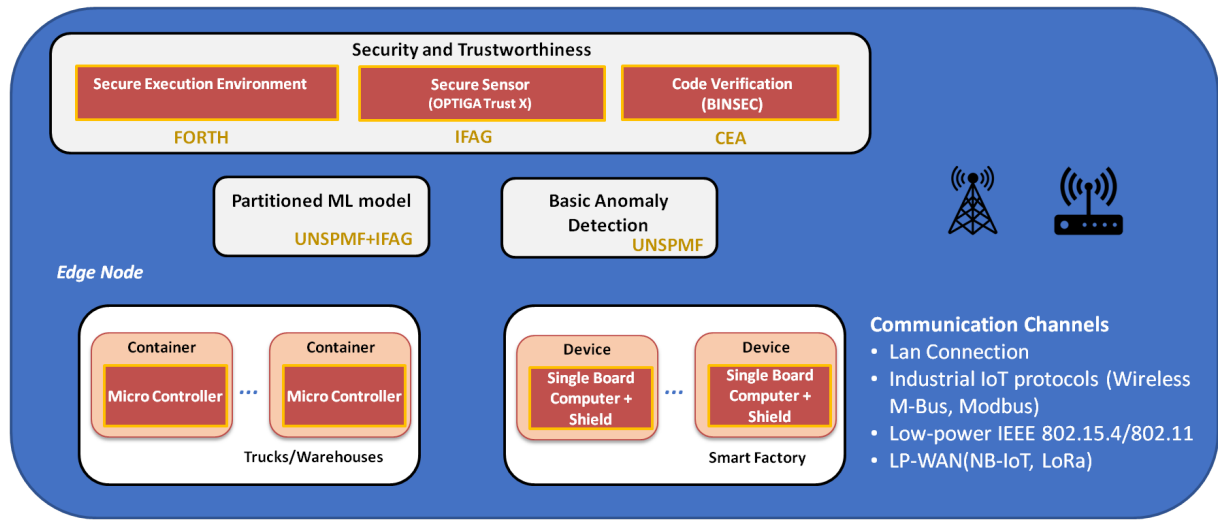


Figure 8: C4IIoT edge node layer

The edge node layer includes the devices and sensors that provide the data that feed the whole framework. Standard commercially available IIoT devices will be used based on different IIoT connectivity options, ranging from low-power Wi-Fi based [41] solutions, for the *Smart Factory* use case, to cellular-based solutions (NB-IoT, LTE-M) [42], for the *Logistics* use case.

The requirements for computational/memory/power-supply resources of the IIoT devices to (a) support emerging hardware-enabled security routines and (b) evaluate existing/emerging embedded hardware platforms to support these tasks, will be defined in the early stage of the project. Wherever needed, custom-designed IIoT devices will be engineered and fabricated to support the required applications and use cases. More precisely, existing IIoT devices used by project partners will be evaluated against the capability of supporting all the proposed hardware-enabled security features, but where required, a new design and fabrication of LPWAN [43] nodes will be done to support the project use cases. The IIoT edge devices will be designed to operate autonomously. However, the IIoT device owners will have physical access to their proprietary IIoT devices and will be able to manipulate them, whenever required, using the available interfaces.

3.1 Edge node layer architecture refinement

3.1.1 Logistics 4.0 use case

In order to balance the contrasting requirements of having a low-power MCU (Micro Controller Unit), but powerful enough to run some of the ML algorithms locally, a 32-bit ARM microcontroller was chosen for the implementation of the logistics edge node. For the communication, an NB-IoT module will be used due to its favourable characteristics, including good network coverage and low power consumption. The edge node needs to have sensors to monitor parameters of interest, such as geographical position, temperature, humidity, acceleration/vibrations, etc. As end-to-end data flow needs to be properly secured, the hardware crypto module will be used to handle the key storage, cryptographic algorithm

execution, and random number generation, in a secure and tampering-resistant way. Finally, the power supply will need to enable stable, reliable and uninterrupted functioning of the device in all circumstances, over a long period of time.

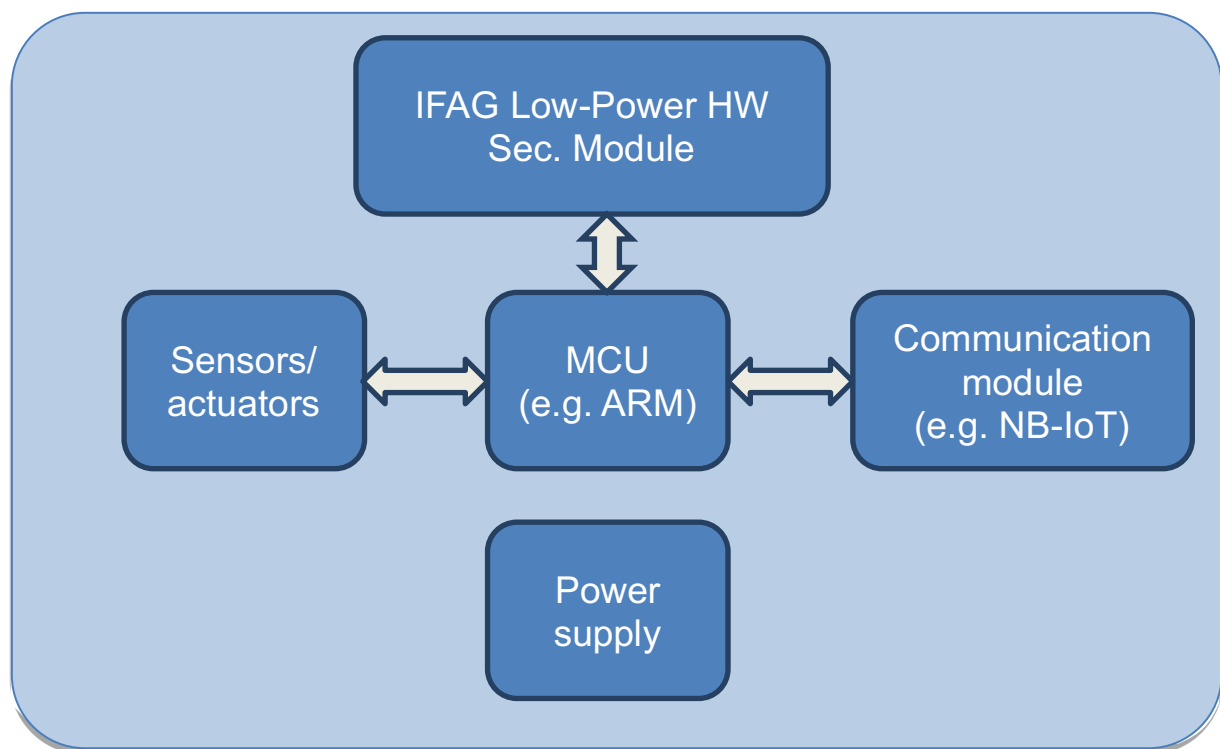


Figure 9: Logistics 4.0 use case – microcontroller residing inside a container

IFAG provides a low-power version of its secure element, providing the required functionality to establish a secure end-to-end communication, secure credential storage and to sign the transactions sent to the blockchain directly in the edge node (hardware-accelerated Elliptic Curve Digital Signature Algorithm or ECDSA). By adding the secure element, heavy crypto functionality is offloaded from the low-power MCU, which normally has strict memory and computing power constraints that make asymmetric cryptography unfeasible without additional hardware support. Furthermore, inside the secure element keys are protected against physical attacks, which might be performed while the containers are transported. Cellular communication allows to seamlessly establishing remote communication with the cloud/field gateway from the edge nodes, using the existing infrastructure from mobile operators.

3.1.2 Smart Factory use case

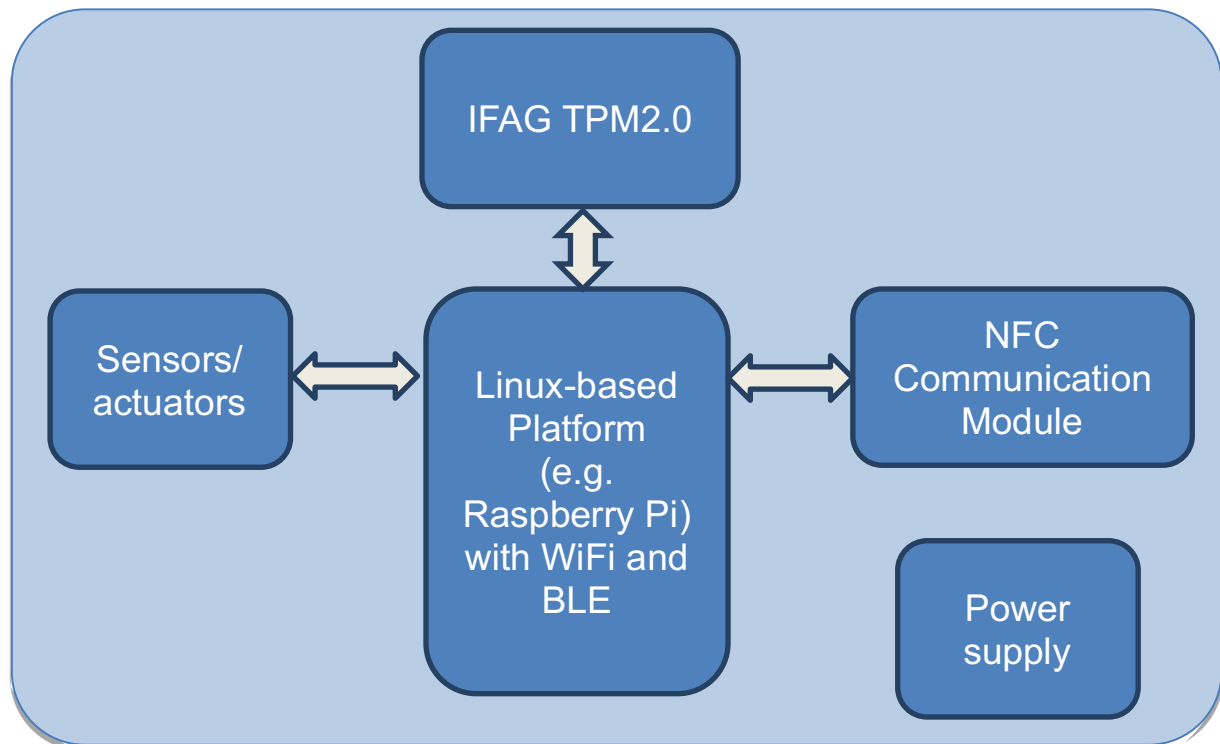


Figure 10: Smart Factory use case – edge node structure

In the *Smart Factory* use case, we use a flexible Linux-based computing platform (e.g. Raspberry Pi), since more energy is available, and a wider feature set and faster development are favoured. In order to satisfy requirements for the analysis of the binaries executed on this platform, the 32-bit version of Linux is needed. Now, we can use the fully-featured TPM2.0 [44], from IFAG, that includes authenticated and encrypted communication, secure credential storage, trusted booting, and signing of the blockchain transactions, including a wider set of secure algorithms and curves than the low-power counterpart used in the Logistic 4.0 use case. In this case, the edge node communicates with the cloud/field gateway using WiFi connectivity. In addition, a NFC interface is added for identification of operators with physical access to the node. Using a secure token (with a personal plastic card form factor), operators can perform a secure identification and access the node for configuration or sensor readout (e.g. using a tablet with BLE enabled after successful identification via NFC).

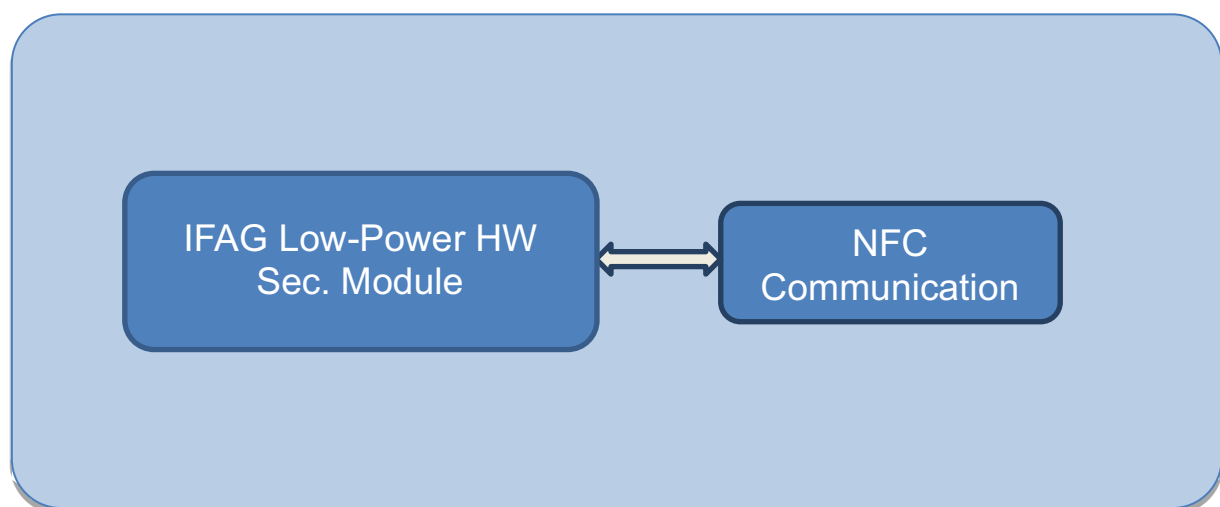


Figure 11: Smart Factory use case – passive personal card NFC token for personal identification

4 Field Gateway Layer

All the IIoT devices are connected to field gateways which form the second layer of our architecture. The field gateway might be under the control of the device owner (for example it might be the IoT gateway as part of the company's private network) or not (for example in the case of smart meter infrastructures where the concentrator is away from the homes). As an intermediate layer, the field gateway acts as an element with increased computational power to perform security-related (e.g. anomaly detection) actions with significantly lower latencies than the cloud since it resides close to the edge nodes. It will support both edge nodes that utilize (local or internet) network connectivity as well as those based on cellular protocols. In the latter case, the field gateway will reside in proximity of suitable mobile core network elements offered by mobile network operators so as to retain its proximity to the edge nodes without requiring any other adaptation of the proposed architecture. The role of the field gateway in C4IIoT is dual. Besides acting as a Software-Defined-Networking (SDN)-enabled network node, i.e., an SDN-enabled switch/router for the network communication facilities, it will also include a local offloading/outsourcing decision mechanism. This mechanism evaluates the trade-off between confidence in anomaly detection and cost of offloading to the cloud (high computational power, high latency), and automatically triggers the latter when appropriate. This allows to run actions at the edge, closer to or on the IoT devices themselves, reduce the attack surface by minimizing communication to the cloud and reduce response times and energy consumption.

4.1 Field gateway layer architecture refinement

4.1.1 Logistics 4.0 use case

As it shown at the Figure 13, the Field gateway will be located in the mobile operator premises, protected by strong security mobile operator infrastructure. The Edge node will be directly accessible from the Field gateway, in such a way that no data exchange between devices will leave trusted mobile network infrastructure. Only data sent during offloading to the cloud procedure will reach the Internet. The operation of the security-aware dynamic offloading decision mechanism is similar to the one described in the next section for the Smart Factory case.

The reason for locating Field gateway in the mobile operator premises is trusted environment for safe and secure communication between edge node and Field gateway. Security in mobile networks relies on standardized technology, security policies and encryption algorithms which provides the confidentiality of user data and prevents hacker attacks to the network and network elements. Also, by means of security, mobile network can assure data integrity protection, access control for the system, and users' authentication. Plus, being a global industry standard, mobile technology, like NB IOT, benefits from a large international ecosystem of vendors and other experts to constantly review and enhance security functions and algorithms as new threats arise.

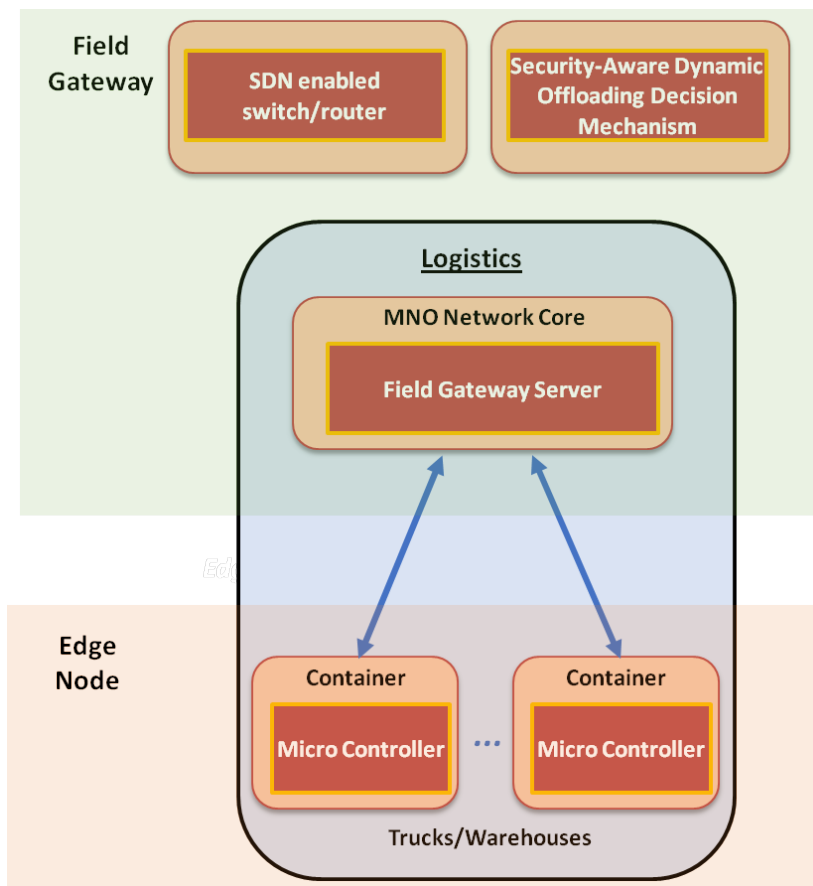


Figure 12: Field Gateway - Logistics 4.0 user case

Security in mobile networks is achieved through a set of protocols, certificates, procedures, and best practices applied to the BTS (Base Station Receiver) and other LTE network elements. Security in the LTE network can be divided into five functional areas (Figure 14):

- Air-link security (ciphering of IP packets for control and user plane)
- BTS security (Firewall, Traffic filtering, Files encryption, Software verification)
- Transport security (using IP security gateways)
- Operation and maintenance security (user authentication via local credential or centralized authentication server)
- Certificate management

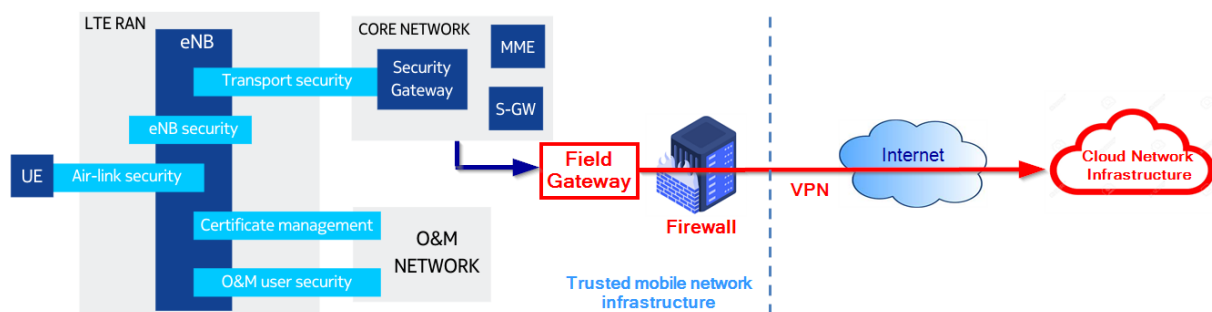


Figure 13: Mobile operator infrastructure

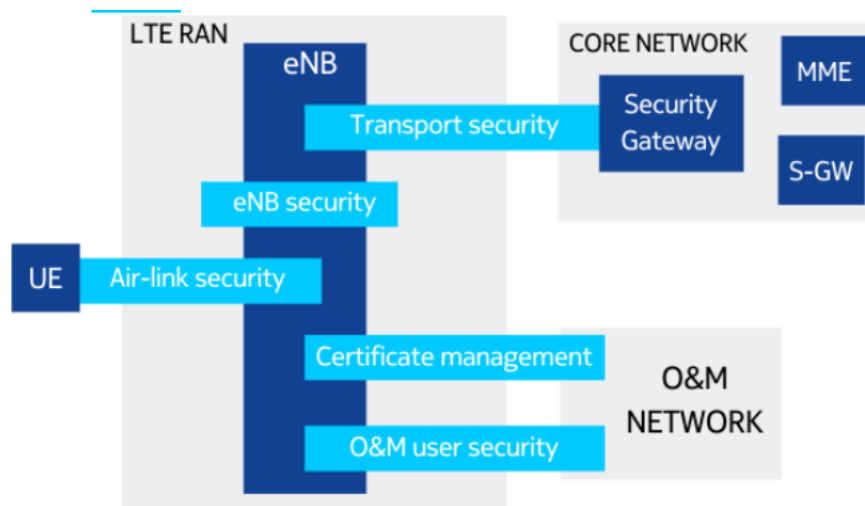


Figure 14: Security functional areas

As shown above, security concepts are very well covered in all area of mobile network even without additional HW encryptions. By moving the Field gateway into a trusted mobile operator environment we heavily reduce the chances for compromising of user data or communication at all.

Inside mobile a network infrastructure, the data travel encrypted and therefore in a secure way and under control of the mobile network operator. Generally, the problem with security of information could appear once the data leaves the mobile network and is sent over the Internet. In order to provide safe connection from the Field gateway and cloud infrastructure additional mechanism like firewall rules and VPNs (Virtual Private Networks) will be applied.

Edge node will send data directly towards the Field gateway where data will be analysed and checked for anomalies using centralized rules and black lists. Only traffic that does not contain malicious signatures or suspicious patterns will be passed towards other devices/systems. Also the Field gateway will filter traffic in order to protect vulnerable or unpatched devices, to keep them functional and safe.

The same approach will also be followed for the *Smart Factory* use case. The Field gateway will be located in the FCA premises and the edge node will send data directly to the Field gateway using WiFi communication. Each plant will have dedicated Field gateways. Additional security mechanism (VPNs and Firewall rules) will be applied in order to establish secure connection with the cloud infrastructure and prevent a disclosure of private information. The VPN security model provides confidentiality, authentication and message integrity. A VPN maintains the same security and management policies as a private network. It provides confidentiality, authentication and message integrity. They are the most cost-effective method of establishing a virtual point-to-point connection between remote users (cloud infrastructure) and an enterprise customer's network (field gateway).

Minimum HW requirements for Field gateway in both cases should be:

- CPU: Intel CPU with 4 Core, 3 GHz
- RAM: 4 GB
- Operating system: Ubuntu 16.04.6 LTS Xenial Xerus or any other Linux distribution

4.1.2 Smart Factory use case

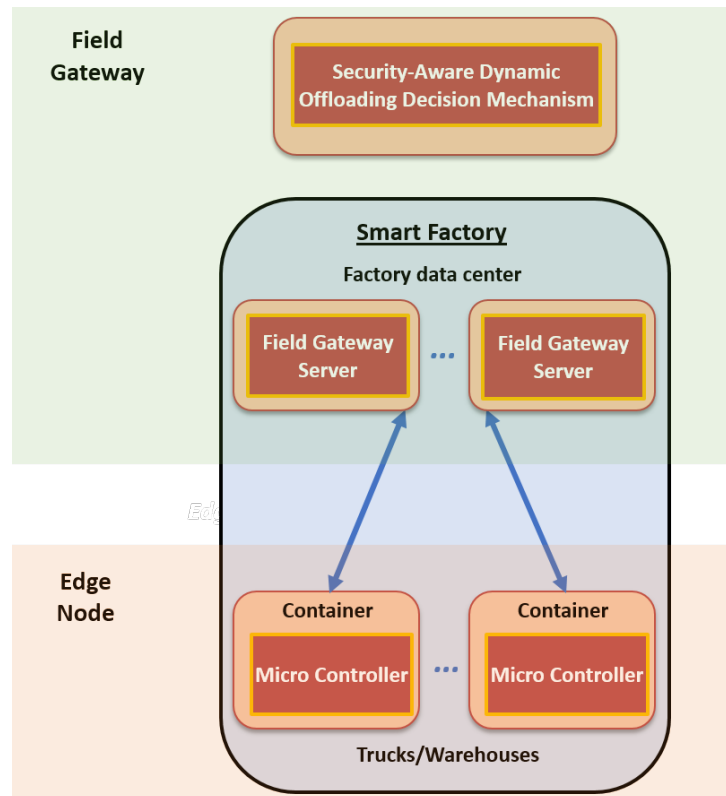


Figure 15: Field Gateway - Smart Factory use case

The security-aware dynamic offloading decision mechanism will be situated at the field gateway layer. Edge node devices will submit an offloading decision request to it when the confidence from their own lightweight anomaly detection component falls below an acceptable level. These requests will contain the necessary task information (e.g. size) to be incorporated into the decision-making processes to determine if the task should be offloaded to the intermediate or advanced anomaly detection components located at the field gateway and cloud layers respectively.

More specifically, the security-aware dynamic offloading decision mechanism evaluates the trade-off between the confidence of the lightweight anomaly detection from the edge device against the network, processing and other delays caused by offloading the task to cloud based on set of heterogenous criteria. Once a decision has been made, the outcome will be reported back to the edge node to offload it to the appropriate location. Once a task has been executed, the actual execution time, network delays and other relevant data are stored at the field gateway to inform future offloading decisions.

5 Cloud Layer

The Cloud layer is the main data-handling component of the architecture. In addition to consuming data and providing the appropriate services to users, the cloud layer components have an essential role for C4IIoT. The Cloud layer is part of the Cybersecurity Framework defined in CSA and adopted for this project.: i.e. components communicate with the field gateways through a cloud gateway, responsible for the communication functionalities. The cloud layer hosts several services, including support of secure data streams, as well as services that ensure the authenticity and integrity of the data flows, the C&C and the maintenance. Inside the Cloud layer, C4IIoT define the components that enable the security-as-a-service aspect of C4IIoT, through the behavioural analysis module, the cognitive security component, the PKI-based authentication based, and the Risk Assessment repository. In the design of Cloud layer, the security is pervasive and includes the main security aspects, following the practices defined in the previous chapters. The former module generates behavioural context-aware models through machine learning techniques (including structured non-convex optimization models, deep neural network models, etc.) based on the data streamed from the cloud gateway. The latter receives models from the Behavioural analysis and cognitive security module and manages the required mitigation actions through C&C and maintenance signaling.

5.1 Cloud layer architecture refinement

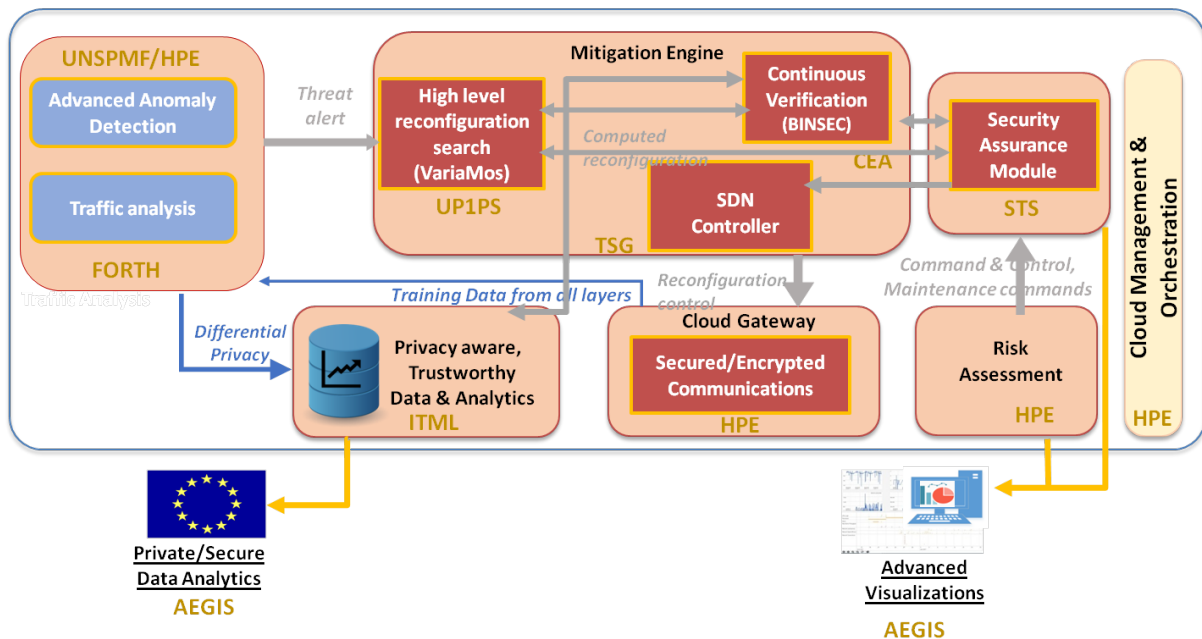


Figure 16: Cloud layer

Cloud Layer hosts software modules and centralized data systems with aim to control and manage objects, communication and security.

For the deployment, the general idea is to consider each module as an entity exposing JSON/REST services (possibly organized as fine granularity, like microservices), and packaged as Docker container to allow/facilitate standardization, increase portability and make the implementation independent from the hardware platform below.

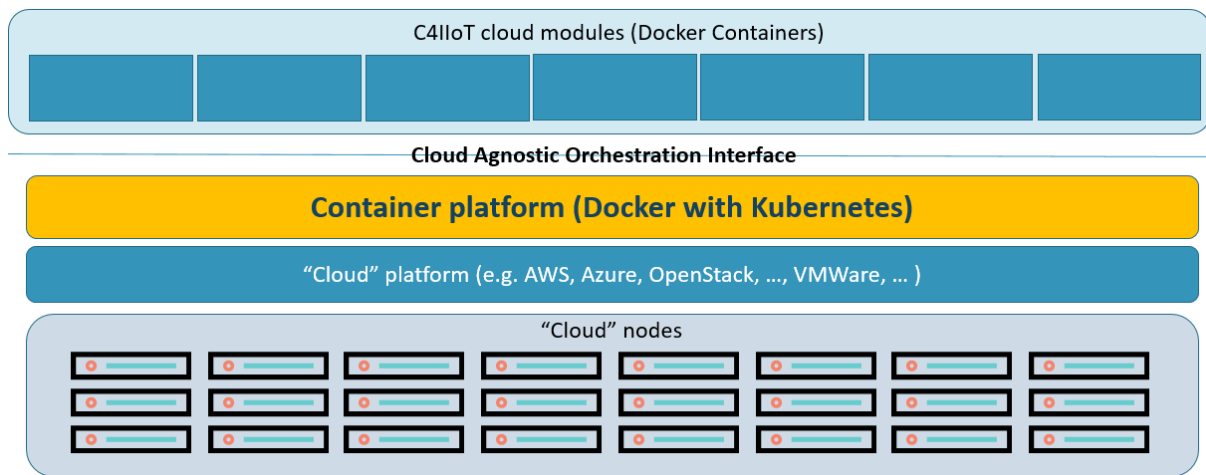


Figure 17: Containerization

In this way, the reference architecture will stay “cloud agnostic”, i.e. able to work on any types of cloud infrastructure (public, private, hybrid using different technologies) as well as virtualized and also traditional Data Centers. The implementation of framework at the trial side will use the Cloud already used by CRF (e.g. AWS).

The use of containerization provides a high degree of portability for C4IIoT components to any container platform. It also simplifies the repeatable development, build, test of modules on Integration Environment (WP4) and their transfer to Demonstrator Environment (WP5).

Base principles for this approach are the following:

- each C4IIoT technology partner provides a module builds (using T4.3 devops) one or more Docker Container(s) containing the SW
- for each module the partner developer specifies:
 - the required resources for the container
 - the dependencies to other containers
- all container images are store inside a Trusted Docker Registry:
 - all images are “signed” by the provider at PUSH time to the Registry
 - deployment will check the “signatures” before starting the container

In this layer, in addition to modules hosted on Cloud, two modules assume a specific job: the Cloud Management & Orchestration and the Cloud Gateway.

The Cloud Management and Orchestration will manage the cloud infrastructure and orchestrate the deployment of container tasks in automatic way whenever possible and convenient,

The module for our implementation on the demonstrator will use open-source software (e.g. AWS Amazon Elastic Container Service for Kubernetes-EKS, Google Kubernetes, ...).

During project execution, the possibility and convenience to extend containerization also to Level 1 and 2 modules will be considered and its potential benefits will be assessed. If it is adopted, , the Cloud Management and Orchestration Module will replicate the same container-based approach at these layers. For these special cases, developer will also specify target deploy for container (Field, or edge).

The Cloud Gateway is the module that, on the one hand receives all requests from the Field Gateways and address them to target modules living on Cloud Layer, and, on the other hand, issues requests, on behalf of Cloud Layer modules, to Field Gateway and edge nodes (where applicable, e.g., for controlling and updating commands). This module will include API Gateway functionalities that convert client calls (e.g. HTTPS requests) to microservice calls in an authenticated and controlled/monitored way. At this early stage of the project, we envisioned to include two firewalls in the Field: one for South-bound communication with the field gateways, and the other for North-bound communication with - the Private/Secure Data Analytics module, the Advanced Visualization module and other company systems external to C4IIoT

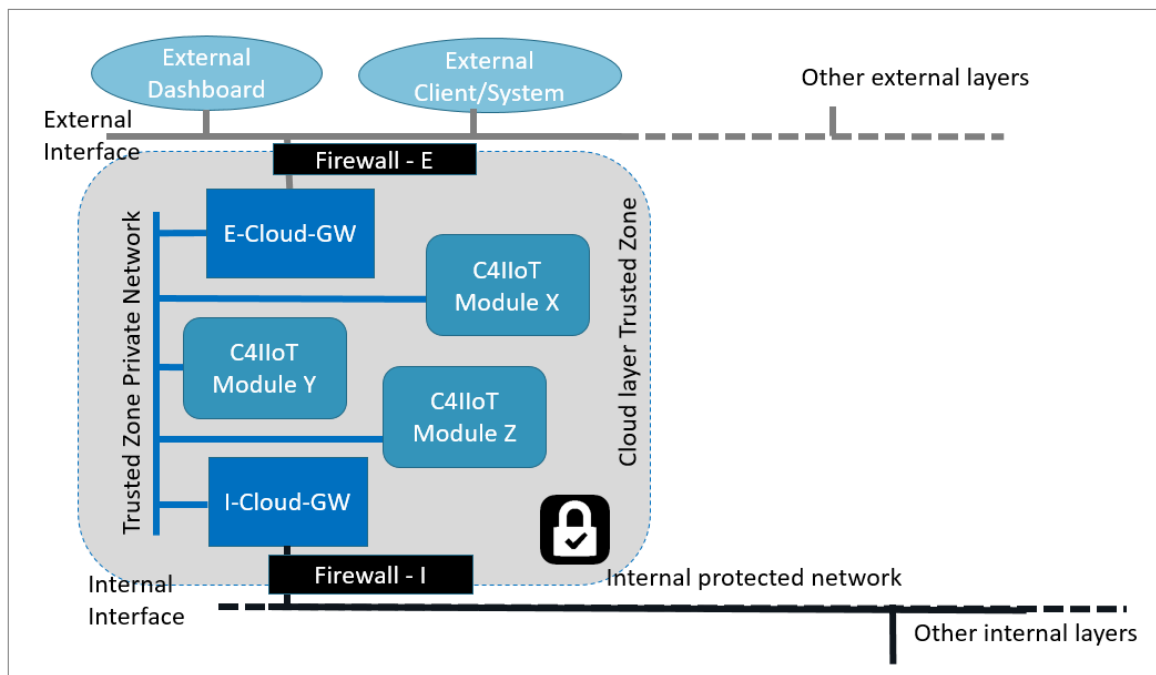


Figure 18: Securing C4IIoT access to Cloud Layer

6 User interface

The Advance Visualization Toolkit (AVT) module is the user interface of the C4IIoT framework. It provides the means to visualise several indicators deriving from the analysis of Big Data. It enables the final user to explore data in a high level through several interconnected, interactive visualisations that also allow drilling into more detailed information to reveal hidden relationships and insights. To achieve this goal, the toolkit follows an internal architecture that comprises of 3 layers, namely: data, business and presentation. This architecture provides the flexibility to adapt to various domains and data sources.

The Data layer includes the original data and the mechanisms that enable their retrieval by the business layer of the AVT. The data layer includes a number of indicators that are domain specific and can be calculated from the original data.

The business layer of Advance visualization module handles the processing of the received Indicator data in a format suitable for the visualisations. A cache mechanism is also available to enhance performance. Finally, this layer serves the REST services required by the visualisations to offer their functionalities.

The presentation layer includes a timeline analysis component and multiple visualisations. The timeline analysis component offers the functionalities for temporal analysis of the Indicator values. Through the timeline control, the end user may move back in time, narrow the viewed time window, and compare two different time points, in order to get insights of the data. The timeline control drives the displayed data in all the other visualisations of the currently opened view inside the AVT. These visualisations include a set of interactive graphs and charts that form the heart of the AVT. Bar charts, line charts and pie charts are some of the standard forms of data representations. Different visualizations are used to display different types of data, in order to make the understanding of data easier. Changes of the time window propagate to the rest of the visualisations, so that they always follow the selected time window.

This way, end users can get situational awareness of the system (where original data come from) at any given time. They can check to see if any abnormal situation seems to be about to happen (or already took place) and generally search for patterns in events and Indicator values that can reveal correlations and help them decide on appropriate actions.

6.1 FVT integration regarding the architecture refinement

The visualization framework will be built based on the input received by other modules of the C4IIoT framework. This communication is foreseen to take place via a service-oriented approach where each component exposes its outputs via web services that can be afterwards consumed by the visualization component. Detected anomalies, security alerts and extracted analytics will be the main information to visualise at different levels of granularity that can reach the log files where the initial event was caught in.

The selected architectural style is REST and during the development period we will continuously structure the service signatures that are going to be available so as to support the visualisation needs of the various components. It must be noted that since the project is still at an early stage, services are subject to define and additions of services are expected as development advances and user needs evolve.

The Risk Assessment will have an interface to the Visualization module regarding the risk findings, the evaluation, the assessments and other related information. Details are subject to discussion.

7 Integration of Security Aspects

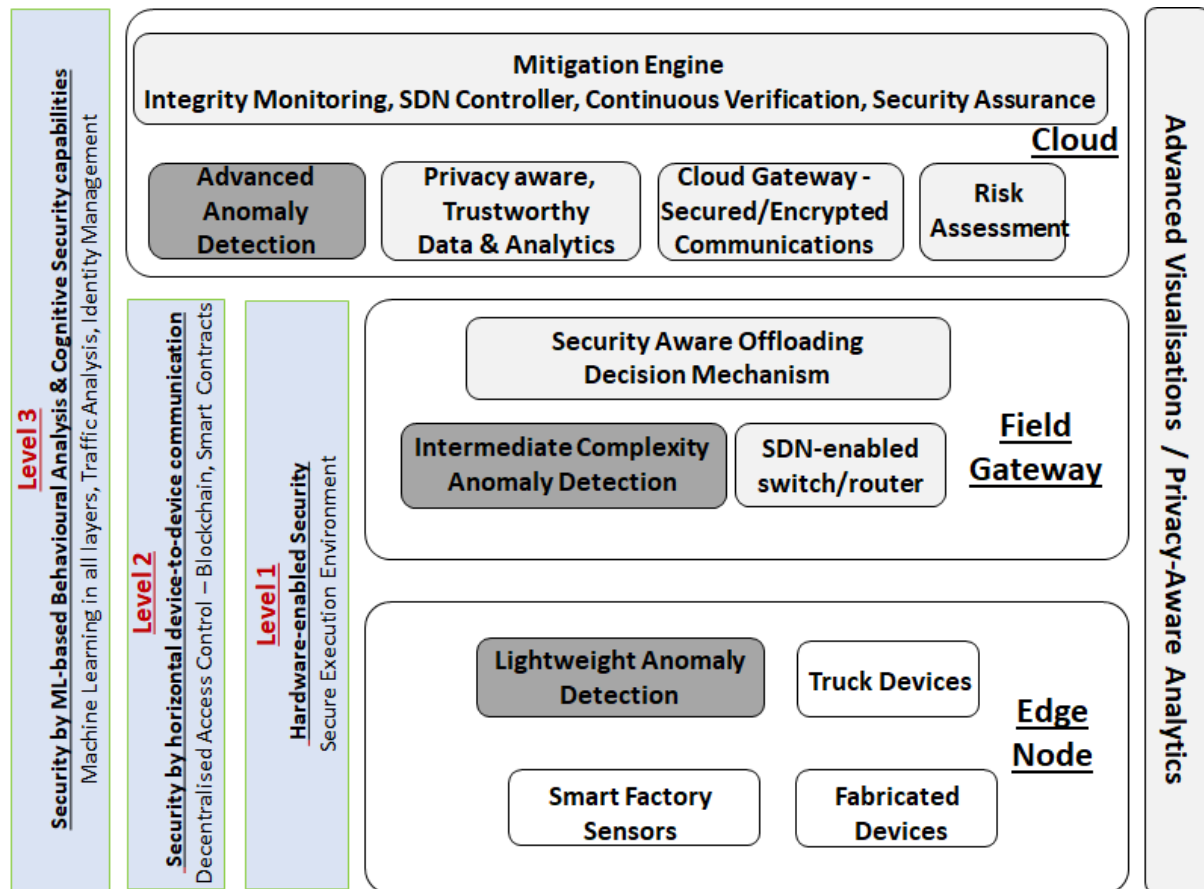


Figure 19: C4IIoT Cybersecurity 4.0 framework

7.1 Level 1 – Hardware-enabled Security

The foundation of our security strategy leverages the capabilities of secure elements that lie in the hardware chip, and support, at least, advanced cryptographic functions and physically-protected storage of private and secret keys. A code integrity mechanism verifies the integrity of the code running within the trusted hardware enclaves and the kernel code as well. The code in the trusted enclave is responsible for ensuring the integrity of the kernel code at runtime, and detecting any modifications by malicious software, once the system is running.

The hardware-enabled components will further offer security to other layers of the architecture (i.e., edge nodes, field gateways, cloud), including the facilitation of strong authentication and authorization schemes that are based on single security tokens. These security tokens, stored in hardware-enforced secure enclaves will be responsible for identity and access management, as well as for ensuring accountability; as a result, they will prevent false identities to commit fraud, acquire sensitive data, commit data theft, or manipulate the system by any means. The described functionalities constitute overall a secure execution environment that is built upon several partners' technologies, including FORTH's privacy-enhanced execution, IFAG's OPTIGA Trust X device for authentication and encryption, IFAG's OPTIGA TPM for secure booting and remote attestation.

7.2 Level 2 – Security by Horizontal Device-to-Device Communication

Within the C4IIoT Cybersecurity 4.0 framework, personal data aggregated by the IIoT devices (edge nodes) may not be shared among different devices or with data federation partners in raw form. Furthermore, a data federation partner may want to protect its data and analytics results from its competitors, or to monetize its data selectively. The blockchain technology can enhance the C4IIoT Cybersecurity 4.0 framework in terms of privacy, auditability and reliability. However, decentralized solutions build on blockchain technologies do not scale yet to the amount of data aggregated by IIoT devices. Moreover, in case a privacy policy or a business decision changes, the system needs to be able to easily revoke access to data. Henceforth, the IIoT aggregated data and analytics results will be stored on a restricted secure storage on the cloud and only the link to the restricted storage will be recorded by the decentralized access management solution. When a privacy policy or a business decision has changed, the decentralized access management solution will revoke the access to the restricted storage to some data federation partners as required. In addition to enforcement of policies, the records documented on blockchain can be also used to prove compliance and for forensics purposes. Level-2 security of the C4IIoT framework will be built upon multiple technologies brought by the partners, including IFAG's smart contracts management module and IBM's decentralized consent management

Since Decentralized solutions built on Blockchain technologies do not scale yet to the amount of data aggregated by IIoT devices and in order to support the “right to be forgotten”, we will develop a hybrid solution. In this solution, in IIoT aggregated data and analytics results will be stored on a restricted secure storage on the cloud and only the link to the restricted storage will be recorded by the decentralized access management solution. When a privacy policy or a business decision is changed, the decentralized access management solution will revoke the access to the restricted storage to data federation partners as required. In addition to enforcement of policies, this solution will leverage the records documented on Blockchain to provide forensics and to prove compliance for audit purposes. IFAG's smart contracts management will support this mechanism via utilising embedded security of edge nodes which will allow them to interact with the blockchain while protecting their proprietary data.

The identity management solution will be able to provide a smart identity to all agents, devices, citizens who interact in the IIoT, while at the same time being able to manage high volumes of internal digital certificates needs and the different types of user credentials and smartcards. In addition, it will integrate all internal PKIs into a single PKI to optimize operational costs, provide support to any device and certificate need, and contextual and risk-based authentication. Finally, it will offer enhanced privacy of Identity for GDPR compliance support.

7.2.1 Blockchain

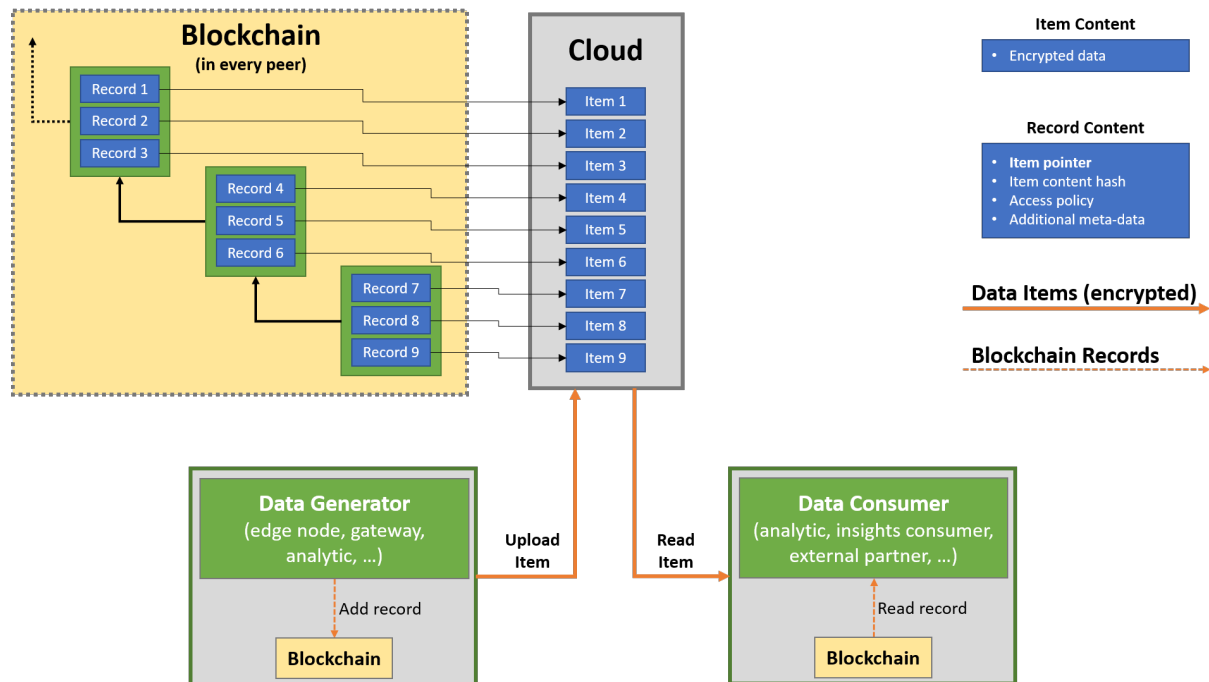


Figure 20: Blockchain network

The Figure 20 demonstrates the C4IIoT's hybrid approach of using a Blockchain network and Cloud Storage to distribute and store data items.

A data generator (e.g. Edge Node) will upload new data item to the cloud storage service, where encrypted data is being stored. The data generator will then add new record to the Blockchain channel, containing pointer (URL) to the data item on the cloud. A hash of the pointed item's content will also be recorded on the Blockchain, to allow verifying the integrity of the data stored on the cloud.

Data consumers will use the Blockchain channel to be notified of newly-uploaded data items. Data consumers will read the URLs stored on the Blockchain records, then access the cloud and download the data items required for analysis.

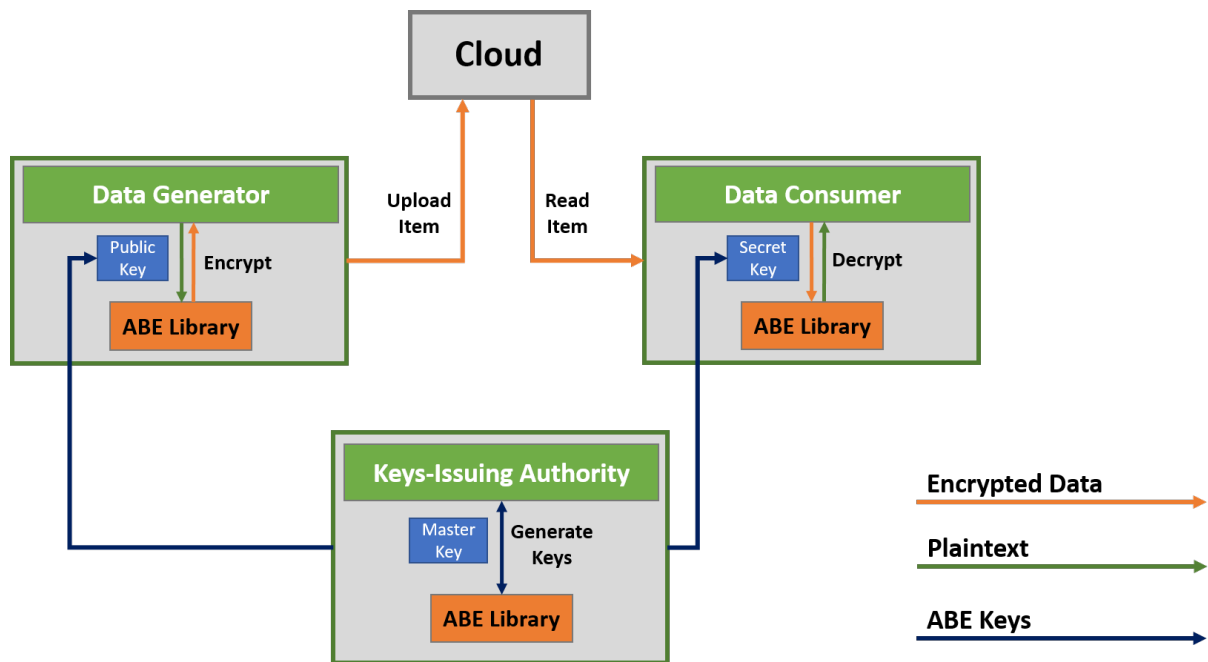


Figure 21: Attribute-based Encryption (ABE)

Figure 21 demonstrates the usage scheme of Attribute-Based Encryption (ABE) within C4IIoT's decentralized access control solution.

Each data consumer is associated with a set of attributes, and is provided with a personal secret key corresponding to its attributes.

Data generators encrypt their data before uploading it to the cloud, using a public key shared by all. As part of the encryption process, data generators specify an access policy to each data item, describing who shall be able to decrypt it.

Data consumers, after reading a data item from the cloud, use their personal secret key to decrypt the data item, given that the access policy of the item allows them to do so.

A keys-issuing authority generates an ABE public key to be used by all data generators for encrypting data, and a personal ABE secret key for each data consumer to be used for decrypting data.

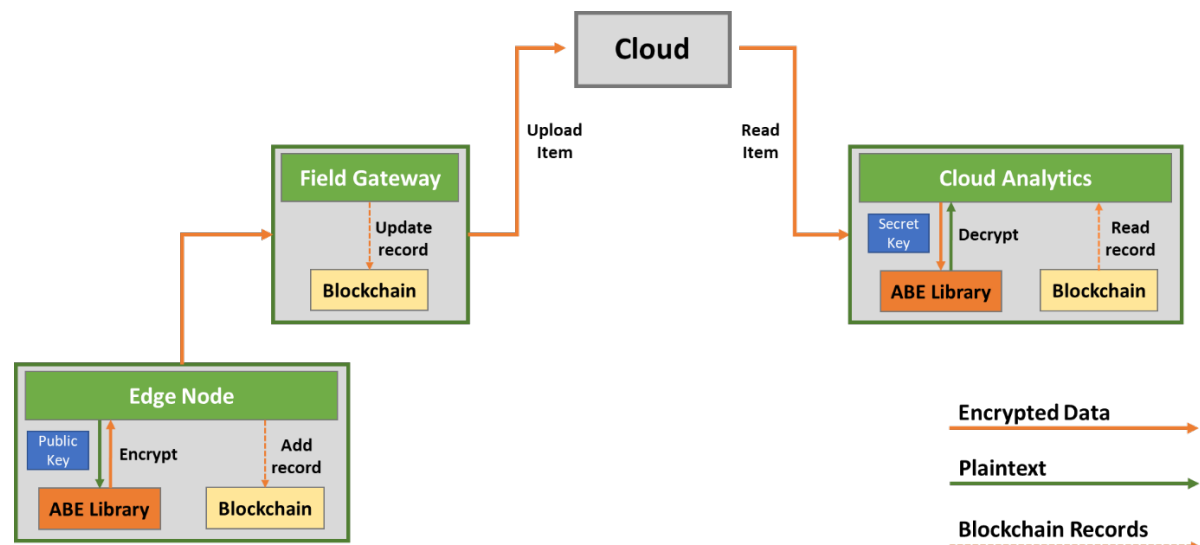


Figure 22: Data-flow

Figure 22 presents an explicit end-to-end example of the data-flow in the context of the decentralized access management solution.

In the example, an Edge node generates a data item and encrypts it using a public key and the ABE library. An access policy is specified by the edge node as part of the encryption process. The edge node then adds a new record to the Blockchain channel with hash of the data item's content, to allow verifying the integrity of the data stored on the cloud.

The Edge node then sends the encrypted data item to a field gateway, that uploads the encrypted data item to the cloud storage service. The URL where the data item is stored is obtained by the field gateway, that adds it to the existing record on the Blockchain corresponding to the relevant data item.

Cloud analytics are able to query the Blockchain channel and made aware of a newly-uploaded data item. The record corresponding to that data item is read and the URL included in it is used to approach the cloud and download the encrypted data item. Using the ABE library and a personal secret key issued by the ABE keys-issuing authority, cloud analytics are then able to decrypt the content of the data item and analyse it.

In the inbound *Logistics 4.0* use case, the plan is to deploy a Blockchain client and an ABE client in the field gateways layer.

7.3 Level 3 – Security by ML-based Behavioural Analysis & Cognitive Security capabilities

This security level reasons that, no matter how strong the hardware (level-1) protection is for the device, the code, or the communications, it remains possible to bypass or exploit it. Besides that, device-to-device communication (level-2 security) may have a localized effect (e.g., within an IIoT device geographical vicinity), and may not cover an IIoT system in its entirety. For these reasons, a separate mitigation mechanism is required, that will perform behavioural analytics. The motivation behind this is to provide a deeper understanding of the whole environment, including detection of advanced threats and anomalies.

Anomaly detection is the first step in raising alarms and devising plans for compensating and responding to the anomaly. By considering all the devices, as well as their external interactions (and not each device in isolation), we will gain powerful contextual information and form an anomaly detection model for the complete IoT ecosystem. This will enable to infer information from different inputs, and train models based on the normal behaviour of the framework. This can correspond to a simple task, e.g., determining abnormal values (e.g., an excessive value for a home thermostat), but also a more complex inference requirement, e.g., taking the input from multiple sensors (for instance, in order to validate a change in the rotational speed of a motor). The actual implementation of level-3 security will be based on parallel and distributed instances of ML algorithms implemented across all layers (edge nodes, fields gateways, cloud); these algorithms, based on the information provided by the offloading mechanism, partition the data and the ML model in question across the layers to trade-off complexity of the current task with time criticality of the reaction times. Among several predictive modelling, anomaly detection, and diagnosis mechanisms, a central mechanism will be based on analysing encrypted network flows. The motivation for focusing on encrypted network flows is that typical deep packet inspection (DPI) applications, based explicitly on clear-text packet payloads (such as firewalls, anti-viruses, intrusion detection and prevention systems) will become obsolete in the near future. Instead, the C4IIoT Cybersecurity 4.0 framework will focus on encrypted traffic analysis mechanisms based on metadata extraction, thus alleviating the problem of high encryption costs. In particular, the

traffic analysis will utilize deep machine learning to investigate the feasibility of identifying specific patterns inside network flows. Some of these patterns are likely meta-data that can be found inside packet headers or times, including packet lengths, timestamps, directions and inter-arrival times. The outcome of this step will provide insight on the proper metadata handling and processing in order to produce network signatures that will be used in order to enrich current network inspection systems' functionality. In the training stages ground-truth datasets of malicious traffic need to be collected and analysed to derive signatures based on traffic metadata. The resulting signatures will be then integrated into a network inspection system to report possible attacks or traffic anomalies inside real network traffic.

Level-3 security of the C4IIoT framework will be built upon several partners' technologies, including the HPE's Aruba Introspect user and entity behavioral analytics, Security-aware offloading mechanism via the UOG MEDICI tool, as well as UNSPMF's machine learning algorithms with ML models partitioned across the edge node, field gateways, and cloud layer of the C4IIoT architecture.

The tool will take into consideration a heterogeneous set of multiple criteria obtained by the output of the reasoning mechanism running Deep Learning models at the edge, real-time network monitoring and measurement of the energy consumption of the IoT device. It will then evaluate the trade-off between confidence in anomaly detection and the cost of offloading to the cloud (network latency, computation/response time and energy consumption) and will automatically trigger it when appropriate.

UNSPMF's models will be based on work on advanced deep learning techniques to provide more powerful contextual information and form an advanced anomaly detection model for the complete IoT ecosystem considering each device not in isolation but including their interactions too. The modelling of contextual data in the cloud, and the corresponding behavioural models, will allow the aggregation of heterogeneous data into a contextual situation and the derivation of complex adaptation processes.

7.4 Privacy-aware analytics and accountable data processing

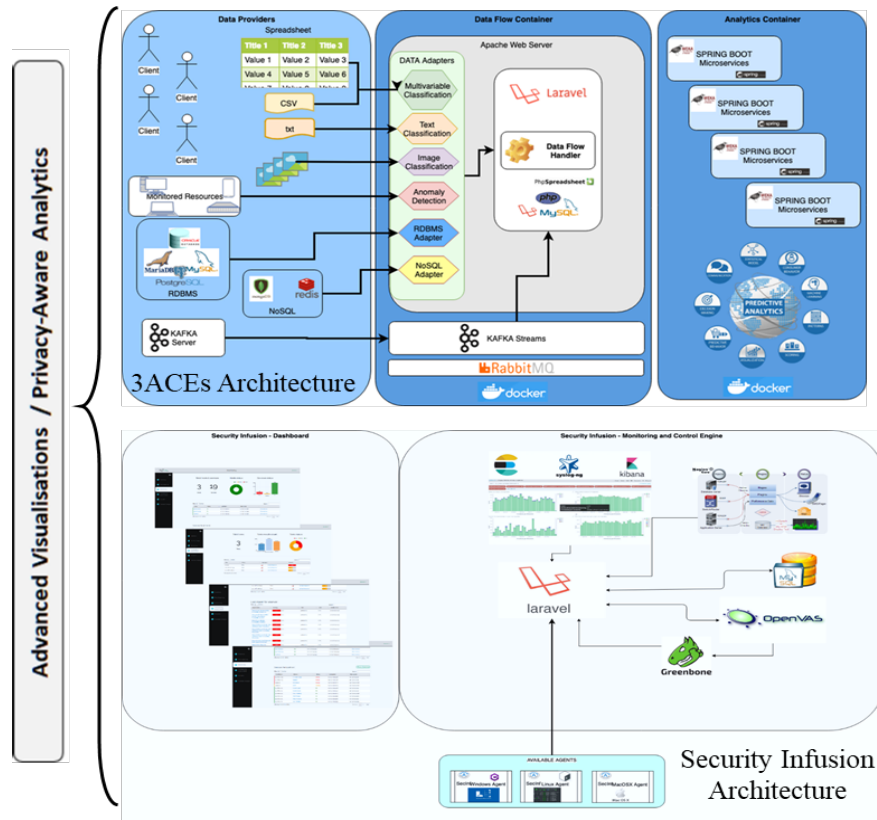


Figure 23: 3ACEs & Security Infusion Architecture

Privacy-aware analytics C4IIoT's functionality will be built upon multiple partners' technologies, including ITML's privacy-aware trustworthy analytics, as well as UNSPMF's differentially private ML algorithms. More specifically, the functionality will offer privacy-preserving storage and analytics to end-users and enterprises by allowing seamless integration and injection of heterogeneous data, and facilitate the adoption of collaborative analytics to enterprises, without exposing private or sensitive information. To achieve this, it will allow the user to select which data (or portion of the data) will be used, via a policy language that will protect data and selectively allow access according to user defined criteria.

Towards this direction, C4IIoT will exploit a number of existing technologies and tools (e.g. 3ACEs and Security Infusion, brought in by ITML) to be integrated in the C4IIoT security framework. The envisioned integrated system will allow on-demand analytics acquisition for whole datasets using Machine Learning algorithms in batches or real-time; it will provide encapsulated functionality for analyzing heterogeneous data, from any technical source, providing insights with short or no training period allowing for an iterative approach that constantly improves results, utilizing all available data sets, and applying a plethora of algorithms for providing data analytics. The industry-ready installation of 3ACEs on ITML's cloud infrastructure can accommodate any data feed through an open, extensible and scalable adapter factory that hosts any kind of adapter, from traditional relational databases, No-SQL databases or even Kafka™ streams [45]. The platform consists of three functional high-level blocks, namely the data providers, the data flow container and the analytics container (Figure 23); all modules comprising both the Data Flow Container and the Analytics Container are built in a Docker™ container allowing for easy installation on any available cloud infrastructure.

Regarding the privacy preservation C4IIoT's functionality secure hardware will be leveraged in order to ensure the compliance of the requested data access criteria. To ease the deployment, the project will offer trusted built-in functions. Privacy aware machine learning and decentralised access management will be key elements to achieve the goal of the framework. Privacy preservation, threat's identification and vulnerability assessment will be achieved with the performance of machine learning algorithms in question. Important classes of algorithms where differential privacy has been successfully incorporated are (deep) neural networks and decision trees.

Exploiting, among others, the Security Infusion tool that performs advanced predictive ML techniques at the edge, C4IIoT will ensure continuous, real time monitoring of valuable assets/resources achieving cybersecurity threats' identification and mitigation. Hence, C4IIoT security framework will have the potential to identify security threats, minimize false positive alarms and finally suggest recommendation actions to limit potential risks. The C4IIoT privacy-aware framework will be installed on the edge, capturing and handling all the data produced within the network boundaries, or select to capture data using the same adapters, pass through the data reduction process and send the minified data over a secure tunnel at the cloud service. Services at the edge will be responsible to collect the minified data from the deployed agents at the monitored devices and provide live feedback to the event controller. This controller will provide the necessary information to the Event Analysis dashboard that will provide a real-time and a timeline analysis through pre-configured views as well as feedback from a threat-response module that is responsible to gather info about potential threats and employ human feedback from IT security experts, or evolve ML algorithms towards suggested actions.

7.5 Secure communication protocols

All communications between the IIoT devices and the cloud will operate in a two-way manner and will be end-to-end encrypted using SSL. Upstream communications correspond to data sensed by the devices and will be transmitted via field gateways to the cloud. Downstream communications correspond to feedback provided by the cloud to the devices and field gateways. This feedback can take two forms, C&C and Maintenance. C&C communications might be subject to stricter availability requirements than maintenance, might occur more frequently, but are also smaller in volume. Maintenance communications are expected to be less frequent but impose significant volume on the network. Both C&C and Maintenance communications will be encrypted using SSL.

Finally, the datasets of the project will be stored in CRF's premises due to their sensitive nature.

8 Concluding Remarks and Next Steps

This deliverable provides the specifications of the C4IIoT platform architecture, based on a thorough overview of state-of-the-art methods, functional and non-functional requirements, as well as reference architectures and best practices. It draws on research and conclusions from deliverable 1.2 - positioning of C4IIoT, deliverable 2.1 - edge-node assets analysis, as well as deliverable 7.2 - data management plan.

Modules of the C4IIoT architecture are described in detail, as well as their integration and interaction through well-defined interfaces, with an indication of the efforts that must be made to implement, integrate and demonstrate such architecture through the two use cases that have been defined. The next steps will include a more elaborate discussion on the variations in the required architecture for each C4IIoT use cases.

The specifications given in this deliverable will serve as input for further work towards the implementation of the C4IIoT platform modules and their integration. The first part of the project will focus on the development and internal testing of system components, while the second part will focus on integration activities, testing and demonstration of the platform as a whole.

The implementation of the system will continue through an iterative-incremental process to produce three releases of the C4IIoT solution - a proof of concept (Minimum Viable Product (MVP)) in M12, the first complete prototype in M18 and the second prototype in M30 of the project.

C4IIoT architecture will be continuously updated following the technical and business achievements of the project and growth in consortium's knowledge of the system modules and their interactions through the process of implementation and integration.

9 References

- [1] S. Lin, B. Murphy, E. Clauer, U. Loewen, R. Neubert, G. Bachmann, M. Pai and M. Hankel, "Architecture Alignment and Interoperability—An Industrial Internet Consortium and Plattform Industrie 4.0 Joint Whitepaper," 2017.
- [2] "System and software engineering - System and software Quality Requirements System and software quality models," BS ISO/IEC 25010:2011, 2011.
- [3] "Raspberry Pi," [Online]. Available: <https://www.raspberrypi.org>.
- [4] "BINSEC tool," [Online]. Available: <https://binsec.github.io/>.
- [5] "SWI-Prolog," [Online]. Available: <https://www.swi-prolog.org/>.
- [6] "Logtalk," [Online]. Available: <https://www.swi-prolog.org/pack/list?p=logtalk>.
- [7] "Vue.js," [Online]. Available: <https://vuejs.org/>.
- [8] "Axios.js," [Online]. Available: <https://github.com/axios/axios>.
- [9] "Amazon Web Services," [Online]. Available: <http://aws.amazon.com>.
- [10] "Microsoft Azure Cloud," [Online]. Available: <http://azure.microsoft.com>.
- [11] "OpenStack," [Online]. Available: <http://www.openstack.org>.
- [12] "Docker Swarm," [Online]. Available: <http://github.com/docker/swarm>.
- [13] "Kubernetes," [Online]. Available: <http://kubernetes.io>.
- [14] "Apache Web Server," [Online]. Available: <https://httpd.apache.org/>.
- [15] "Laravel," [Online]. Available: <https://laravel.com/>.
- [16] "Docker," [Online]. Available: <http://www.docker.com>.
- [17] "Kibana," [Online]. Available: <https://www.elastic.co/products/kibana>.
- [18] "Nagios," [Online]. Available: <https://www.nagios.org/>.
- [19] "Elastic," [Online]. Available: <https://www.elastic.co/>.
- [20] "OpenVAS," [Online]. Available: <https://hackertarget.com/openvas-scan/>.
- [21] "Greenbone," [Online]. Available: <https://www.newnettechnologies.com/greenbone.html>.
- [22] "Advanced Visualization Toolkit (AVT)," [Online]. Available: <http://aegisresearch.eu/solutions-2/advanced-visualization-toolkit/>.
- [23] "DISCO," [Online]. Available: anr-disco.ens-lyon.fr.
- [24] "NETCONF," [Online]. Available: <https://en.wikipedia.org/wiki/NETCONF>.

- [25] "Intel SGX," [Online]. Available: <https://software.intel.com/en-us/sgx>.
- [26] "Aho-Corasick," [Online]. Available: https://en.wikipedia.org/wiki/Aho%E2%80%93Corasick_algorithm.
- [27] "OpenCL," [Online]. Available: <https://www.khronos.org/opencv/>.
- [28] "PKI," [Online]. Available: https://en.wikipedia.org/wiki/Public_key_infrastructure.
- [29] "M2M," [Online]. Available: https://en.wikipedia.org/wiki/Machine_to_machine.
- [30] "I2C," [Online]. Available: <https://en.wikipedia.org/wiki/I%C2%B2C>.
- [31] "SPI," [Online]. Available: https://en.wikipedia.org/wiki/Serial_Peripheral_Interface.
- [32] "GPIO," [Online]. Available: https://en.wikipedia.org/wiki/General-purpose_input/output.
- [33] "ISO14443," [Online]. Available: https://en.wikipedia.org/wiki/ISO/IEC_14443.
- [34] "Hyper-ledger fabric," [Online]. Available: <https://www.hyperledger.org/projects/fabric>.
- [35] "Hyper-ledger fabric prerequisites," [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/latest/prereqs.html>.
- [36] "Go," [Online]. Available: <https://golang.org/>.
- [37] "Node.js," [Online]. Available: <https://nodejs.org/>.
- [38] "OpenABE," [Online]. Available: <https://github.com/zeutro/openabe>.
- [39] "TEE," [Online]. Available: https://en.wikipedia.org/wiki/Trusted_execution_environment.
- [40] "Optiga Trust X," [Online]. Available: <https://www.infineon.com/cms/en/product/security-smart-card-solutions/optiga-embedded-security-solutions/optiga-trust/optiga-trust-x-sls-32aia/>.
- [41] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, 2015.
- [42] U. Raza, P. Kulkarni and M. Sooriyabandara, "Low Power Wide Area Networks: An Overview," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, 2017.
- [43] "LPWAN," [Online]. Available: <https://en.wikipedia.org/wiki/LPWAN>.
- [44] "TPM," [Online]. Available: https://en.wikipedia.org/wiki/Trusted_Platform_Module.
- [45] "Kafka streams," [Online]. Available: <https://kafka.apache.org/documentation/streams/>.