



Horizon 2020 Program

Dynamic countering of cyber-attacks

SU-ICT-2018



Cyber security 4.0: Protecting the Industrial Internet of Things

D4.2: C4IIoT Minimum Viable Product [†]

Abstract: This deliverable serves as an accompanying report which refers to the release of the MVP of C4IIoT. The described work gives emphasis to the development of the envisioned MVP integrated C4IIoT prototype, ensuring a smooth and effective integration of the components that compose this proof of concept demonstrator.

Contractual Date of Delivery	31/05/2020
Actual Date of Delivery	31/05/2020
Deliverable Security Class	Public
Editor	<i>Spyridon Vantolas (AEGIS), Mohammad Hamad (AEGIS)</i>
Contributors	AEGIS, FORTH, ITML, UNSPMF, UoG, VIP
Quality Assurance	<i>Zarko Bodroski, Srdan Skrbic (UNSPMF), Alberto Terzi, (HPE)</i>

[†] 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.

The *C4IIoT* Consortium

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

Document Revisions & Quality Assurance

Internal Reviewers

1. *Zarko Bodroski, Srdan Skrbic (UNSPMF)*
2. *Alberto Terzi, (HPE)*

Revisions

Version	Date	By	Overview
0.11	29/05/2020	AEGIS	Final document
0.10	26/05/2020	HPE	Review
0.9	21/05/2020	UNSPMF	Review
0.8	18/05/2020	AEGIS	Draft prepared for internal review
0.7	15/05/2020	UoG	Input from UoG
0.6	13/05/2020	VIP	Input from VIP
0.5	11/05/2020	FORTH, ITML	Input from FORTH, ITML
0.4	10/05/2020	UNSPMF	Input from UNSPMF
0.3	01/04/2020	AEGIS	First draft
0.2	25/03/2020	All partners	Comments on the ToC.
0.1	26/02/2020	AEGIS	ToC.

Table of Contents

LIST OF FIGURES	5
LIST OF ABBREVIATIONS.....	6
EXECUTIVE SUMMARY	7
1 INTRODUCTION	8
1.1 PURPOSE OF THE DOCUMENT	8
1.2 RELATION TO OTHER TASKS AND WORK PACKAGES	8
2 MVP DEFINITION	9
2.1 MVP – LEAN STARTUP METHODOLOGY	9
2.2 C4IIoT MVP	10
3 MVP ARCHITECTURE	11
3.1 ARCHITECTURE OVERVIEW	11
3.2 DEPLOYMENT	11
3.3 MVP INTEGRATION	13
3.4 MVP COMPONENTS	15
3.4.1 BACS	15
3.4.2 MEDICI	17
3.4.3 Traffic Analysis	17
3.4.4 Data Fusion Bus	17
3.4.5 Visualisation Toolkit	21
4 MVP USE CASE SCENARIOS	23
4.1 ENABLING SECURITY OF IIoT IN INBOUND LOGISTICS (LOGISTICS 4.0)	23
4.2 ENABLING SECURITY OF IIoT IN A SMART FACTORY	25
5 CONCLUSIONS	27

List of Figures

Figure 1: Lean Startup build-measure-learn feedback loop	9
Figure 2: MVP Architecture	11
Figure 3: MVP Deployment	12
Figure 4: Mobile operator infrastructure	13
Figure 5: FG components communication	14
Figure 6: Cloud layer components communication.....	15
Figure 7: Alerts to Kafka from Traffic Analysis	17
Figure 8: Logistics Field Gateway	19
Figure 9: Smart Factory Field Gateway	20
Figure 10: Cloud.....	20
Figure 11: AVT List of devices and real-time data.....	21
Figure 12: AVT - Detected Events overview and details.....	22
Figure 13: Modules communication (Logistic).....	24
Figure 14: Modules communication (Smart Factory)	26

List of Abbreviations

AGV	Autonomous Ground Vehicles
AVT	Advanced Visualisation Toolkit
BACS	Behavioural Analysis and Cognitive Security
BTS	Base Station Receiver
DFB	Data Fusion Bus
EC	European Commission
FCA	Fiat Chrysler Automobiles
FG	Field Gateway
LTE	Long-Term Evolution
MVP	Minimum Viable Product
NB	Narrow Band
VPN	Virtual Private Network
WP	Work Package

Executive Summary

The current deliverable presents the work that has been carried out towards the delivery of the C4IIoT Minimum Viable Product - MVP. The development of the MVP demonstrates the effective integration of C4IIoT components into a simple, yet substantial integrated prototype with minimum functionality that showcases the potential of the proposed solution.

The MVP will serve as the basis for further developments and will drive the implementation toward the release of complete prototypes (M18-M30). MVP release will act as proof of concept demonstrator and additionally will be used to approach C4IIOT stakeholders and validate fundamental business hypotheses (or leap-of-faith assumptions).

The deliverable is organized into five sections whose purpose is briefly described next.

Section 1 introduces the deliverable and highlights relationship to other C4IIoT deliverables and tasks.

Section 2 presents the MVP definition and the Lean Startup Methodology that supports the rationale behind the MVP development and the subsequent steps towards the establishment of fully functional prototypes.

Section 3 describes the C4IIoT MVP architecture and provides deployment details of the MVP infrastructure. The section also describes the integration actions and the communication mechanism used to connect the components. Finally, lists the work performed, for all the involved components, in the context of the MVP development.

Section 4 presents the use case scenarios with several details and data samples.

Section 5 highlights the overall conclusions and future plans.

1 Introduction

1.1 Purpose of the document

The document aims to present the development of C4IIoT MVP version, a prototype with minimum functionality that showcases the potential of the proposed solution and forms the basis for further developments towards the release of the first and final versions of the C4IIoT integrated prototypes.

1.2 Relation to other Tasks and Work Packages

This deliverable is closely related to all the deliverables of WP4 "An end-to-end integrated industrial IoT cybersecurity framework". Furthermore, there is a close interrelation between this deliverable and tasks performed for WP2 "Edge computing cybersecurity technologies" and WP3 "Cyber assurance and protection in an industrial cloud infrastructure".

The deliverable is strongly connected to D2.2. "Deep learning breakthroughs and security-aware dynamic offloading mechanisms", with a detailed description of the anomaly detection modules integrated in the MVP prototype.

Moreover, the deliverable is the first implementation of the D1.2 "Architecture definition".

2 MVP Definition

2.1 MVP – Lean Startup Methodology

A **Minimum Viable Product (MVP)** is a version of a product with just the core features to satisfy early customers or end-users and to allow the development teams to collect the maximum amount of feedback about the product with the minimal effort. An MVP should demonstrate sufficient features and ensure the usability of these features that encourage the end-users or the customers to use it and provide a feedback about it. Such feedback is the primary benefit of the MVP since it will be used to guide the future development of the product, to make better understanding about customer's interests, and to facilitate the testing process of the product features and assumptions. The minimum viable product is "that version of a new product a team uses to collect the maximum amount of validated learning about users with the least effort", according to Eric Ries, the author of "The Lean Startup"¹.

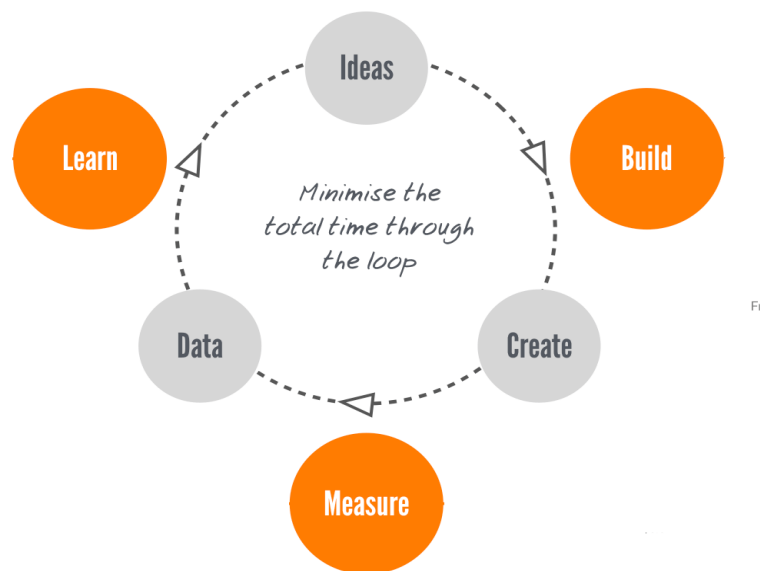


Figure 1: Lean Startup build-measure-learn feedback loop

A core component of the Lean Startup methodology is the build-measure-learn feedback loop (Figure 1). The first step is to figure out the problem that needs to be solved and then to build an MVP to begin the process of learning as quickly as possible. Once the MVP is established, the team can work on tuning the engine. This will involve measurement and learning and must include actionable metrics that can demonstrate cause and effect question.

¹ <https://leanstartup.co/what-is-an-mvp>

2.2 C4IIoT MVP

The C4IIoT MVP is the initial integrated version of the C4IIoT system, which demonstrates the functionality of the system based on predefined basic scenarios. The MVP is a simple but meaningful prototype that showcases the potential of the proposed solution by integrating proposed components and validates that the goal is approached in a reasonable manner. The C4IIoT MVP was implemented for both proposed use cases, Logistics and Smart Factory.

Even though our intention was to integrate the MVP version in pilot's site (for both use cases), the Covid-19 disease outbreak, imposed restrictions or total shutdown in companies' operation and affected the transportation and travel domain as well, for several weeks. To cope with this situation without affecting the development process, we decided to simulate the real environment with synthetic data.

Data from the IIoT infrastructure (simulated) are collected and fed to the first tier of lightweight anomaly detection at the lower-most, the edge nodes layer. Following this, data is forwarded to the intermediate, Field Gateways (FGs) layer, where a security-aware dynamic offloading decision mechanism decides if an anomaly detection task should be triggered in the FG or the Cloud. FG anomaly detection in incoming data streams will be covering a set of the C4IIoT edge node devices. Finally, data streams are fed to, the top-most, Cloud layer, where we will perform anomaly detection for the complete IIoT ecosystem.

The data streams and the detected anomalies will be saved in the storage and then used by the visualisation toolkit to provide meaningful information to the end-user.

Logistics and Smart Factory are the two use cases selected for the project and are both implemented in the C4IIoT MVP.

Inbound logistics must deal with the complex problem of managing different parts from many origin points to manufacturing plants. IIoT is the key to enable the optimisation of the logistics along the supply chain (localisation, status, conditions, etc), to maintain low stocks along the supply chain and inside the plant, to reduce working capital and warehousing costs, and to avoid stock-outs that could cause a production stop in the plant. FCA involves all suppliers and carriers to foster end-to-end visibility of components in containers, also trying to cover inter-regional flows, which are managed by inter-regional poles (warehouses and companies).

In this case microcontrollers, integrated with the containers on trucks or warehouses, connect to a Field Gateway that resides in the mobile network operator infrastructure using mobile network. For MVP we use synthetic data that contain critical parameters such as localization. These data are simulating real-life data coming from the devices attached on the containers of the pilot.

On the other hand, Smart Factory operations include several elements, such as asset management, intelligent manufacturing, performance optimization and monitoring, planning, human machine interaction, end-to-end operational visibility, and the cyber-physical systems as we know them from Industry 4.0. MVP data for this use case include sensor measurements for Automated Ground Vehicles (AGVs) as provided by FCA which contain acceleration and velocity values for the AGVs' engines.

The C4IIoT MVP aims to demonstrate the potential capabilities of the framework and receive feedback from end-users early enough to undertake any required adjustments before the first integrated prototype is released. Following the MVP approach will allow the consortium to rapidly provide immediate value while minimizing the development effort and gathering of data and feedback towards future improvements. Moreover, it will provide evidence on the benefits of the C4IIoT system and support the dissemination and exploitation activities.

3 MVP Architecture

3.1 Architecture overview

In this subsection, we provide an overview of the MVP architecture. Figure 2 illustrates a high-level overview of our MVP architecture and the component within each one of the three layers of our system: Edge, Field Gateway (FG), and Cloud. In the upcoming subsections, we will describe the component within each layer in more details.

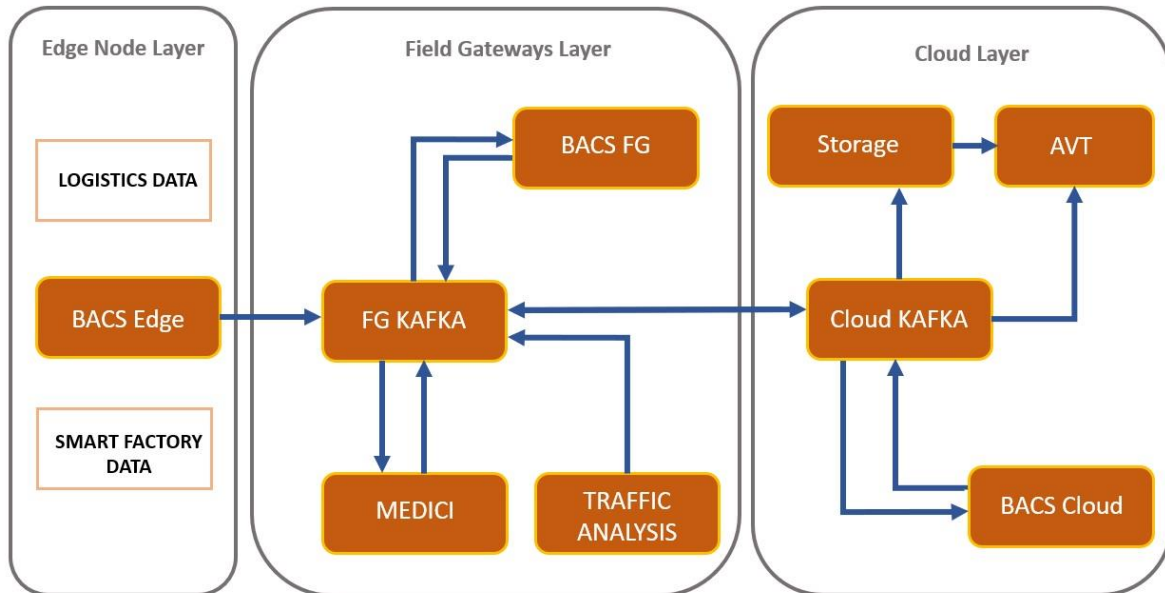


Figure 2: MVP Architecture

3.2 Deployment

The infrastructure used to deploy the components of the MVP consists of the devices residing at the edge, 2 servers acting as FGs for each use cases respectively and one server acting as the Cloud layer as depicted in Figure 3:

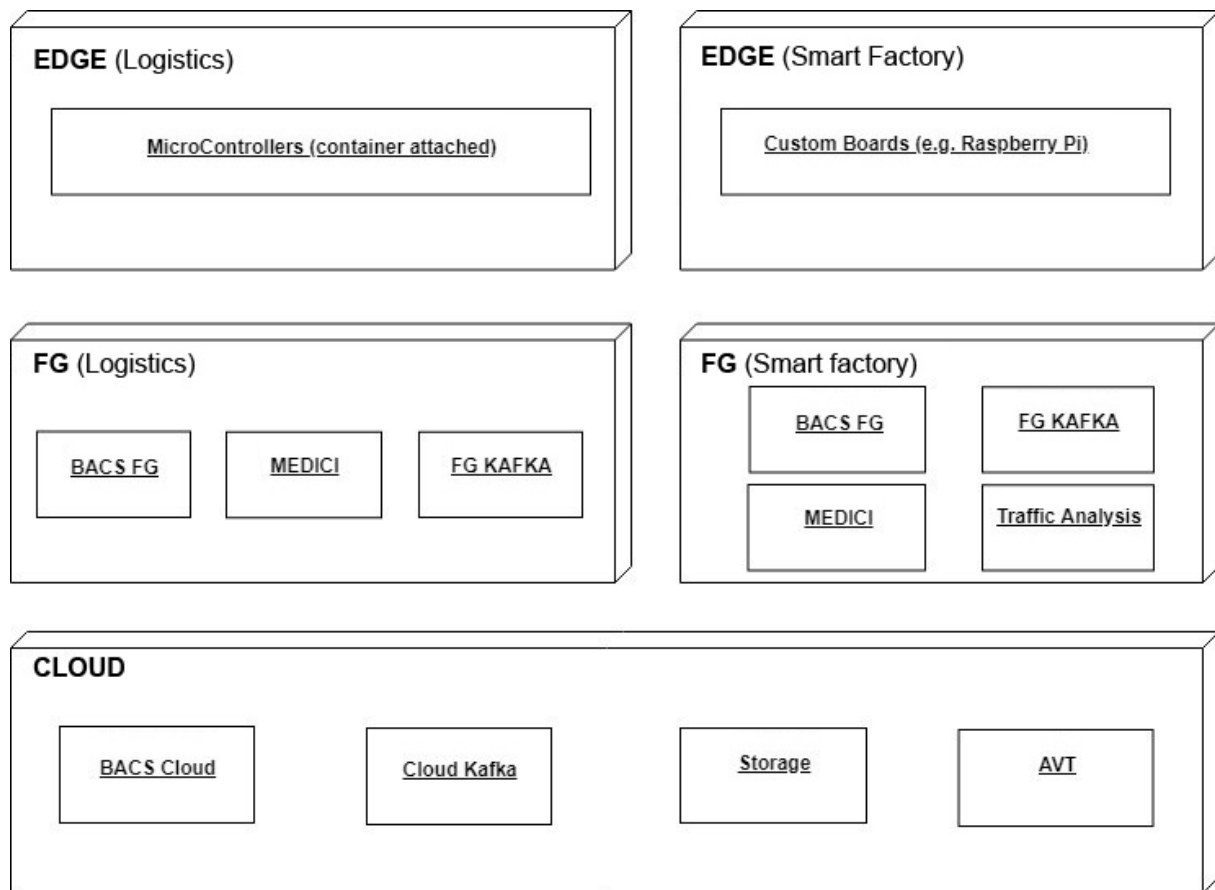


Figure 3: MVP Deployment

The **edge node layer** contains two types of devices. The first one includes 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 that of the previous ones, so at this level it is possible to execute more complex algorithms related to machine learning at the edge (Figure 3).

Field gateways forms the second layer of our architecture. As shown in Figure 4, in the Logistics case the Field Gateway will be located in the mobile operator (VIP) premises, protected by strong security mobile operator infrastructure. The Edge nodes are directly accessible from the Field Gateway. Communication between edge devices and Field Gateway will not leave trusted mobile network infrastructure. Additionally, this communication is very well encrypted with standard set of protocols, certificates, procedures, and best practices applied to the BTS (Base Station Receiver) and other LTE network elements. Only data sent during offloading to the cloud procedure will reach the Internet. To provide end-to-end secure communication, additional mechanism like firewall rules and Virtual Private Networks (VPNs) are applied.

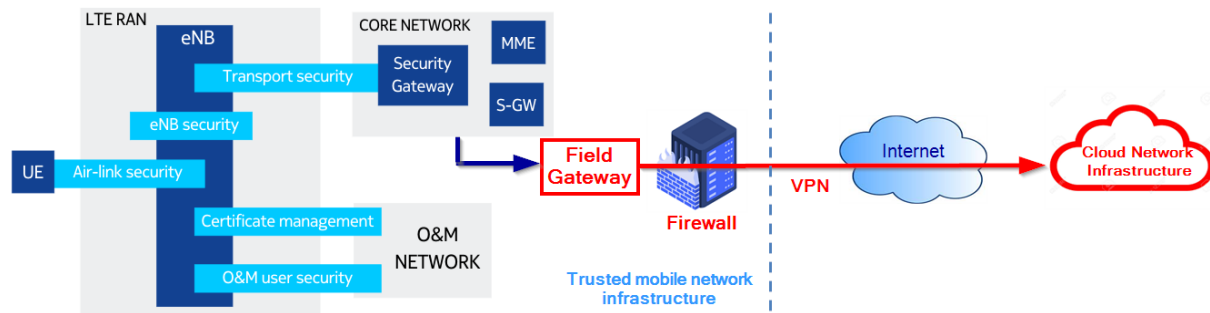


Figure 4: Mobile operator infrastructure

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. Edge node will send data over the air using NB-IoT technology provided by mobile operator (VIP).

Narrowband Internet of Things (NB-IoT) is a recent addition to the 3GPP standards offering low power wide area networking (LP-WAN) for a massive amount of IoT devices communicating at low data rates in the licensed bands. NB-IoT was designed to support very low power consumption and low-cost devices in extreme coverage conditions.

In *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 Wi-Fi communication. Each plant will have dedicated Field Gateways. Additional security mechanism (VPNs and Firewall rules, authentication) will be applied to establish secure connection with the cloud infrastructure and prevent a disclosure of private information. For the VPN we can adopt authentication based on certificates that allow a secure way to authenticate the FG and the Cloud Infrastructure. For the purposes of MVP, a dedicated server outside factory (provided by ITML) simulates the "Factory Field Gateway".

In both cases, Field Gateway will run the following SW modules:

- Software modules realizing both unsupervised and supervised anomaly detection in IIoT (BACS FG provided by UNSPMF)
- The C4IIoT security-aware dynamic offloading decision module (MEDICI provided by UOG)

Moreover, the Field Gateway of the Smart Factory use case will also include the Traffic Analysis module.

The **Cloud layer** is where advanced anomaly detection takes place using data aggregated and offloaded from the Field Gateways. It contains the storage of data and analytics produced by the rest of the components and also the user interface that exposes the monitoring and event detection related information to end users. The Cloud layer in the scope of MVP is simulated from a dedicated server provided by ITML, as seen in Figure 3 above.

3.3 MVP integration

After all the components and functionalities of the MVP are defined and developed, an initial deployment of the components and the definition of their interactions took place. This section

describes the basic workflows of the platform that served as input for the integration of the components in the MVP.

There are three different Kafka Docker components part of the Data Fusion Bus (DFB) running on the servers. Two dockers hosts the environment for the FGs (Logistics and Smart Factory) and one for the Cloud. On the FGs the Kafka brokers facilitate the communication between the edge device simulators and the components deployed on FGs. The communication between components of the system is achieved by publishing and subscribing to streams of records, in the data fusion bus, called **topics**.

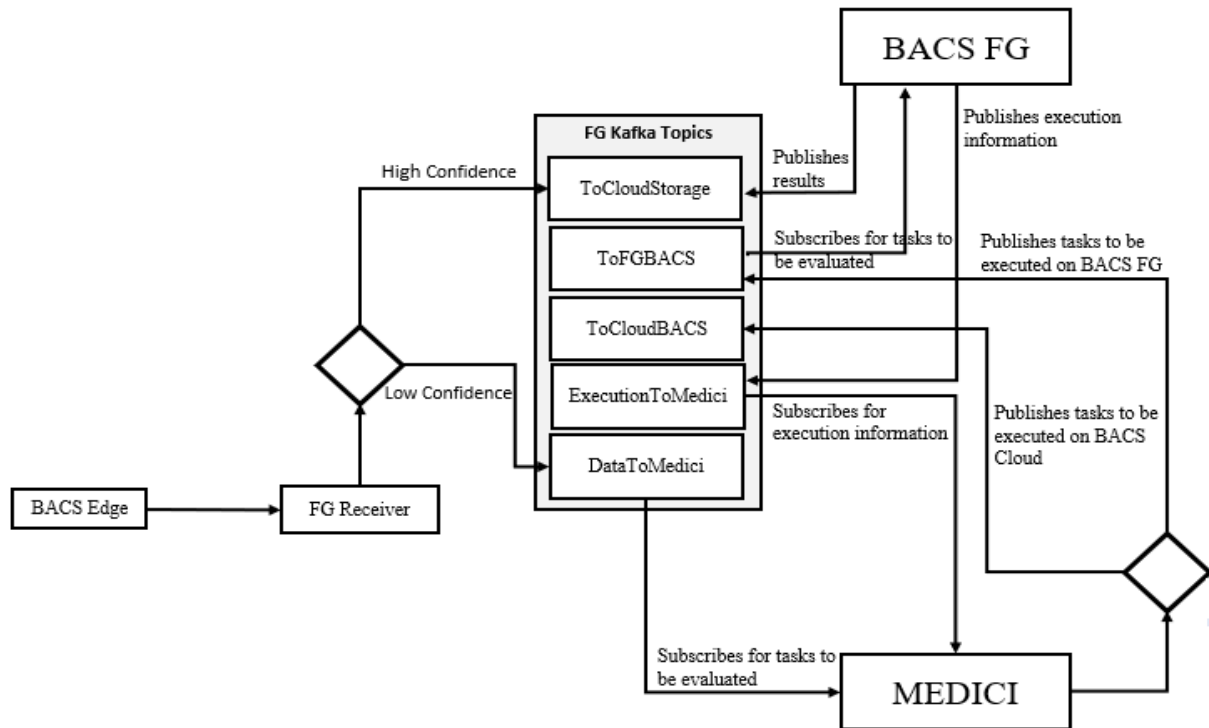


Figure 5: FG components communication

Two simulators, one for each use case, simulating the input from IoT devices at the edge node. Depending on the confidence level of BACS edge model, by starting the edge simulators procedure, data are going to be either published to the cloud storage (topic ToCloudStorage) or to a topic where the MEDICI component is a subscriber (topic DataToMedici). After proceeding the initial iteration inside FG Kafka Topics component, MEDICI the data to the appropriate topic so they can be re-evaluated by either the BACS model running on the FG (topic ToFGBACSL1 for the Logistics use case and ToFGBACSF1 for the factory use case) or the ones running on the Cloud (topic ToCloudBACSL1 for Logistics and ToCloudBACSF1 for Smart Factory use case). The results will be written on the ToCloudStorage topics that exist on both FGs and on the Cloud. They will also separately report to the appropriate topics their execution information which is used by the MEDICI component (topic ExecutionToMediciL on Logistics FG and Cloud and ExecutionToMediciF on factory FG and Cloud). The Traffic Analysis component also publishes data on the factory FG ToCloudStorage topic (Figure 5).

There are three MirrorMakers (Section 3.4.4) deployed on each FG and the Cloud layer. These MirrorMakers are used to transfer data from ToCloudStorage, ToCloudBACSL1 and ToCloudBACSF1 on the FGs which send the data to their respective topics on the Cloud and ExecutionToMediciL and ExecutionToMediciG which send data from the Cloud to the respective topics on the FGs.

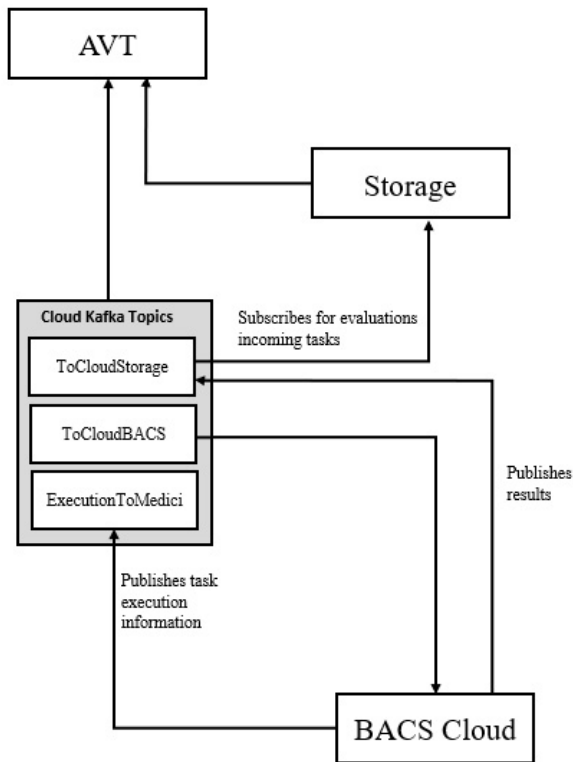


Figure 6: Cloud layer components communication

The ToCloudStorage topic on the Cloud, through Kafka connect, stores the data on the Elasticsearch component. AVT component subscribes to this topic in order to receive real time information or retrieve historical data from the Elasticsearch component (Figure 6).

3.4 MVP Components

3.4.1 BACS

BACS (Behavioural Analysis and Cognitive Security) is a collection of software modules realizing both unsupervised and supervised anomaly detection in IIoT sensory data and network traffic flows based on machine and deep learning algorithms. Unsupervised anomaly detection BACS modules identify large and unusual deviations from expected device or network behaviour. Supervised BACS anomaly detection modules detect specific security and operational threats for which they were explicitly trained. BACS support both standalone and federated learning of anomaly detection models. Standalone anomaly detection models are trained from centralized datasets. In federated learning settings, anomaly detection models are collectively trained without any data exchange between C4IIoT devices.

BACS modules are realized in Python and C. The implementation of BACS modules started with this project and it will be continuously expanded to fulfil necessary behavioural analysis and cognitive security requirements at various granularity levels. At the moment BACS provides fully operational unsupervised anomaly detection at the level of individual data points and multivariate time series. This is achieved by utilizing various outlier detection and representation learning algorithms, including also (deep) autoencoders realized using Tensorflow. Regarding supervised anomaly detection, BACS currently provides identification of two types of network attacks (DDoS and PortScan). The current status of BACS and future development plans are in more details described in Deliverables 3.1 and 2.2.

Software components derived from BACS modules are present at all three layers of the C4IIoT architecture:

- **BACS Edge** performs anomaly detection considering data collected at a particular C4IIoT edge node device,
- **BACS FG** is responsible for anomaly detection in data streams coming to a C4IIoT Field Gateway covering a set of C4IIoT edge node devices,
- **BACS Cloud** performs anomaly detection at the level of the whole C4IIoT operational setting.

In the C4IIoT Logistics use case **BACS Edge** executes on low-power NB-IoT devices without an operating system. For this particular use case, considering all present constraints, anomaly detection in **BACS Edge** is performed by lightweight autoencoders (autoencoders with one hidden layer) implemented in C that are directly integrated in the device's firmware. Lightweight autoencoders are trained offline due to the processing and capacity constraints imposed by NB-IoT devices.

In the C4IIoT smart factory use case **BACS Edge** runs on more powerful RaspberryPi-based C4IIoT devices. In this use case, **BACS Edge** is able to perform anomaly detection using any of the methods available in the BACSPY package of BACS (Python-based implementations of anomaly detection methods relying on scikit-learn, PyOD and Tensorflow libraries, for more details please see Deliverable 3.1). A Tensorflow-based autoencoder working on individual data points is the default unsupervised anomaly detection method for the smart-factory use case.

BACS contains generic realizations of **BACS FG** and **BACS Cloud** that are further specialized to fit needs of a particular use case. Both components are implemented in Python. **BACS FG** detects anomalies in multivariate time series using methods available in the BACSPY package. **BACS Cloud** also identify anomalies in multivariate time series (which are longer than time series inspected by **BACS FG**), but it may use both methods present in the BACSPY package (non-partitioned anomaly detection models) and the BACSCCL package (partitioned anomaly detection models; for more details please see Deliverable 3.1). Tensorflow-based autoencoders working on multivariate time series of length 5 and multivariate time series of length 10 are default unsupervised anomaly detection methods for **BACS FG** and **BACS Cloud**, respectively (in both use cases).

The role of the BACS components within the C4IIoT system is the following. **BACS Edge** makes an initial anomaly detection decision for a currently observed data point, additionally providing a confidence score (a real number in the interval $[0, 1]$). Data points with high confidence scores are immediately sent to the cloud storage. For data points with low confidence scores, Medici activates either **BACS FG** or **BACS Cloud** by sending a message to an appropriate Kafka topic. Considering the last k data points (k is the length of time series) emitted by the device, **BACS FG** and **BACS Cloud** re-evaluate the initial decision made by **BACS Edge**. Then, the final decision is sent to the cloud storage, while the execution time and confidence score are reported to Medici via a Kafka topic dedicated for that purpose (Figure 2).

For the integration and testing purposes of BACS components we have also implemented simulators of C4IIoT edge node devices for both use cases.

3.4.2 MEDICI

In the MVP, for both use cases, UOG's MEDICI tool will be responsible for deciding which of the low confidence decisions of the edge BACS anomaly detection module, should be further offloaded to another BACS module, either at the Field Gateway or at the Cloud. To do so, MEDICI subscribes to the "DataToMEDICI" FG Kafka Topic, which provides the requests from the low confidence anomaly detections of the Edge BACS.

Then, based on parameters such as previous execution times and confidence levels of the different BACS implementations, MEDICI will choose to trigger the most appropriate BACS implementation, in a timely and confident manner. The decision will then be published to either the "ToFGBACS" or "ToCloudBACS" in the appropriate Kafka service. To inform its future decisions, MEDICI also subscribes to the "ExecutionToMEDICI" Kafka Topics which provide MEDICI with the actual execution times, after the anomaly detection task has finished running in the corresponding BACS module (Figure 5). More about the operation of MEDICI can also be seen in deliverable D2.2.

3.4.3 Traffic Analysis

The Traffic Analysis module utilises signatures to detect specific traffic patterns inside network flows. The signatures will be generated during the training stage, where public datasets of malicious traffic from previous research work will be used as ground truth. These datasets contain attacks from IoT environments, like worms and botnets and more general attacks. Our signatures relay on packet metadata (e.g. packet-length, direction, timestamp) in order to be able to be used for matching even if the traffic is encrypted.

For the MVP, the Traffic Analysis module will run on the Factory FG simulator inspecting all the network traffic which goes through it. Using a preliminary version of our dataset of malicious signatures it will conduct signature-based detection, at this point it is capable of capturing portscan attempts, login attempts and traffic from certain IoT botnets. Both normal status alerts and detection of attacks will be published to the "ToCloudStorage" topic of the Kafka at the FG (Figure 2).

In the Figure 7 below we can see an example of the format of the alerts.

```
[{"device_id": "tr_fg_factory", "timestamp": "2020-05-14 10:48:06", "attack": 0}
{"device_id": "tr_fg_factory", "timestamp": "2020-05-14 10:48:06", "attack": 1, "AttackType": "portscan", "ip_src": "192.168.2.2", "ip_dst": "192.168.2.6"}
{"device_id": "tr_fg_factory", "timestamp": "2020-05-14 10:48:11", "attack": 0}]
```

Figure 7: Alerts to Kafka from Traffic Analysis

More details about the operation of Traffic Analysis component can also be seen in deliverable D2.2.

3.4.4 Data Fusion Bus

In the current implementation, the data fusion bus is a component which ensures the transfer of data between FG and Cloud components and the permanent storage. It comprises a collection of dockerized open source components which allows easy deployment and configuration as needed. Its main component is Apache Kafka which is used to transfer streams of data using publish/subscribe (pub/sub) messaging. This is complemented by Apache Zookeeper which acts as the server which handles services such as Kafka, Kafka MirrorMakers which transport

the data from one Kafka topic to another and Elasticsearch which allows the storage and search of the data. This setup makes it extremely fault tolerant and able to handle large amount of data.

It currently has components deployed on the three servers which have the roles of the FG for the Logistics and Smart Factory use case and the Cloud.

Apache Kafka

Apache Kafka is used for "building real-time streaming data pipelines that reliably get data between systems or applications"² supporting multiple sources that can even be in a distributed system. As a message broker it offers higher fault tolerance in comparison to traditional message brokers and it can support significantly higher throughput. It is thus capable of handling great amounts of data and deliver them in real time to the components that require them. It can support multiple topics, which are used by producers and consumers of data to write and read data from respectively. A new topic can be easily created when needed in order to support the communication between new components or the new component can become a publisher or subscriber of an already existing topic. Multiple producers can write messages in the same topic and multiple subscribers can receive messages from the same topic. Subscribers can either compete for the same message, so that only one of them will receive it if they belong to the same group or they can all receive it if it is needed by multiple components. Apache Kafka offers various options for security as well, which include encryption of the data and client authentication and authorization, allowing only specific publishers and subscribers per topic.

Elasticsearch

Elasticsearch³ resembles a nosql database being able to store data as json documents, but also works as a powerful search engine. Data is stored in the indices, which can be separated into shards and stored into a distributed system with replicas if needed. In contrast to traditional databases, the stored documents can have different format. This component is currently the used for storing the data from the edge devices so that it can be retrieved when needed.

Apache Zookeeper

Apache Zookeeper⁴ is the server on which the Kafka is deployed. ZooKeeper is a centralized service for maintaining configuration information, naming, providing distributed synchronization, and providing group services It can support multiple clusters if needed and ensure the availability of the system.

Kafka MirrorMaker

Kafka⁵ mirroring is used to transfer data from the topic of one system to another. It is essentially a packaged consumer-producer which subscribes to one topic and publishes to another. Multiple

² <https://kafka.apache.org/intro>

³ <https://www.elastic.co/>

⁴ <https://zookeeper.apache.org/>

⁵ <https://cwiki.apache.org/confluence/pages/viewpage.action?pageId=27846330>

MirrorMakers can be used in order to forward the data between two or more different Kafka deployments.

Kafka Connect

Kafka Connect⁶ is a component that moves the data from the appropriate Kafka topic to the Elasticsearch component for storing. It can easily handle high traffic of data and it can be set to evaluate the schema of the incoming data.

Data Fusion Bus Setup

The following figures present a detailed view of the DFB setup (the above described components) in the FGs for both use cases (Logistics (Figure 8) and Smart Factory (Figure 9)) and the Cloud server (Figure 10).

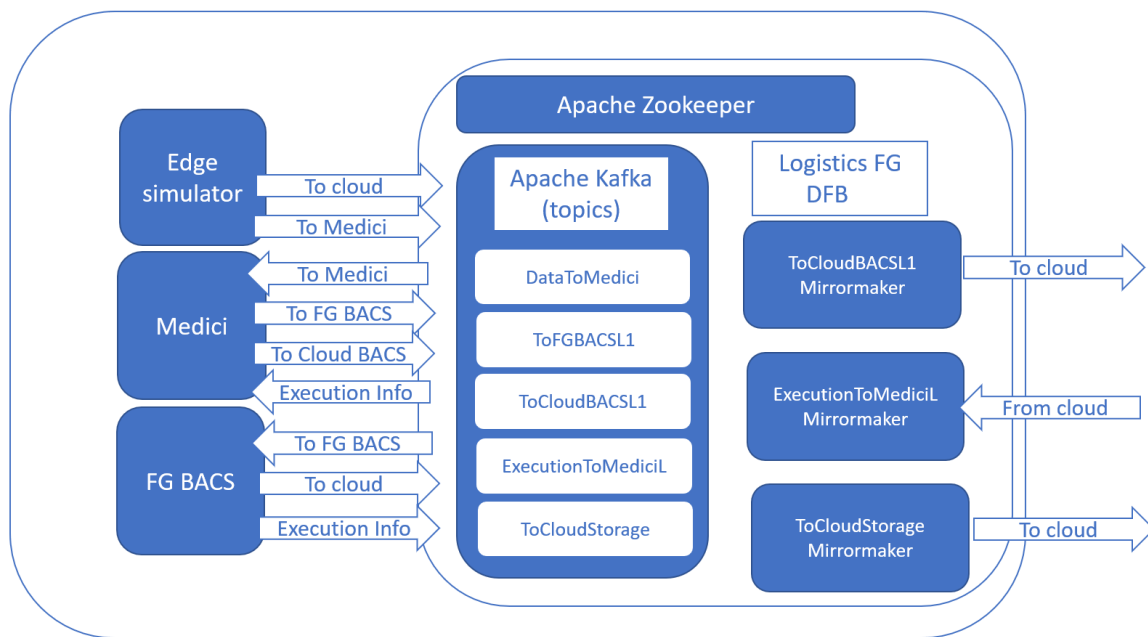


Figure 8: Logistics Field Gateway

⁶ <https://docs.confluent.io/current/connect/index.html>

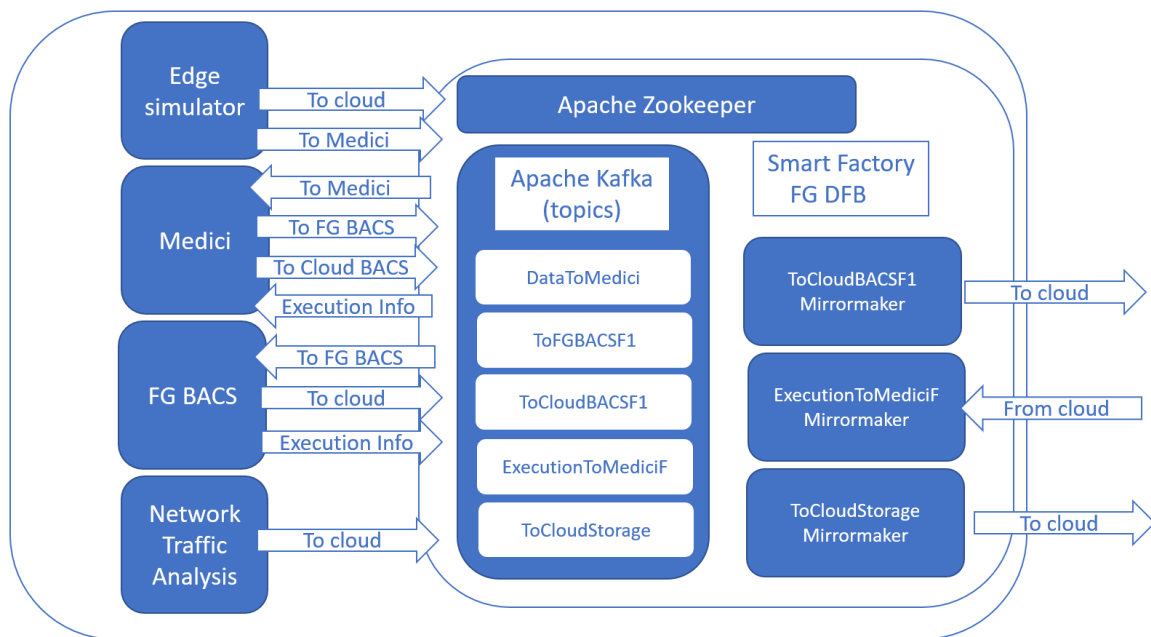


Figure 9: Smart Factory Field Gateway

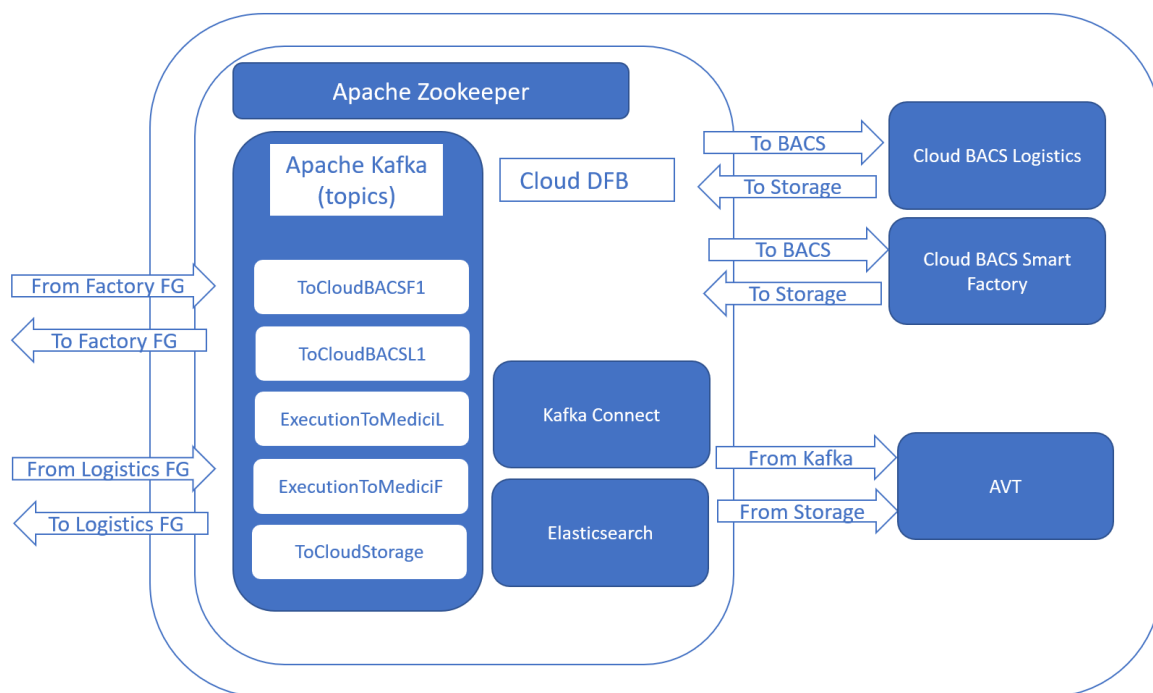


Figure 10: Cloud

3.4.5 Visualisation Toolkit

The Advanced 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 data coming from the Edge layer. AVT 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.

AVT supports a timeline analysis component and multiple visualisations. 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.

In the scope of the MVP, the toolkit connects to Cloud Kafka to retrieve real-time data showing the IIoT devices functional data and events from the anomaly detection procedure. On the other hand, historical data retrieved from the Elasticsearch storage are displayed so as to help users understand the evolution of the edge devices behaviour and look for potential relationships or behavioural patterns among the monitored assets.

The following images present the initial basic screens that allow users to look at an overview of the monitored devices and detected events.

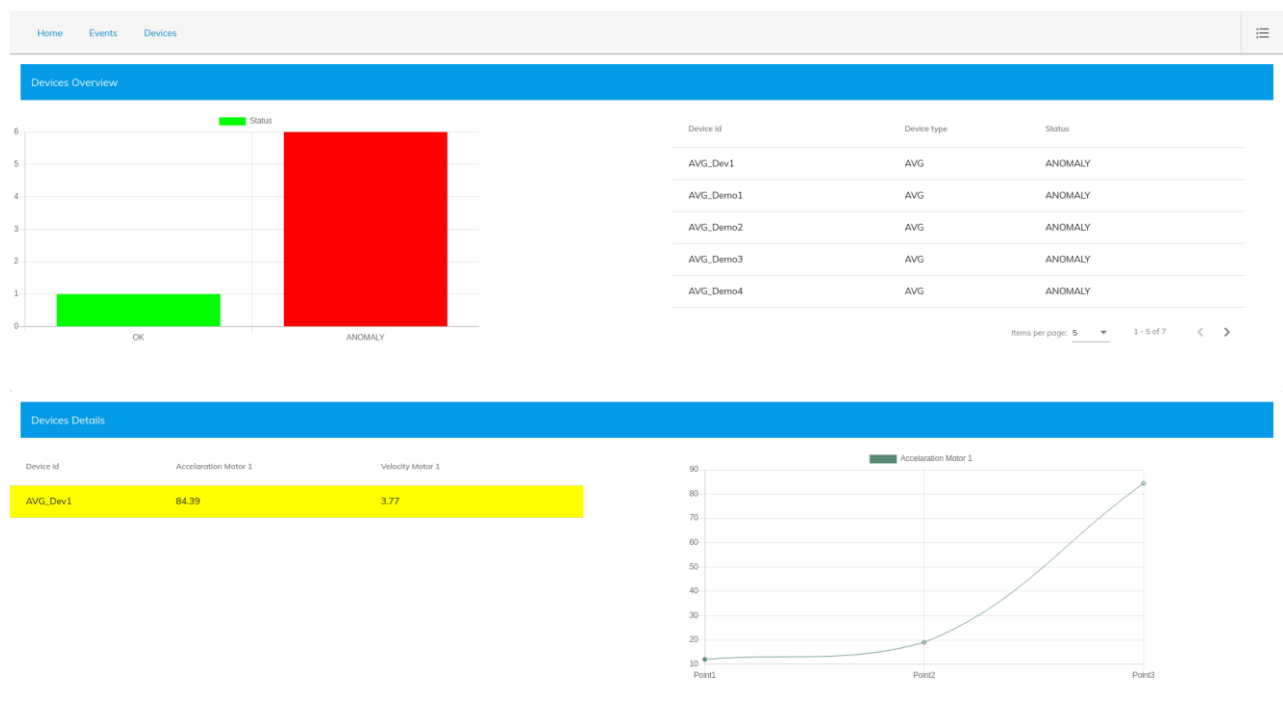


Figure 11: AVT List of devices and real-time data

Figure 11 includes the list of monitored devices and a quick view of their current status. Upon selection of a device, the details of it appear at the lower part of the screen together with a real-time visualisation of the currently received values from this specific device. This gives users the capability to view the current state and performance of any sensor with a minimum set of actions required through the interface.

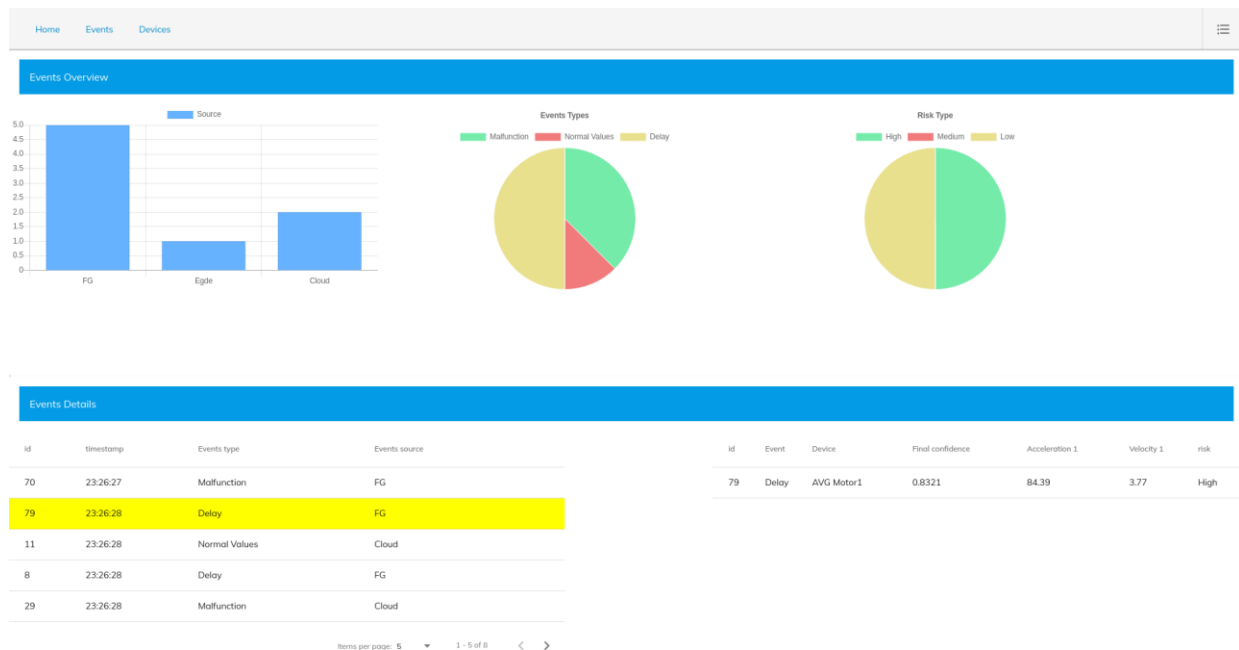


Figure 12: AVT - Detected Events overview and details

Similarly, to the devices overview, Figure 12 presents the detected events screen. The overview of currently detected events consists of a number of charts presenting the statistics of the events, i.e. source of the event (Edge, FG or Cloud), the event type and the risk factor. Although the latter are not yet integrated to the MVP, we considered that their inclusion contributes to the completeness of the presented information and therefore included them using sample data.

Moreover, clicking on an event presented in the list reveals another section on the lower right corner of the screen with details for the specific event. This section will serve as placeholder for any additional information regarding the event that will be available in the next fully integrated versions of the C4IIoT prototype.

4 MVP use case scenarios

4.1 Enabling security of IIoT in Inbound Logistics (Logistics 4.0)

Logistics must deal with the complex problem of managing different parts from many origin points to manufacturing plants and IIoT is the key to enable the optimisation of this procedure. FCA involves all suppliers and carriers to foster end-to-end visibility of components in containers. Currently, two categories of containers are used: 22 types of standard plastic containers and 150 specific metallic containers (for each car model). Altogether they move about 6000-part numbers.

Microcontrollers integrated with the containers loaded on trucks or stored in warehouses are sending sensory data via 2G/3G/4G mobile networks to the system. We will evaluate using esim or sim cards, according to the needs, to allow this communication.

For the purposes of MVP, we have collected data using the same devices attached on a vehicle moving along custom routes.

A sample of collected data has the following form:

```
{ "timestamp": "2020-03-11 14:24:45" , "GPS_lat": 45.2419, "GPS_lon": 19.83439,
"GPS_alt": 86.8, "GPS_speed" : 0.742, "GPS_num_sats": 12, "ACC_x_RMS": 6.86952,
"ACC_y_RMS": 7.24027, "ACC_z_RMS": 2.372, "ACC_samples": 670, "MAG_x_RMS":
30.1567, "MAG_y_RMS": 19.53, "MAG_z_RMS": 27.5869, "MAG_samples": 670 }
```

Among with information for the specific device like "device_id" and device_type" (in the Logistics use case there is a single device type) the information flows to the system. In the next steps, data are validated and labels are added to the message.

Figure 13 shows the detailed data flow and modules communication for the Logistics use case.

BACS on the Edge Layer performs a first assessment, on this specific device, by performing an unsupervised anomaly detection based on standalone and federated autoencoders and provides a result with a confidence level. In case that confidence level is above a selected threshold data saved to Storage in the Cloud Layer.

Otherwise, if BACS at the Edge confidence level is low, then the MEDICI module decides if an anomaly detection task should be performed in the FG BACS or the Cloud BACS. BACS FG performs anomaly detection by covering a set of C4IIoT edge node devices and sends the result to Storage or interacts with MEDICI for further evaluation.

If BACS in the Cloud receives data streams the perform anomaly detection at the level of the whole CIIoT ecosystem publishes the result to Storage.

An example of the final message to the Storage is as follows:

```
{ "timestamp": "2020-03-11 14:24:45" , "GPS_lat": 45.2419, "GPS_lon": 19.83439, "GPS_alt":
86.8, "GPS_speed" : 0.742, "GPS_num_sats": 12, "ACC_x_RMS": 6.86952, "ACC_y_RMS":
7.24027, "ACC_z_RMS": 2.372, "ACC_samples": 670, "MAG_x_RMS": 30.1567,
"MAG_y_RMS": 19.53, "MAG_z_RMS": 27.5869, "MAG_samples": 670, "edge_detection":
"ANOMALY", "edge_confidence: 0.999961439665897, "device_id": "test_device",
"device_type": "test", "iterations": 0, "final_detection_timestamp": "2020-03-11 14:25:05",
"final_detection": "ANOMALY", "final_confidence: 0.999961439665897,
"final_detection_place": "EDGE" }
```

The AVT shows to the final user detailed information of devices and detected events in real-time together with historical data in order to speed up situational awareness.

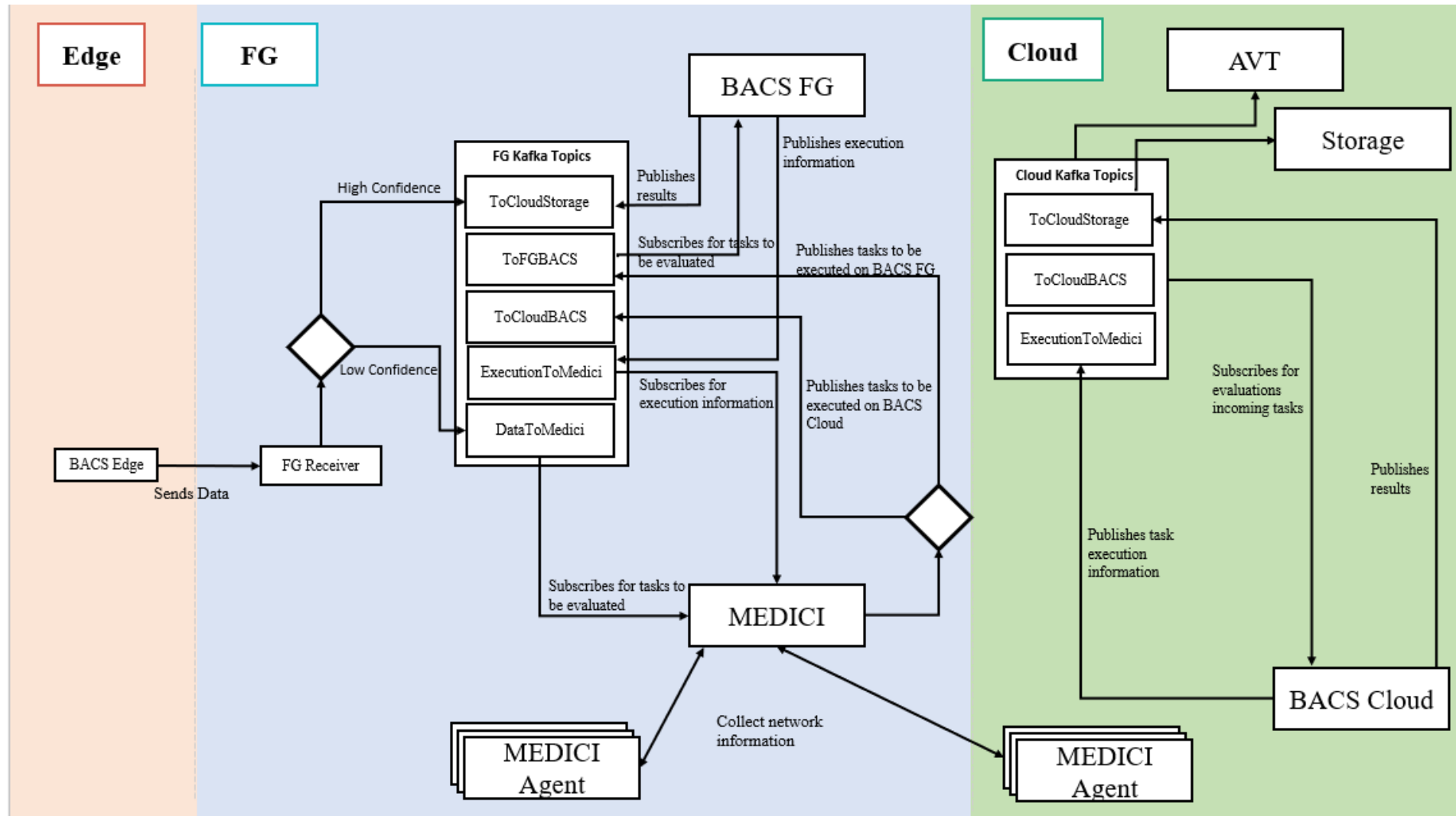


Figure 13: Modules communication (Logistic)

4.2 Enabling security of IIoT in a Smart Factory

Smart factory operations include several elements, such as asset management, intelligent manufacturing, performance optimization and monitoring, planning, human machine interaction, end-to-end operational visibility, and the cyber-physical systems.

For the purposes of MVP, we use real-life data from sensors in Automated Ground Vehicles (AGVs) as provided by FCA. Data referred to two different motors of the same AGV. The motors are of the same type. The data contain acceleration and velocity values. For each of the two motors of the AGV there is one sensor detecting both data at the same time. The data are handled in a similar way to the Logistics use case data, which we described in the previous paragraph

In addition, in the Smart Factory use case Traffic Analysis component is used to analyse the network traffic on the Factory FG simulator and send the results to storage in the Cloud.

An example of data received among with device id and type information follows:

```
{ "timestamp": "2020-04-15 13:29:17", "acceleration_motor1": 10.76, "velocity_motor1": 0.39,
  "acceleration_motor2": 0.76, "velocity_motor2": 0.59, "device_id": "AVG_Dev1",
  "device_type": "AVG" }
```

Figure 14 shows the detailed data flow and modules communication for the Smart Factory use case.

The final message sent for storage has the following form:

```
{ "timestamp": "2020-04-15 13:29:17" , "acceleration_motor1": 10.76, "velocity_motor1":
  0.39, "acceleration_motor2": 0.76, "velocity_motor2": 0.59, "edge_detection": "OK",
  "edge_confidence": 0.999961439665897, "device_id": "AVG_Dev1", "device_type": "AVG",
  "iterations": 0, "final_detection_timestamp": "2020-03-11 14:25:05", "final_detection":
  "ANOMALY", "final_confidence": 0.999961439665897, "final_detection_place": "EDGE" }
```

The AVT shows to the final user detailed information of devices and detected events in real-time together with historical data in order to speed up situational awareness.

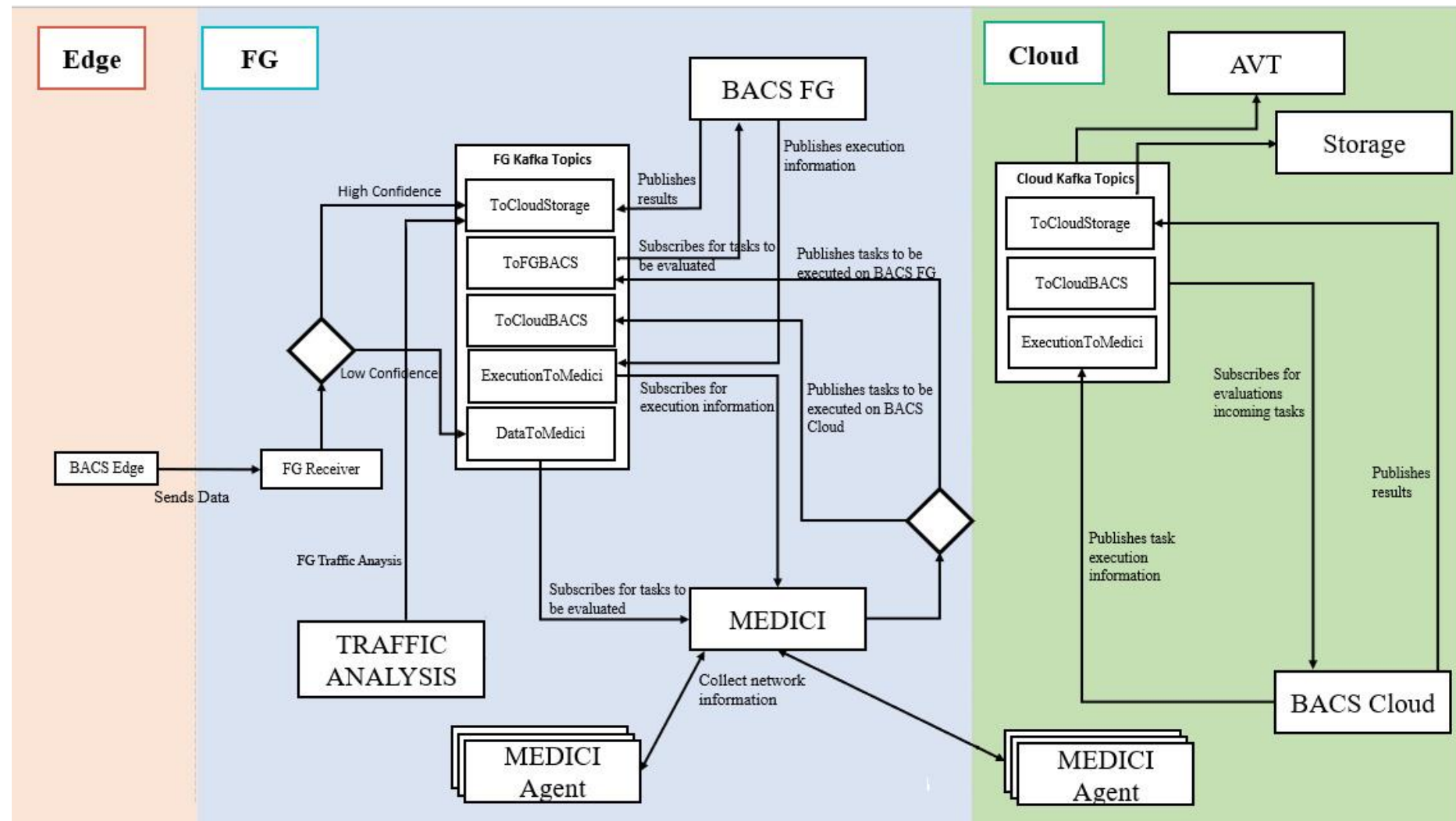


Figure 14: Modules communication (Smart Factory)

5 Conclusions

The current deliverable describes the implementation and rationale behind the C4IIoT MVP. It analyses the MVP use case scenarios and explains how and why they were selected. The main objective of these use cases is to validate and test the C4IIoT architecture and to kick-start the integration between the C4IIoT components. For each component, we describe the work performed to support the operation of the C4IIoT. We also present the communication methods and the integration elements that enabled the delivery of the MVP, based on the proposed use cases (Logistics and Smart Factory).

Following an agile development approach, the technical decisions presented in this deliverable are subject to further refinements and modifications, based on the progress of the technical work packages, as well as the feedback to be obtained from the MVP. Such advancements will be analytically reported in D4.3 “C4IIOT integrated framework”.