University of Zurich[UZH]

BACHELOR THESIS – Communication Systems Group, Prof. Dr. Burkhard Stiller

Inventorying and Secure Life-Cycles of IoT Devices

Armin Richard Veres
Zürich, Switzerland
Student ID: 20-700-118

Supervisor: Dr. Eryk Schiller
Date of Submission: December 1, 2023

University of Zurich
Department of Informatics (IFI)
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland

ifi

Bachelor Thesis
Communication Systems Group (CSG)
Department of Informatics (IFI)
University of Zurich
Binzmühlestrasse 14, CH-8050 Zürich, Switzerland
URL: http://www.csg.uzh.ch/

Abstract

Das ist die Kurzfassung...

Acknowledgments

Optional

Contents

Chapter 1

Introduction

1.1 Motivation

There are daily more and more Internet of Things, IoT, devices connected to the internet, with the need to gather and process massive amounts of real-time information, especially with 5th Generation networking, which allows for extensive information exchange.

Enhanced connectivity and adoption of IoT trigger cyber attacks, which are increasingly sophisticated and affect considerable amount of IoT-related infrastructures, raising security concerns with consumers, as well as businesses. This stresses the importance of appropriate IoT security management and enhancement of IoT life-cycle management. Considering the heterogeneity of IoT devices, the dynamism of the security landscape and number of IoT stakeholders make these tasks quite challenging, especially considering that a single insecure update can put a whole IoT system at risk.

As emphasized by the European Union Agency for Cybersecurity, ENISA, Cyber Security Act, CSA, the management of IoT infrastructures encompassing the entire life-cycle of products, as well as the continuous certification, are fundamental tools to guarantee a high level of security. [1] Also as pointed out by the Network and Information Security, NIS2, directive a pragmatic security framework must stimulate active collaboration between the IoT Stakeholders. [2]

Providing access to cybersecurity information is central for realization of a homogeneous perspective on cybersecurity. CSA and NIS promote strategic cooperation among stakeholders to support and facilitate information sharing, leading to an approach that helps respond to large-scale incident by creating more effective synergies against cybersecurity vulnerabilities.

1.2 Description of Work

This thesis will develop a service to support security information sharing between IoT stakeholders to support continuous security assessment throughout the IoT device life-

cycle. We will consider the use of Distributed Ledger Technology, DLT, as a possible approach to facilitate a trustworthy and transparent platform for sharing cybersecurity information without a trusted third-party. [3] It is important to integrate the presence of several entities with different responsibilities and roles in sharing cybersecurity knowledge. So while performing security monitoring activities, the user, i.e., a device, may detect vulnerabilities that will be shared with the manufacturer for further investigation, prompting for mitigation and or resolutions. This requires the device to be re-configurable throughout its life-cycle according to the changing threat landscape and to the device manufacturers publishing of secure updates, i.e., patches, and device profiles.

Established approaches of secure IoT deployment and bootstrapping have significant challenges. Using pre-shared credentials for every device is the simplest approach, but it prevents the identification of specific devices and the verification, whether the device is corrected to the correct network. [4] This thesis will develop a bootstrapping service to provide a lightweight bootstrapping protocol, supporting different authentication methods, depending on the characteristics of the device and providing key management.

The infrastructure of the developed bootstrapping mechanism will enable the inventorying of IoT devices. Said infrastructure will keep track of all embedded IoT devices and their respective security levels. To ensure security throughout the life-cycle of a device, an update/patching mechanism will be developed, where manufacturers and software providers will provide fixes to a security issue after an attack or vulnerability detection.

As most updating proposals are based on centralized models using e.g., client-server architectures, this thesis will design a scalable and secure approach for disseminating software updates in scenarios with selected IoT devices. The design shall entail decentralization, robustness and efficiency, bringing the upgrading functionality closer to the end devices. Blockchain based technologies will be leveraged by providing a transparent ledger to manage different versions of software elements composing and IoT device or system and share relevant security aspects, such as vulnerabilities or device information. As interoperability is crucial, this thesis will analyze the use of Bifröst [5] and Interledger [6] approaches to interconnect different blockchain implementations.

Finally this thesis will consider mitigation for IoT devices using Threat Manufacturer Usage Description, MUD, proposed by NIST [7], which provides a flexible and dynamic way to alert about new threats and mitigation to apply before and update of patch is released. Threat MUD is intended as a complement to MUD file, dynamically reconfiguring a device in case of detection of a vulnerability.

1.3 Thesis Outline

This thesis has two main emphasis, Inventorying and Secure Firmware Updating.

Chapter 2

Background

2.1 Distributed Ledger Technology

2.1.1 Smart Contract

2.2 Manufacture Usage Description

MUD has been developed by the International Engineering Task Force, IETF, with following goals and intents in mind: [8]

- Substantially reduce the threat surface on a device to those communications intended by the manufacturer.
- Provide a means to scale network policies to the ever-increasing number of types of devices in the network.
- Provide a means to address at least some vulnerabilities in a way that is faster than the time it might take to update systems. This will be particularly true for systems that are no longer supported.
- Keep the cost of implementation of such a system to the bare minimum.
- Provide a means of extensibility for manufacturers to express other device capabilities or requirements.

MUD does not entail address network authorization of general purpose computers, it simply creates a suggestion than can be followed. The architecture of Devices using MUD can be seen in Figure 2.1, which is the reference architecture by NIST [7] but it can be found in similar fashion inside the RFC specification.

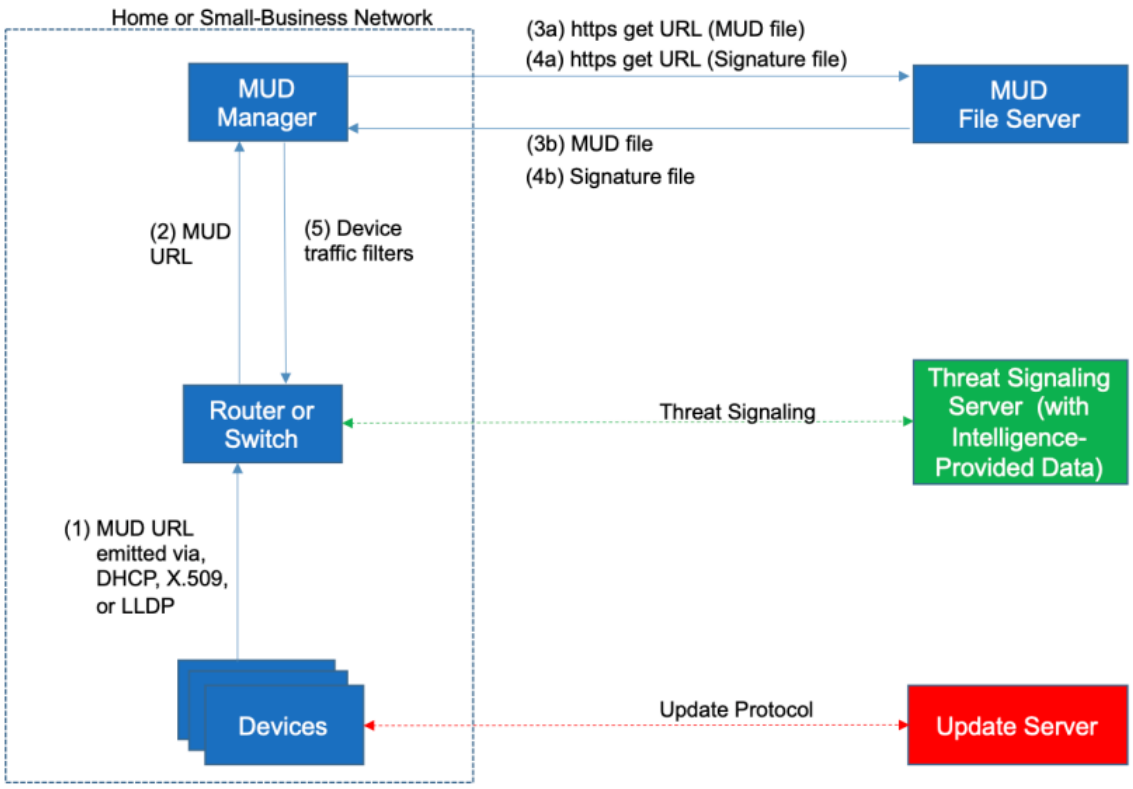

Figure 2.1: NIST MUD Reference Architecture

2.3 Physical Unclonable Function

In order to be able to track IoT nodes in a blockchain, they need to be uniquely identifiable, in our case even in a distributed manner, using e.g., Distributed Identifiers, DIDs. Common practices are based on placing a cryptographic key into a nonvolatile electrically erasable programmable read-only memory (EEPROM) or battery-backed static random-access memory (SRAM) and use hardware cryptographic operations such as digital signatures or encryption, which is all expensive in design and power consumption. [9]

PUFs are unpredictable and uncontrollable, therefore making it unclonable and an ideal security vector. They are dependent on random physical factors introduced during manufacturing, e.g., inequalities of SRAM cells, although factors such as the altering of the physical components, voltage and temperature need to be taken into account. [10] For reasons of simplicity and because it is not the main focus of this thesis, we will neglect this aspect.

By implementing the Challenge-Response Pair, CRP, is used to evaluate the microstructure, whereas a physical challenge makes the device react, the response, in an unpredictable, but repeatable way. In order to turn this 'silicon key' into a cryptographic root key, processing algorithms need to be applied, that ensure that the distribution of 0s and 1s are uniform. [9]

2.3.0.1 SRAM-Based PUF Readouts

Methods of creating identifiers that are unique to devices exist, such as SRAM-Based Physical Unclonable Function, PUF, readouts. Therein PUFs are among the most cost-effective security primitives to establish hardware trust. [11]

Even though the evaluation process of the characterization of guarantee over lifetime and differing operating conditions are still subject to research following metrics have become widespread: *reliability*, the variation of bit-wise startup patterns; *uniformity*, i.e., the repeatability and reproducibility on a given device after any amount of restarts; *uniqueness*, the probability of other devices with same signatures; *bit-aliasing*, the probability of specific bit position of the signature to be biased towards 0 or 1. [10]

Chapter 3

Related Work

3.1 CERTIFY

This thesis is carried out in conjunction with the CERTIFY project [12].

The National Institute for Standard and Technology has a few ongoing projects and white papers on security related mitigation methods for IoT devices.

3.2 DLT-based Asset-Tracking

Neisse et al. (2017) analyzed how blockchain-based approaches might be used for data accountability and provenance tracking under the then recently released GDPR legislation, highlighting challenges of scalability and considering sharding as a method to address it. [3] Further they also mentioned issues of clonability of the tracked assets, which we can also correlate to the physical assets that are tracked inside blockchain.

3.3 Cybersecurity of IoT Devices

In order to maintain participation rights for only valid users/clients, Manufacturer Usage Descriptions, MUDs, are getting more and more relevant, as also the National Institute for Standards and Technology, NIST, have been considering their use cases. [7]

3.3.1 SRAM-based PUF Readouts

We will also be considering using SRAM-based PUF readouts in our device configurations in order to get DIDs that are absolutely unique for each device.

[10]

3.3.2 Device Fingerprinting

For classification of device capabilities NIST has been considering the usage of MUDs, so that devices do not step out the bounds of their official and appointed capabilities. [7]

Chapter 4

Architecture and Design

4.1 Actors

In our architecture there will be a few actors, that play various roles, outlined in the below sections.

There will be a summarized device entity, that will be split up into two devices, as seen below in Section 4.2, and further explained in Chapter 5. There will be a Manufacturer, that is responsible for the fabrication of said devices, as well as the creation of Certificates and credentials, that will later be used by the devices and other entities for verification. Further there shall be an auditor, that inspects said credentials and verifies them, or the necessary cases also invalidation. Finally an entity Infrastructure shall exist, that will enable the communication facilities for the entities.

We will assume a simplified and already predefined *Authentication, Authorization and Accounting*, AAA, server in order to simplify our environment.

Further external stakeholders include a certification authority and a vulnerability database.

4.2 Physical Components

The composition of our physical architecture will entail two main devices of importance for us, as later defined in more detail in Chapter 5, Section 5.3.

4.2.1 Edge Nodes

The edge nodes, also Internet of Things, IoT, nodes, are the frontier where direct interaction happens, be that sensing, actuating or interaction through a Human Machine Interface, HMI.

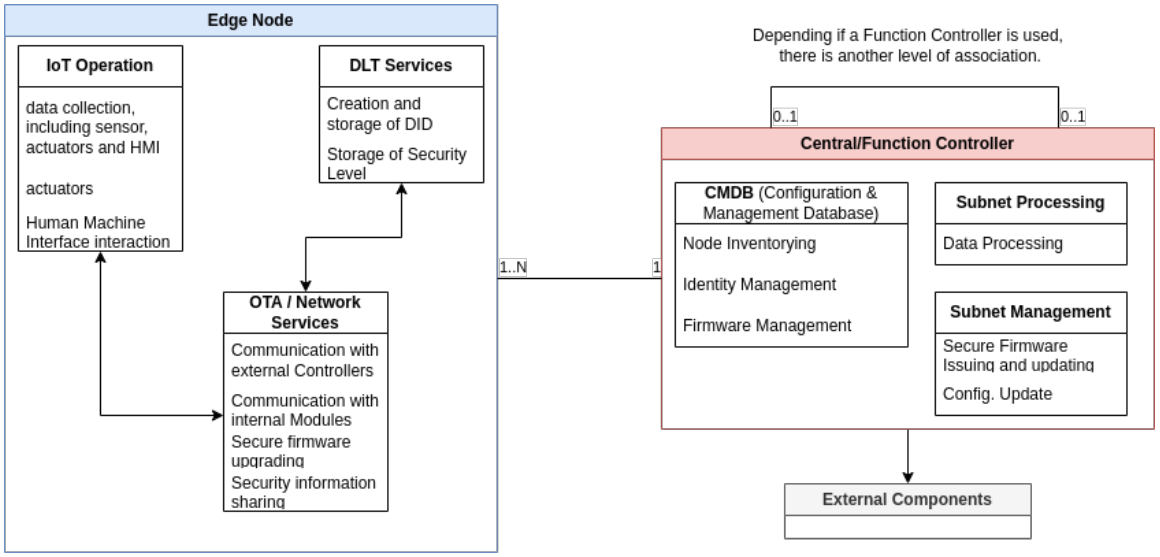

Figure 4.1: Device Architecture Overview

Each edge node is uniquely identifiable through a Decentralized Identifier, DID, which will later be used for device verification for outside entities through usage of a DLT. Furthermore, security levels, access levels will be stored on each device. Through the use of MUDs, each device will be defined to have certain capabilities and access privileges, which all will be enforced by the controller node.

Warning: Whether we will store our DID in a Secure Element, SE, or on the SoC FLASH, is not yet decided.

4.2.2 Controller Nodes

The controller nodes are managers of the edge nodes. They fulfill various responsibilities and shield edge nodes in a subnet for increased security control

4.3 Onboarding

The onboarding of a device can be observed in a sequence diagram in Figure 4.2. All of the actors in Section 4.1 are involved.

We assume a device has been manufactured and that it is now at the desired location. In the first time setup, the device will be either programmed to create a DID on setup, or will be assigned/created one in the manufacturing process, depending on the computational and hardware facilities. Both an edge and controller node shall have their DID, since both will be managed and or inventoried by another instance or entity. Necessary for this process is a valid MUD file, also issued by the manufacturer, on device fabrication or on first time setup. This MUD does not necessarily need to be on the device, but a URL is needed, which points to the location of such file, see Section 2.2. Each device shall request

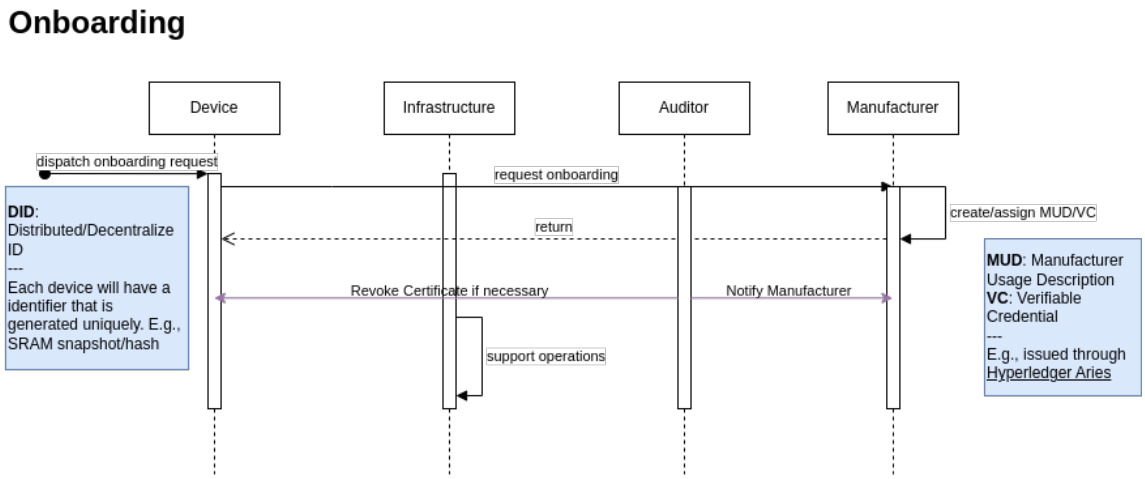

Figure 4.2: Onboarding Sequence Diagram

and onboarding by the manufacturer, which then verifies the DID, and in case of validity issues a Verifiable Credential, VC, to the device.

In more detail, an edge node shall request onboarding from the controller node, which then forwards the request to the manufacturer.

The Auditor then proceeds to verify the validity of the VC and checks for any known vulnerabilities, that might be affect the device that requested the VC. The Auditor shall also monitor the VC periodically, in order to check for any vulnerabilities the devices themselves might have missed. In case of an insecurity or vulnerability the auditor shall notify both the device and the manufacturer and revoke the VC, until the device has been known to update its Firmware to a secure version, see Section 4.6.

4.3.1 VC Verification

The verification of a VC may happen through the usage of a Smart Contract. Therein the invalidation of said smart contract happens through the fulfillment of the contract condition, which could be a discovered vulnerability.

4.4 Inventorying

Getting to the first emphasis of our thesis, we will explore, how we can safely and efficiently store and keep an inventory of our devices.

For this we will rely on a DLT, in which any action will be saved as a block. The DLT used shall be permissioned and possibly private, so that only those permissioned may be a part of this blockchain. The DLT will work as a decentralized Configuration and Management Database, CMDB, as seen in Figure 4.3. Transaction will be fulfilled by the controller node as indicated in Figure 4.1.

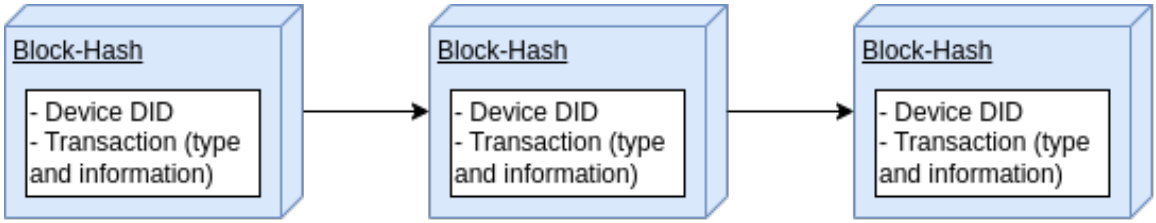

Figure 4.3: DLT Architecture

4.5 Operations

In order to be able to simulate the whole device lifecycle, we will also implement simple day-2-day operations, such as reading out sensors, actuating or minor interactions through HMIs.

Depending on the device type, edge or controller, further responsibilities are expected. A controller node has to constantly check against the DLT, whether there were any new vulnerabilities discovered and if the VC was consequently revoked, as mentioned before, the controller node has to check its own status, as well as the ones of the subnet of edge nodes. If such were the cases, concerned devices need be recertified.

4.6 Secure Firmware Updating

As the second emphasis of this thesis we aim to provide a secure update mechanism for outdated and or vulnerable firmware versions.

4.7 Secure Information Sharing

In order to enable the sharing of information related to the security of the device, we will rely on directional information sharing, meaning, the devies receive vulnerability information, updates and MUD files, while external sources receive status updates and co. of the devices concerned.

These information can be embedded into transaction inside the DLT, depending on the importance of the information, it can also be shared through less sophisticated measures.

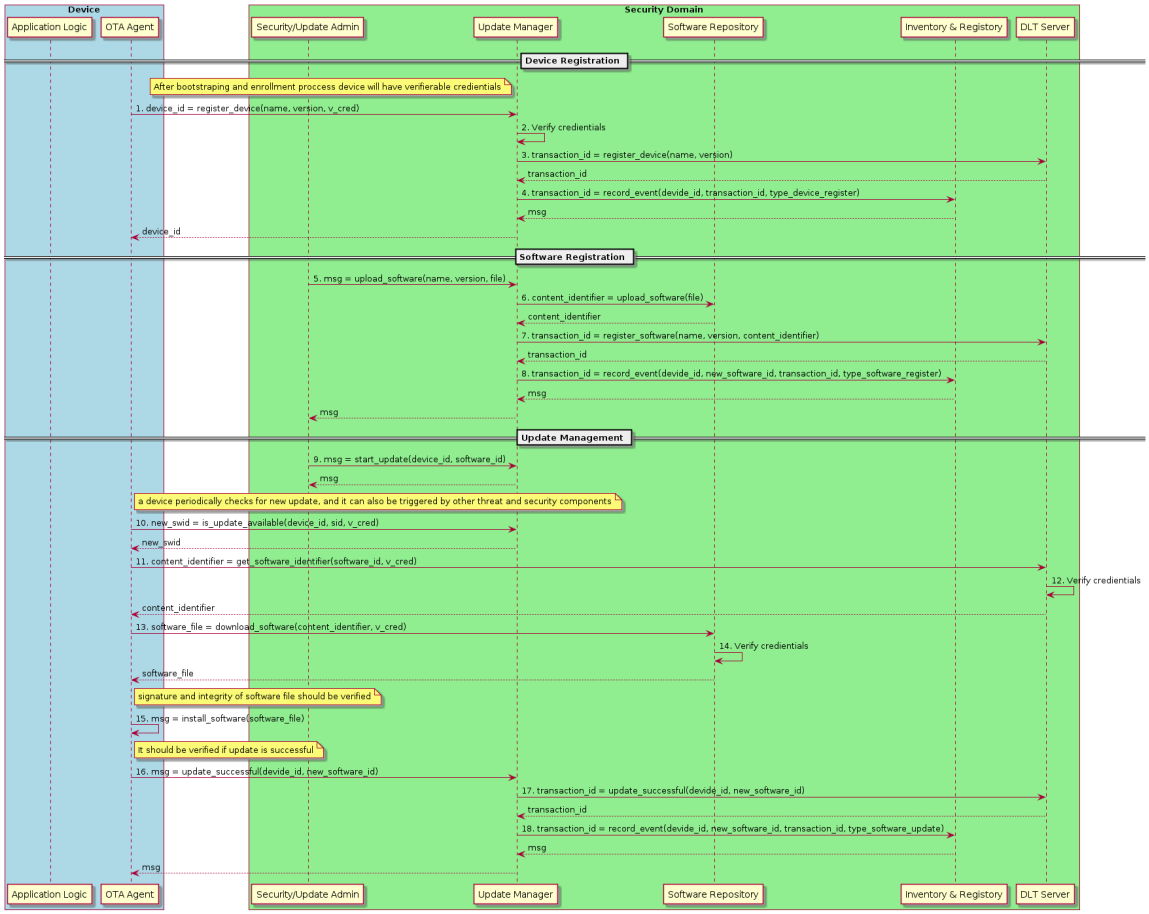

Figure 4.4: CERTIFY OTA Update [12]

Chapter 5

Use Case Definition - Connected Cabin System

5.1 Background

Our use case will take the CCS scenario from Figure 5.1 into consideration and build up on their use cases, which were borrowed from the CERTIFY project [12].

Nowadays more and more IoT devices are being deployed to aircraft cabins to improve passenger experience and airline operations. Benefits span from remote PHM to reduced maintenance time, while also supporting a continuous (re)certification process.

5.2 Actors

We will consider following actors for our use case.

- Airline: Owns the aircraft and oversees daily interactions and systems operations.
- Airplane maintainer: They could be e.g., the airplane manufacturer. Oversees maintenance of the aircraft, including the integration of systems designed by different manufacturers and their configuration.
- Product Owner: Oversees design and maintenance of systems deployed in the aircraft on assignment of the airplane maintainer.
- Maintenance operator: They work for the airplane maintainer. Their responsibilities include e.g., the replacement of devices or on-site software upgrades of e.g., portable data loaders.
- Passenger, Attendant, Pilot: They interact with the aircraft through sensors, actuators or HMI.

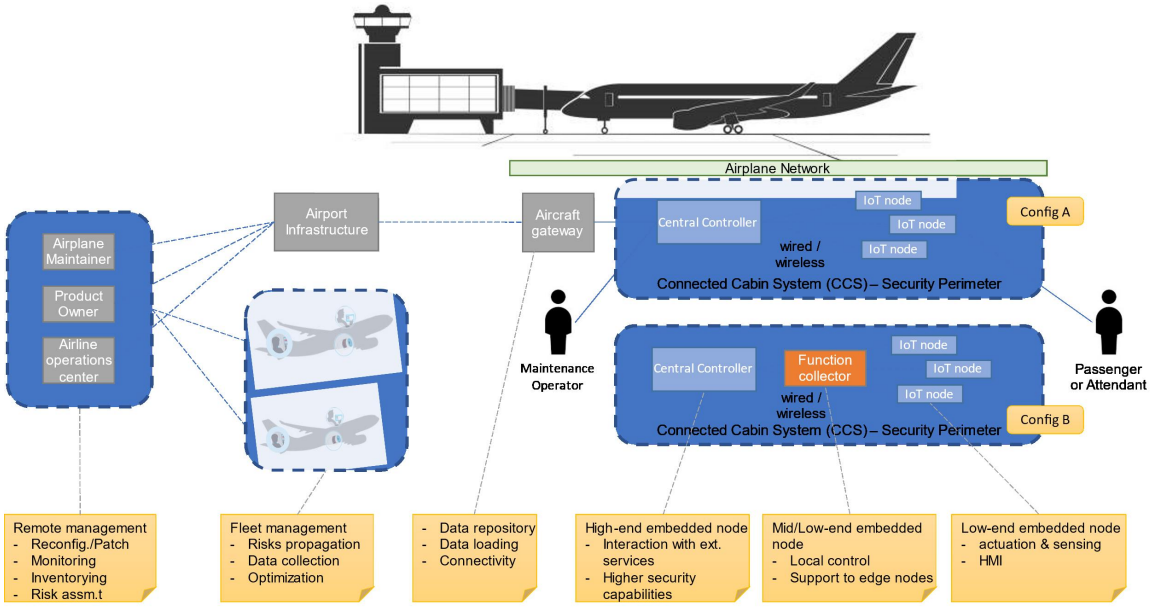

Figure 5.1: Collins CCS

5.3 System Components

We will consider an aircraft to have multiple networks, covering various aspects.

- In-flight entertainment system
- Aircraft System
- Flight Maintenance

For our use case we will assume config 'A' as the main configuration of the networks, where edge nodes are connected to a central controller that manages the edge nodes as a subnet.

- IoT / Edge Nodes: low-end devices, including actuation, sensing or HMI capabilities, with limited room for hardware and software based cybersecurity, that requires offloading to a more capable instance.
- Central Controller: High-end devices with ability to host full-fledged security functionalities.

External communication will take place through aircraft gateway offering services for data repository, data loading and connectivity with external environment. The airline operations center, product owner and airplane maintainer can interact through the airport infrastructure. A technician may directly access the aircraft if necessary.

Table 5.1: Actors involved

Airline	Airplane Maintainer	Product Owner	Maintenance Operator	Passenger, Attendant, Pilot
X	X	X	X	X

Table 5.2: Lifecycle stages involved

Bootstrapping	Operation	Update	Repurposing	Decommissioning
X	-	X	-	X

5.4 Scenarios

5.4.1 Installation of Connected Cabin Systems

5.4.1.1 Goals

The goals of this scenario include bootstrapping and customization of devices for specific deployment, updating and decommissioning of previous systems, guaranteeing a reset to a known and fresh, wiped data, state. Table 5.1 highlights the involved actors and Table 5.2 shows the stages involved in this scenario.

5.4.1.2 Pre-condition

In order for this scenario to be valid, following pre-conditions need to be met:

- Actors involved can establish a secure connection with the aircraft, wireless or wired, through airport infrastructure
- Airport and Aircraft network infrastructure can receive authorization requests for needed connections from the external environment.
- The Maintenance Operator is provided access to the airplane and to maintenance ports of the target CCS.

5.4.1.3 Sub-Scenario 1: Component Installation

Flow of Events The flow of events can be tracked in Figure 5.2 and is verbalized as follows:

- Airline requests installation of new component to the Airplane Maintainer also issuing an authorization request to Airport and Airplane gateways.

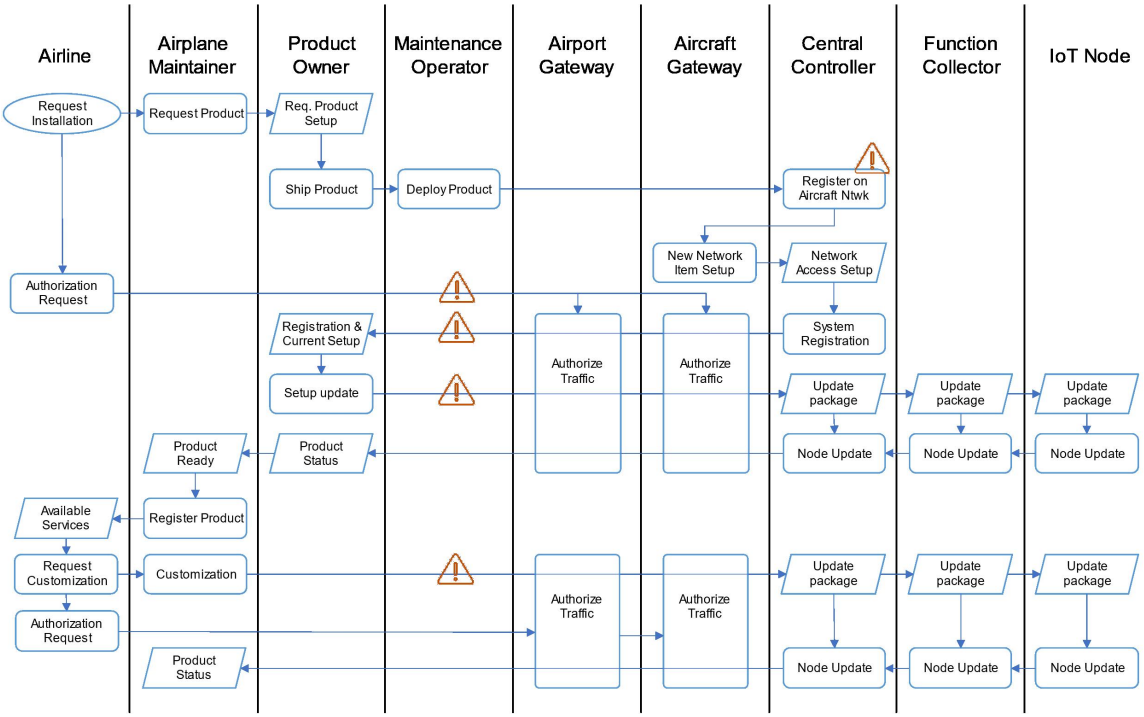

Figure 5.2: Collins Scenario 1: Device Installation

- The request is forwarded to the Product Owner and then to the Maintenance Operator, who oversees the physical deployment of the product.
- Once connected, Central Controller registers on the Aircraft Network and receives required setup to complete network access and system registration.
- Product Owner is now able to reach the CCS, push configuration and security updates to the Central Controller, as well as the Function Collector and IoT nodes.
- After the update, the product is registered and the Airplane Maintainer can offer remote services to the Airline.
- The Airline requests a customization of the CCS. It is performed by the Airplane Maintainer by pushing an update package and/or modifying specific configurations as allowed by the Product Owner API for Maintenance.
- The new product status is confirmed with a feedback message.

5.4.1.4 Sub-Scenario 2: Component Replacement

Flow of Events

- The Airline requests replacement of a component to the Airplane maintainer, also issuing an authorization request to the Airport and Airplane Gateways.

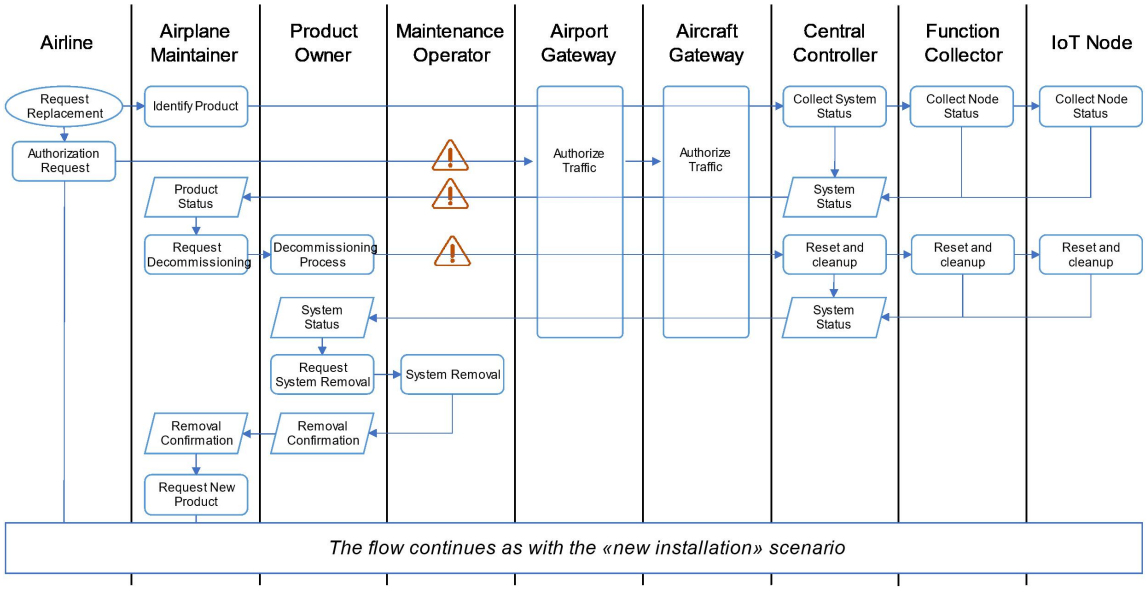

Figure 5.3: Collins Scenario 1: Device Replacement

- The Airplane Maintainer identifies the target product (location) and collects latest system status.
- The Airplane Maintainer issues a decommissioning request to the Product Owner and starts the decommissioning process, which causes a reset and cleanup of all the nodes that will be replaced.
- After remote reset and clean-up, product owner requests the Maintenance Operator to physically remove the system from the cabin.
- The product is then unregistered and can be dismissed.
- The remaining part continues with the 'New installation process flow'

5.4.1.5 Post-Condition

After completion of the Installation Scenario, following post-conditions need to be met:

- New component is deployed in the CCS, integrated into the network, updated with latest security patches and configured by the Airline for their specific needs.
- Component is securely onboarded in the CCS, unique identity and certificates are dispatched for authentication.
- Regarding the configurations, their integrity is verified and confidentiality has been preserved.

Table 5.3: Actors involved operations

Airline	Airplane Maintainer	Product Owner	Maintenance Operator	Passenger, Attendant, Pilot
X	X	X	-	X

Table 5.4: Lifecycle stages involved

Bootstrapping	Operation	Update	Repurposing	Decommissioning
-	X	X	-	-

5.4.1.6 Attack Scenario

As an alternative flow of events, i.e., in an attack scenario, highlighted by the yellow triangles in Figure 5.2 and Figure 5.3 following points were identified:

- Attacker can inject malicious payloads in place of the intended one, (confidential) credentials provided to the Central Controller for network access and authentication ca be stolen.
- IP sensitive data can be leaked by the Airplane Maintainer when retrieving the system status.
- The integrity of maintenance/reset/cleanup procedures can be compromised.

5.4.2 System Operation and Monitoring

5.4.2.1 Goals

The goals of System Operations and Monitoring incorporate following points:

- Periodic collection of data from airplane.
- Data offload/upload to ground stations for performance monitoring, optimization and PHM operations.
- Attendants interact with CCS through a HMI
- CCS Information is collected and stored in Gateway.
- A limited set of predefined reconfigurations may be performed on plane, when requested by Airline, Maintainer or product owner.

5.4.2.2 Pre-conditions

- Maintainers have established remote secure connection with the aircraft (wireless or wired) through the airport infrastructure.
- Passengers/Attendants/Pilots can interact through HMI or are connected through other devices e.g., through WiFi (possibly in the sense of bring you own device, BYOD).
- Device bootstrapping, enrollment, configuration, provisioning are completed for all devices, that are (statically) part of the network.
- IoT devices are equipped with sensors to collect and store data that are then forwarded to their root controller.
- Devices can securely store and transmit collected data.

5.4.2.3 Flow of Events

5.4 5.5 5.6

- IoT devices collect data during airplane operations, see Figure 5.4
- Data is securely stored on-board, see Figure 5.4
- Local computations over critical and non-critical data are executed in separate environments to reconfigure the airplane/flight, see Figure 5.4 **What is meant by this?**
- Remote entities are authenticated and a connection is established with the aircraft network through the gateway, see Figure 5.6 **What Gateway? We are basically in flight so shouldn't we be disconnected? Otherwise this is a scenario for Roaming**
- Data is downloaded from plane to ground
- In case of an Airplane fleet, Collective Analysis is performed, see Figure 5.5
- Upload new data and configurations, from ground to plane
- Data authenticity and integrity are verified before updating the configurations on the plane.

5.4.2.4 Post-Conditions

After any of the sub-scenarios from Figures 5.4, 5.5, 5.6, the following post-conditioned need to be met:

- A new configuration, computed either locally or through the remote connection with the operations center is available.

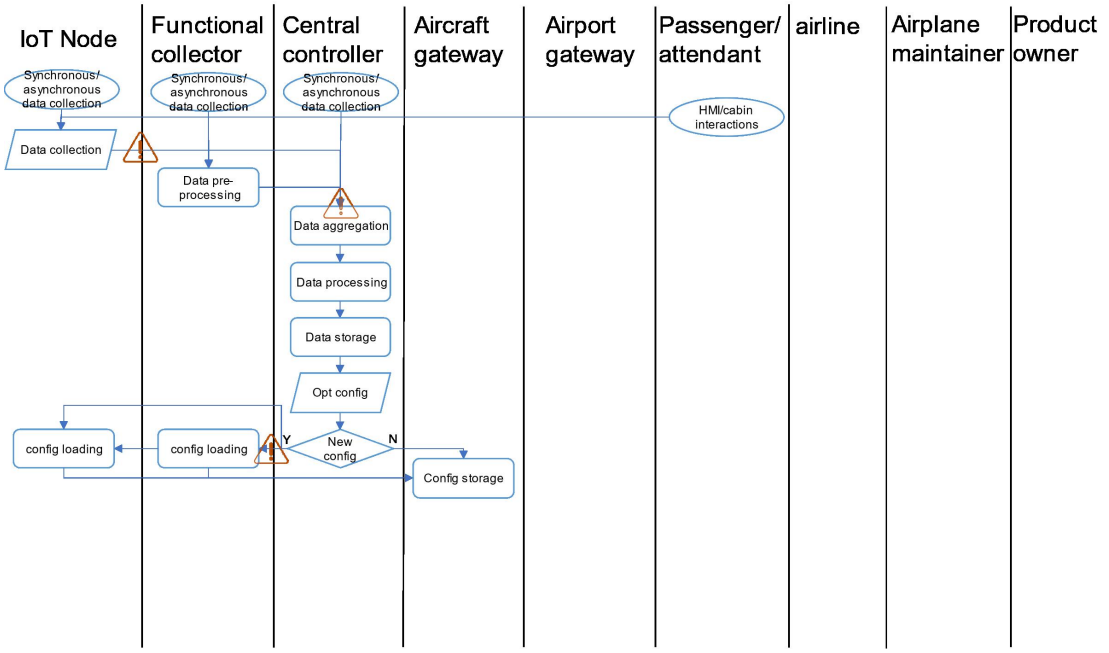

Figure 5.4: Collins Scenario 2.1: Data Collection and Local Reconfiguration

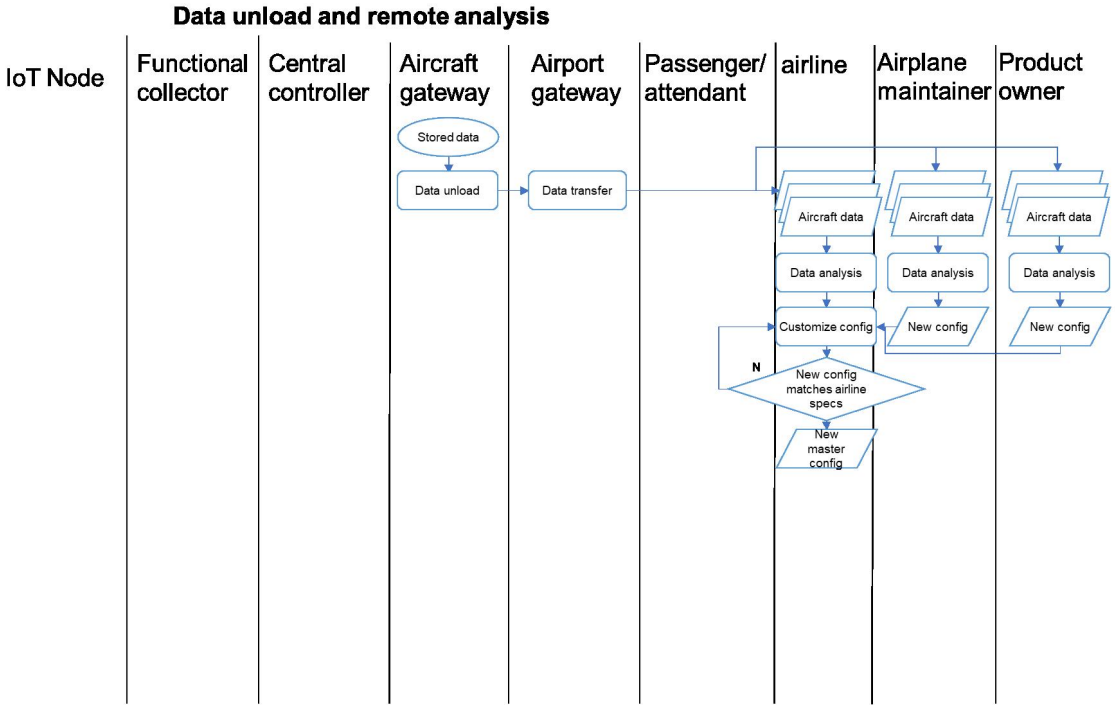

Figure 5.5: Collins Scenario 2.2: Data Unload and Remote Analysis

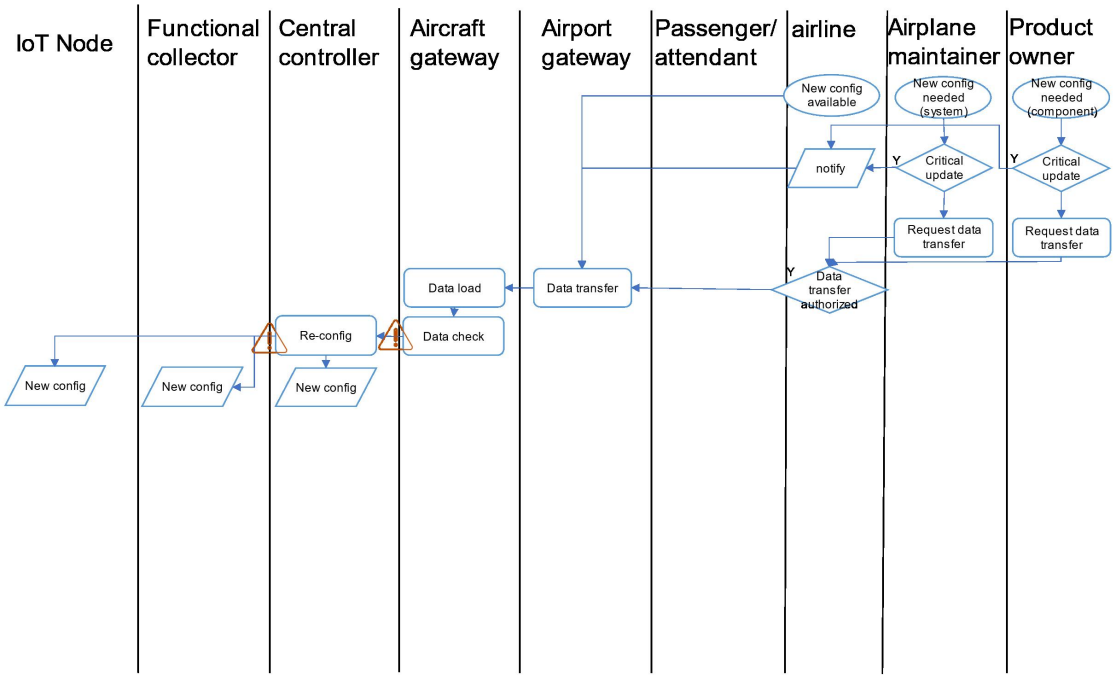

Figure 5.6: Collins Scenario 2.3: Data Load and Remote Reconfiguration

- Data from the aircraft are available for further analysis of Airline, Maintainer or Product Owner.
- For the configurations, integrity is verified, and confidentiality has been preserved (as it could involve IP issues) in the process.
- For the data, in addition to integrity and confidentiality, it is important to also ensure availability (to detect early sings of potential problems).

5.4.2.5 Attack Scenario

As an alternative flow of events, where an attack might happen, the main entry point is through the WiFi access point. A rogue device could inject false data, configurations or software to facilitate subsequent attacks or even cause system unavailability or device malfunctioning. Alternatively in stealth mode, sensitive information could be captures and could be used to perpetrate other attacks.

5.4.3 LRU Replacement and Repurposing

Akin to Scenario 1 in Subsection 5.4.1 in this Scenario we also consider similar steps to replace existing devices, with the difference, that the devices that replace the broken devices were not exactly meant for this situation.

Table 5.5: Actors involved

Airline	Airplane Maintainer	Product Owner	Maintenance Operator	Passenger, Attendant, Pilot
X	X	X	X	-

Table 5.6: Lifecycle stages involved

Bootstrapping	Operation	Update	Repurposing	Decommissioning
-	-	X	X	X

5.4.3.1 Goals

The goal of this scenario is to quickly replace a malfunctioning Central or Functional Controller, but a LRU is not directly available. To minimize downtime, a spare LRU is retrieved from the same manufacturer and repurposed to the specific target system. Airline, Airplane Maintainer, Product Owner, and Maintenance Operator are all involved to take care of different steps in the process.

5.4.3.2 Pre-Conditions

Following pre-conditions must be met:

- Actors involved can establish remote secure connection with aircraft.
- Airport has a spare LRU, that is compatible with CCS.
- The Maintenance Operator is provided access to the airplane and to the maintenance ports.

5.4.3.3 Flow of Events

- The Airline requests to the Airplane Maintainer the repair of a cabin system, issuing and authorization request to the Airport/Airplane gateways for following remote software update operations.
- The Airplane Maintainer identifies the target product, including its location, and the failure condition then requests repair to Product Owner.
- The Product Owner starts the removal process of the LRU, including reset and cleanup
- Failed LRU removal executed locally by Maintenance Operator (also in charge of reset and cleanup)

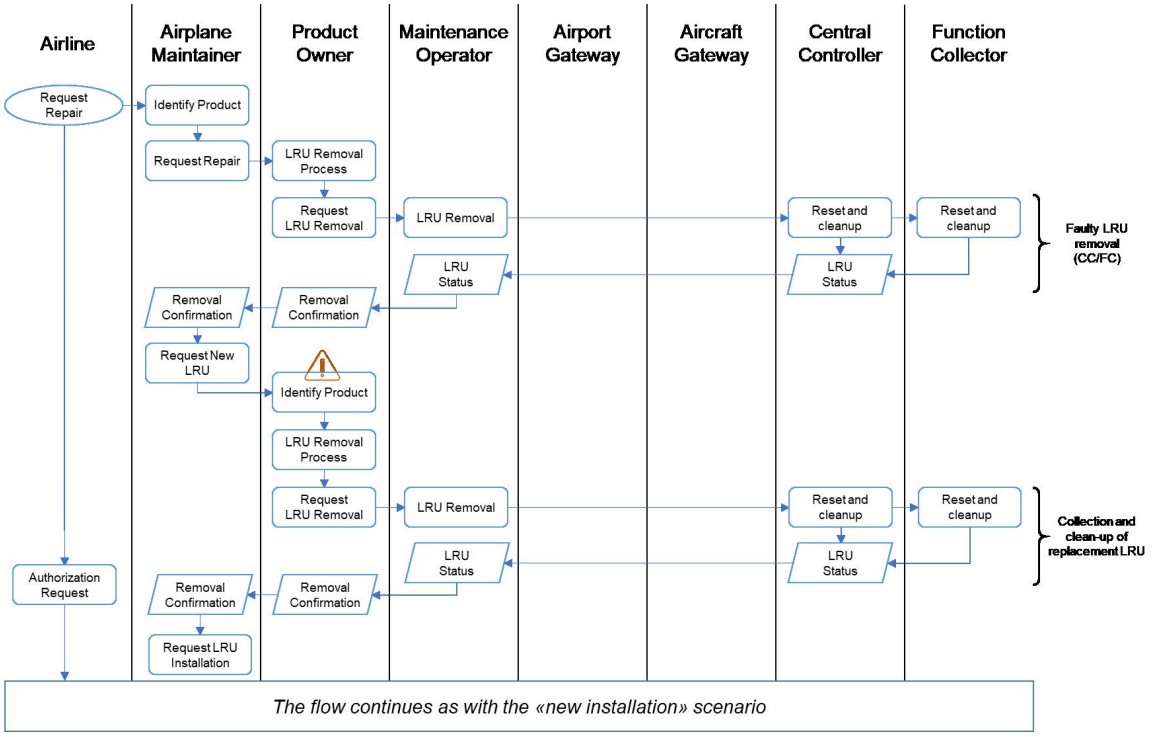

Figure 5.7: Collins Scenario 3: LRU Replacement

- Product owner informs Airplane Maintainer of the removal and receives information of available replacement LRUs.
- The replacement LRU shall be available from a remote location.
- The remaining flow proceeds as in Scenario 1 in Subsection 5.4.1

5.4.3.4 Post-Conditions

- A new LRU is deployed, integrated into network, updated with latest security patches and configured by the Airline for specific needs
- CCS is registered with unique identifier and certificates are dispatched for authentication.
- For the configuration, integrity is verified and confidentiality has been preserved (for IP protection) in the process

5.4.3.5 Attack Scenario

Attacks follow same pattern as in Subsection 5.4.1 from Scenario 1. An attacker can inject malicious software or a counterfeit LRU through the supply chain.

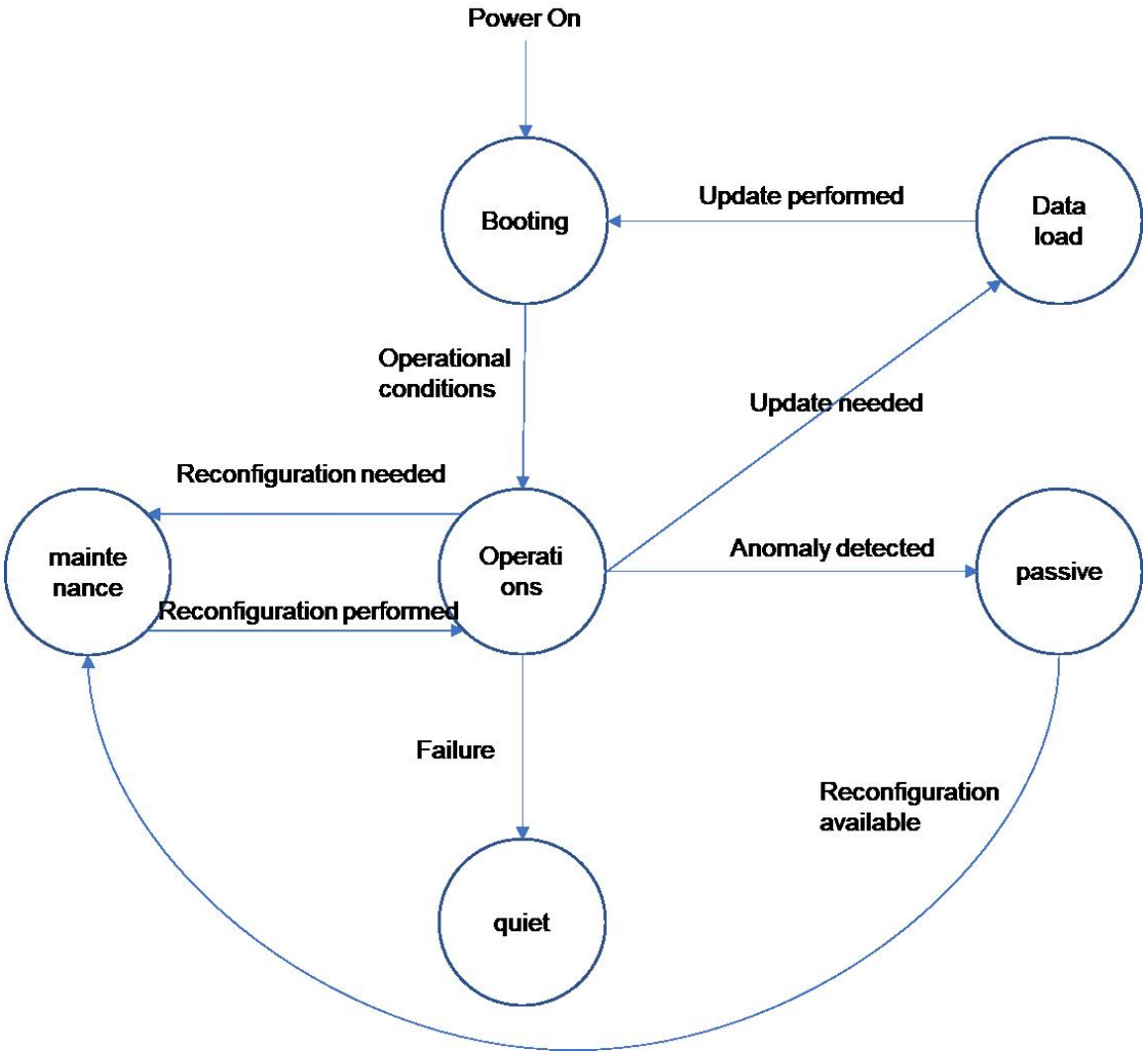

Figure 5.8: Collins Operation Modes for an On-Board Device

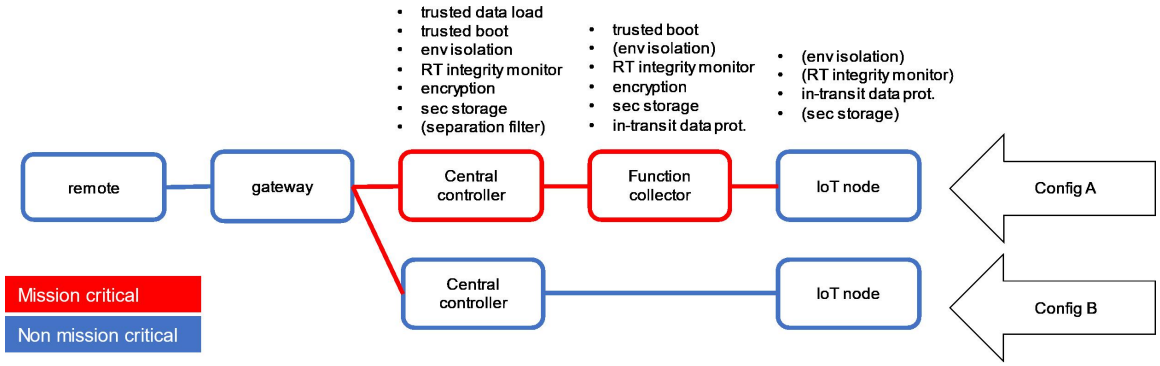

Figure 5.9: Collins Desired Security Features for the Different CCS Components

5.5 Cybersecurity Assessment

The CERTIFY [12] project describes following security levels:

- Basic: Focus on common and simple cyber threats. User authentication, access controls and basic security configuration.
- Substantial: More robust security measures including intrusion detection systems, IDS, incident response plans and periodic security assessment.
- High: Requires organizations to establish comprehensive and proactive cybersecurity programs. Further advanced security measures, network segmentation, encryption, continuous monitoring and regular vulnerability assessments are needed. Threat intelligence sharing and regular security audits are also in order.

Our use case, CCS, falls into the category of level *HIGH*, as from a cybersecurity viewpoint the aircraft cabin is considered a highly volatile and highly targeted environment.

Chapter 6

Evaluation

Chapter 7

Summary and Conclusions

Bibliography

[1] "The eu cybersecurity act." [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/cybersecurity-act 1

[2] "Directive on measures for a high common level of cybersecurity across the union (nis2 directive)." [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/nis2-directive 1

[3] R. Neisse, G. Steri, and I. Nai-Fovino, "A blockchain-based approach for data accountability and provenance tracking," in *Proceedings of the 12th international conference on availability, reliability and security*, 2017, pp. 1–10. 2, 7

[4] (2023) Trusted iot device network-layer onboarding and lifecycle management. [Online]. Available: https://www.nccoe.nist.gov/projects/trusted-iot-device-network-layer-onboarding-and-lifecycle-management 2

[5] E. J. Scheid, T. Hegnauer, B. Rodrigues, and B. Stiller, "Bifröst: a modular blockchain interoperability api," in *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. IEEE, 2019, pp. 332–339. 2

[6] V. A. Siris, P. Nikander, S. Voulgaris, N. Fotiou, D. Lagutin, and G. C. Polyzos, "Interledger approaches," *Ieee Access*, vol. 7, pp. 89 948–89 966, 2019. 2

[7] D. Dodson, D. Montgomery, W. Polk, M. Ranganathan, M. Souppaya, S. Johnson, A. Kadam, C. Pratt, D. Thakore, M. Walker *et al.*, "Securing small-business and home internet of things (iot) devices: Mitigating network-based attacks using manufacturer usage description (mud)," National Institute of Standards and Technology, Tech. Rep., 2021. 2, 3, 7, 8

[8] E. Lear, R. Droms, and D. Romascanu, "Manufacturer Usage Description Specification," RFC 8520, Mar. 2019. [Online]. Available: https://www.rfc-editor.org/info/rfc8520 3

[9] C. Herder, M.-D. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1126–1141, 2014. 4

[10] S. Vinagrero, H. Martin, A. de Bignicourt, E.-I. Vatajelu, and G. Di Natale, "Sram-based puf readouts," *Scientific Data*, vol. 10, no. 1, p. 333, 2023. 4, 5, 7

[11] D. E. Holcomb, W. P. Burleson, K. Fu *et al.*, "Initial sram state as a fingerprint and source of true random numbers for rfid tags," in *Proceedings of the Conference on RFID Security*, vol. 7, no. 2, 2007, p. 01. 5

[12] (2023) active security for connected devices lifecycles. [Online]. Available: https://certify-project.eu/ 7, 13, 15, 27, 39

[13] E. Lear and V. Andalibi. Example mud json file. [Online]. Available: https://www.mudmaker.org/examples.html 43, 49

Abbreviations

AAA	Authentication, Authorization, and Accounting
ACL	Access Control List
CCS	Connected Cabin System
CRP	Challenge-Response Pair
CTIS	Cyber Threat Information Sharing
EVM	Ethereum Virtual Machine
GDPR	General Data Protection Regulation
HMI	Human Machine Interface
IDS	Intrusion Detection System
IETF	International Engineering Task Force
IFE	In-flight Entertainment System
IPS	Intrusion Prevention System
IoT	Internet of Things
LRU	Line Replacable Unit
MUD	Manufacturer Usage Description
NIST	National Institute of Standards and Technology
PHM	Prognostics and Health Management
PUF	Physically Unclonable Function
SCADA	Supervisory control and data acquisition
SRAM	Static Random-Access Memory
TEE	Trusted Execution Environment
TOE	Target of Evaluation
VC	Verifiable Credential

Glossary

Trust Model In the trust model the issuer issues credential to a holder while the holder can prove identity by showing the credential to a verifier.

Manufacturer Usage Description A component-based architecture specified in Request for Comments (RFC) 8520 that is designed to provide a means for end devices to signal to the network what sort of access and network functionality they require to properly function.

Cloud Computing Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user.

Fog Computing As an extension of Cloud computing, Fog Computing brings the computation closer to IoT Edge devices.

Edge Computing Edge computing is the placement of storage and computing resources closer to source, where the data is generated.

Trusted Execution Zone

Line-Replaceable Unit modular component of airplane, designed to be replaced quickly

Firmware Computer programs and data stored in hardware - typically in read-only memory (ROM) or programmable read-only memory (PROM) - such that the programs and data cannot be dynamically written or modified during execution of the programs.

List of Figures

List of Tables

List of Code Snippets and Examples

Appendix A

Installation Guidelines

Appendix B

Code Snippets and Examples

```
{
  "ietf-mud:mud": {
    "mud-version": 1,
    "mud-url": "https://iot-device.example.com/dnsname",
    "last-update": "2019-01-15T10:22:47+00:00",
    "cache-validity": 48,
    "is-supported": true,
    "systeminfo": "This is an example of a device that just wants to talk
                   to its cloud service",
    "mfg-name": "Example, Inc.",
    "documentation": "https://iot-device.example.com/doc/dnsname",
    "model-name": "dnsname",
    "from-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-96898-v4fr"
          },
          {
            "name": "mud-96898-v6fr"
          }
        ]
      }
    },
    "to-device-policy": {
      "access-lists": {
        "access-list": [
          {
            "name": "mud-96898-v4to"
          },
          {
            "name": "mud-96898-v6to"
```

```
        }
      ]
    }
  }
},
"ietf-access-control-list:acls": {
  "acl": [
    {
      "name": "mud-96898-v4to",
      "type": "ipv4-acl-type",
      "aces": {
        "ace": [
          {
            "name": "cl0-todev",
            "matches": {
              "ipv4": {
                "ietf-acldns:src-dnsname": "cloud-service.example.com"
              }
            },
            "actions": {
              "forwarding": "accept"
            }
          }
        ]
      }
    },
    {
      "name": "mud-96898-v4fr",
      "type": "ipv4-acl-type",
      "aces": {
        "ace": [
          {
            "name": "cl0-frdev",
            "matches": {
              "ipv4": {
                "ietf-acldns:dst-dnsname": "cloud-service.example.com"
              }
            },
            "actions": {
              "forwarding": "accept"
            }
          }
        ]
      }
    },
    {
      "name": "mud-96898-v6to",
```

```
      "type": "ipv6-acl-type",
      "aces": {
        "ace": [
          {
            "name": "cl0-todev",
            "matches": {
              "ipv6": {
                "ietf-acldns:src-dnsname": "cloud-service.example.com"
              }
            },
            "actions": {
              "forwarding": "accept"
            }
          }
        ]
      }
    },
    {
      "name": "mud-96898-v6fr",
      "type": "ipv6-acl-type",
      "aces": {
        "ace": [
          {
            "name": "cl0-frdev",
            "matches": {
              "ipv6": {
                "ietf-acldns:dst-dnsname": "cloud-service.example.com"
              }
            },
            "actions": {
              "forwarding": "accept"
            }
          }
        ]
      }
    }
  ]
 }
}
```

Code 1: Example MUD file [13]