Security Analysis of a Blockchain Based Data Collection Method for Cross Company Information Sharing

Tobias Bux(✉), Oliver Riedel, and Armin Lechler

Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart, Seidenstr. 36, 70174 Stuttgart, Germany
tobias.bux@isw.uni-stuttgart.de

Abstract. Digitization within medium-sized enterprises advanced in the last years. Collecting and analyzing data for optimizing internal production processes therefor is the current state of many companies. The next step of digitization is using this collected data not only for internal processes but for cross company business models along the value network. This step brings new requirements for how data is collected, stored and shared. In this paper those requirements are listed and explained. Afterwards, an implemented solution for data collection fulfilling the requirements is analyzed. The focus of the analysis lies on security issues within the data flow between data creation and cross-company usage. Therefore, the timespan between data creation on a sensor, processing the data within local IT-systems and reliably storing data within a blockchain is considered. A threat modeling approach considering attack vectors along the described data flow is used to quantitatively compare the proposed solution to regular industrial solutions. The analysis will highlight the differences of the compared solutions on different topics like data integrity and immutability. Lastly, an outlook on industrial usage of the analyzed solution is given.

Keywords: Blockchain · Data Integrity · Cross-Company Data Exchange

1 Introduction

Several IT systems are usually involved in the collection and processing of industrial data. These systems are located in differently secured networks, have different operating systems and each use their own software. It therefore becomes more difficult to make statements about data security in industrial data acquisition the more systems are involved. For in-house data usage, the problem is limited because the standard data acquisition process, shown in Fig. 1, is usually covered by the security policies of the company's IT department [1]. The requirements for internal company use of collected data are usually met here. However, if external partners within the value network are also involved in the processing of the data, the requirements change. In addition to data security, data integrity plays a role in this use case because there is no inherent trust between the companies. This means that when data is exchanged, not only must security be guaranteed, but it must also be ensured that the data to be exchanged is unchanged and legitimate. Accordingly, the threat analysis in this paper does not focus exclusively on data and transport security, but also on the risks for preserving integrity.

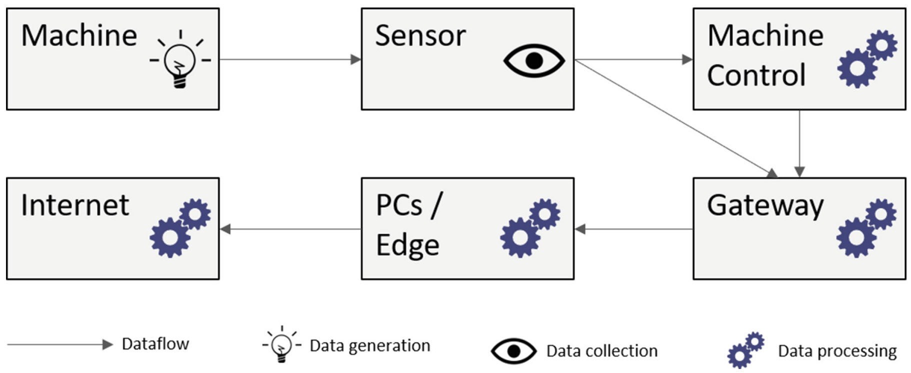

Fig. 1. Common Data Flow within a production environment [1]

2 Related Work

Before discussing the results of this work, the state of the art is considered in two basic areas. Firstly, the solutions that currently exist for cross-company data exchange and how their implementation is structured for security and integrity will be presented. Then, threat modeling techniques for industrial data exchange are examined.

2.1 Cross Company Data Exchange

A cross-company data exchange has requirements for the data to be exchanged that do not exist in an internal company environment. This is the conclusion reached by Uygun [2], who has therefore defined these requirements in more detail. In addition to structural and organizational requirements, which are not considered in this paper, he has established technical requirements. These include data security, communication security and consistency of data. A multi-agent tool was developed for validation, which was used by over 40 companies to verify the statements of Uygun. However, a model-based analysis of the technical requirements was not performed.

Ruf et al. [3] did exactly this. They looked at a framework for industrial data exchange called KOSMoS [4] and analyzed its security and integrity using threat model analysis. The considered framework uses a blockchain-based data exchange to guarantee the integrity of data. However, the results of Ruf et al. show that even in this environment, risks such as faulty smart contracts or incorrect interface usage must be considered [3]. In addition, communication between the data origin and the blockchain needs to be critically considered. The threat analysis by Ruf et al. mentions the risks of man-in-the-middle, identity spoofing and inside attacker [3].

The risks described by Ruf et al., were considered in more detail in a paper by Korb et al. [1] and a solution was proposed. The basic idea behind Korb et al.'s solution is to create and sign blockchain transactions close to the sensor. This way, data can be verified before it arrives in a Blockchain. The entry of false data, through manipulation or spoofing, into a blockchain is thus prevented. The implementation of Korb et al. is realized by using an ESP32 microcontroller that is directly linked to the data generating sensor via GPIO connection. The data is signed on the microcontroller before it passes through larger technical systems of internal IT. Although this solution offers increased security, the use of the Ethereum blockchain is not optimal for an industrial deployment, as quantitatively demonstrated by Polge et al. [5].

In summary, it can be stated that new requirements must be observed in the case of cross-company data exchange. There are already solutions for meeting the technical requirements, but they are not suitable for industrial use. Bux et al. have therefore developed a new technical solution based on Hyperledger Fabric [6], which is used as the basis for the threat analysis in this paper.

2.2 Threat Modelling for Production

There are several ways to analyze and assess the threats in an IT system. Shevchenko et al. [7] compared twelve currently used models in a report for the Defense Technical Information Center. As a result of their work, features were assigned to each model to help select an appropriate model. An overview of the four most common models can be seen in Table 1. According to Shevchenko, STRIDE [8] is the most widely used and mature model. In the described work by Ruf et al. [3], they also performed a STRIDE based risk analysis. Since STRIDE is the leading threat model, software tools were created for STRIDE to assist in performing the analysis. As STRIDE was invented by Microsoft, they offer software built around STRIDE called Thread Modelling Tool [9] which can be used within Microsoft Azure. However, besides paid software, there are also free alternatives. Threat Dragon, for example, is an open-source tool under Apache 2.0 license that is constantly maintained [10] and implemented STRIDE functionality.

To ensure the most representative analysis of the architecture evaluated in this thesis, STRIDE is used as a model for risk analysis. Threat Dragon is used to perform the analysis and graphically record the results.

Table 1. Threat modelling methods features based on Shevchenko et al. [8]

Threat Modeling Method	Features
STRIDE	• Helps identify mitigating techniques • Most mature • Easy to use but time consuming
PASTA	• Contributes to risk management • Rich documentation • Stakeholder collaboration intended • […]
LINDDUN	• Helps identify mitigating techniques • Built in prioritization of threat mitigation
CVSS	• Consistent results when repeated • Automated components • Not transparent score calculations

3 Proposed Solution Architecture

To understand the threat analysis, it is first necessary to present an overview of the software system under consideration. For this purpose, a UML-based architecture is presented in Fig. 2. A detailed description of the solution, an explanation of individual components, and the exact data flow within the solution are explained in detail by Bux et al. in [6]. The flow of this architecture is explained below:

Four hardware components exist in the architecture under study. A sensor (1) is seen as a data generation component and is directly linked via GPIO to the second hardware component, a microcontroller (2). On the microcontroller C++ code is executed, which is used for an information exchange with an industrial PC (3). Within this IPC, different services of the Hyperledger Fabric architecture communicate. A single service, called Client (Proxy) in Fig. 2, forms the interface between the local Hyperledger Fabric services and the microcontroller logic. This structure ensures that data collected by the sensor is converted into a Hyperledger Fabric compliant transaction under the supervision of the microcontroller. The transaction is signed on the microcontroller and then sent by the Committing Peer to a cloud instance (4) running a Hyperledger Fabric node. The data is stored there and can then be queried via REST interface.

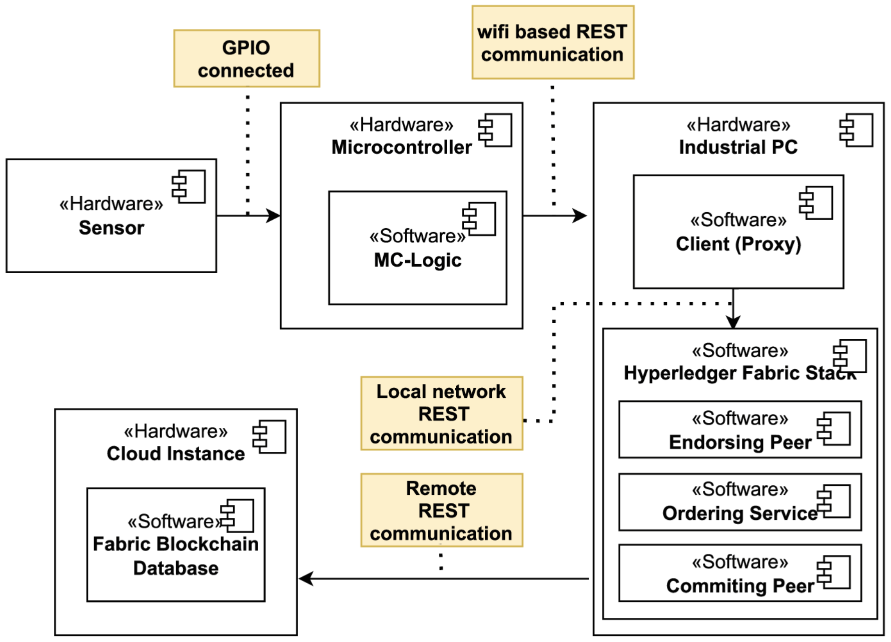
GPIO connected
wifi based REST communication
«Hardware» Sensor
«Hardware» Microcontroller
«Software» MC-Logic
«Hardware» Industrial PC
«Software» Client (Proxy)
«Software» Hyperledger Fabric Stack
«Software» Endorsing Peer
«Software» Ordering Service
«Software» Commiting Peer
Local network REST communication
Remote REST communication
«Hardware» Cloud Instance
«Software» Fabric Blockchain Database

Fig. 2. UML overview on the solution that is analyzed within this paper [6]

4 System Analysis

At the beginning of the analysis, some assumptions were made about the system state. The sensor, which provides data, functions without restrictions. According to its definition, the blockchain is an immutable storage. Furthermore, it is assumed that no faulty functionality is contained in the third-party services used, such as the software development kit for Hyperledger Fabric. With these assumptions, a STRIDE analysis was performed of the system described above. A visual overview of the results is displayed in Fig. 3. Meanings of the different colors, shapes and connectors are according to STRIDE standards.

During the analysis, different threats have been found. These are summarized and explained in the following:

Information Disclosure: The data flow between the components in Fig. 3 contains confidential data. All communication is therefore encrypted. The communication between sensor and microcontroller is an exception since the transmission of digital or analog sensor data is realized directly via GPIO. This security gap can be solved by sealing the sensor and the microcontroller into a Blackbox.

Denial of Service (DoS): In the architecture under consideration, several services exist that are vulnerable to a DoS attack. The largest hardware component is the microcontroller. If this component no longer functions, the entire system is disabled. This is

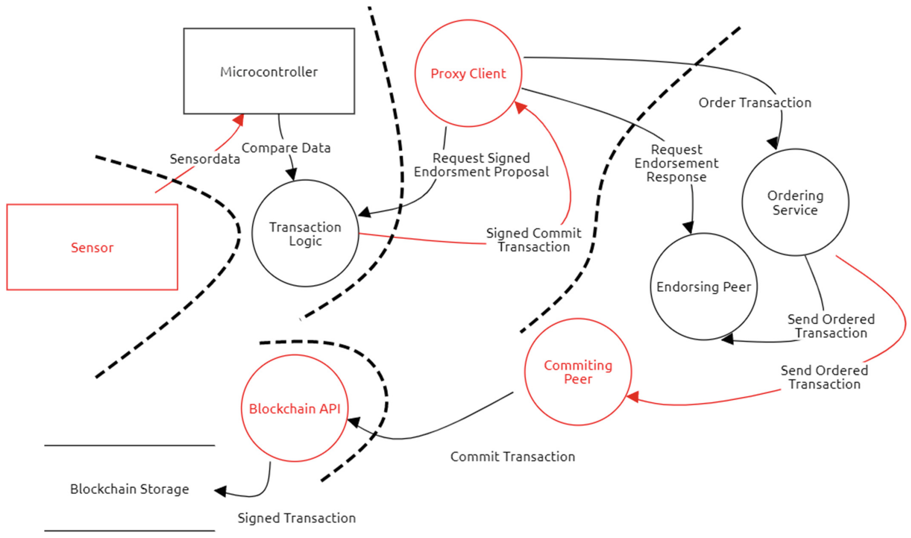

Fig. 3. Threats to using the proposed architecture for blockchain based data storage

possible, for example, by exhausting the CPU or completely filling the internal memory. However, since no write access to the microcontroller is allowed in the proposed solution, this threat is not considered further.

The proxy client, however, is much more susceptible to a DoS attack. It runs on an IPC with a conventional operating system and has a connection to the Internet. Accordingly, it depends on the security of the respective company IT whether the proxy client is sufficiently protected. Redundant systems and life-cycle management can minimize a service outage due to DoS attacks in this case.

The last component to be protected against a DoS attack is the blockchain interface. For this component, too, protection depends heavily on the respective implementation. It is worth mentioning that a temporary DoS attack on the blockchain interface has no impact, as the signed transactions are kept locally until they arrive in the blockchain storage.

Tampering: Using a signed blockchain transaction is a direct measure against tampering. Nevertheless, this transaction must first be created and signed. Here, Hyperledger Fabric has a disadvantage compared to other blockchains, such as Ethereum. Several components are involved in the signing process. These components cannot all be executed on one microcontroller. This architecture makes Hyperledger Fabric susceptible to tampering. For this reason, the protocol for signing transactions in this solution has been adapted. Each step of the protocol, described in more detail in [6], is checked by the microcontroller. In addition, the final signing process is performed directly on the microcontroller. This adaptation results in the signing process being tamper proof. However, this does not apply to the connection between the microcontroller and the sensor. The already proposed blackboxing solution is the only approach for this, where no adjustments must be made on the sensor itself.

Escalation of Privilege: The use of a microcontroller-based solution, which is as close to the hardware as possible, ensures difficult access and modification of execution logic. However, this does not apply to the IPC on which the second part of the architecture is executed. For example, to prevent an inside attack by an employee who has gained access to the IPC, a user management and access control system is necessary. Similar to DoS prevention, threat assessment depends on the internal IT systems.

5 Findings

In the following, the threats found are assigned to different systems. The systems are technical (Infrastructure), person-related (Employee) and business model (Business) related. Each threat is classified systematically. For this purpose, the impact is defined on the one hand and the risk of the threat occurring is assessed on the other. Impact can be categorized into critical (C), medium (M) and low (L). Risk is expressed in high (H), medium (M) and low (L). In addition, common methods are listed that can be used for threat mitigation. Table 2 displays an overview of the risk assessment results.

Table 2. Risk Assessment based on Exposures

Exposure	*Threat*	*Impact*	*Risk*	*Mitigation*
Infrastructure	Malware	M	M	Malware detection
	DoS	C	M	Design, Firewall
	Information Disclosure	C	L	Encryption
Employee	Misconfiguration	C	L	Audits, Tests
	Inside Attack	C	L	Logs, Privilege management
Business	Tampering	M	L	Sealing, Transaction signature
	Man in the middle	L	L	Signature, Data buffering
	Data manipulation	C	L	Anomaly detection

5.1 Infrastructure

Malware is an often-used technique to corrupt processes or data storages. In this architecture, it is mainly the IPC that is vulnerable, since the microcontroller without a proprietary operating system is a difficult target for malware. If malware were to influence the Hyperledger Fabric components of the architecture, the integrity of the subsequent blockchain could potentially no longer be guaranteed. An up-to-date operating system and malware detection provide the necessary protection.

Denial of Service (DoS) attacks, as the name suggests, are mainly used to prevent systems from working. Exposed interfaces are usually used for this purpose. In the case of this architecture, such an interface is provided for the interaction with a blockchain. However, since the internal service only addresses a REST interface and cannot be operated itself, the risk of a DoS attack is reduced. Redundancy of systems and an up-to-date firewall can protect against this risk.

Information Disclosure is a serious threat once unencrypted data is accessible. Permanent encryption and a user-based access system is sufficient in most cases to prevent this risk.

5.2 Employee

Misconfiguration is a common error as soon as people are involved in processes. In the case of this architecture, it is necessary to configure both individual services and the communication between them. It is therefore necessary to take precautions. Unit and integration tests should be performed before rolling out the system. Finally, an audit of the running system helps to prevent incorrect configuration.

Inside attacks are very difficult to prevent. Despite rights management, access to data and services is necessary for selected personnel. The only effective protection is to know your personnel and to recognize changes in behavior. Logging systems help to detect inside attacks. Like surveillance cameras, logging systems can have a disabling effect on offenses.

5.3 Business

Avoiding **tampering** within this architecture is necessary mainly between the sensor technology and the microcontroller. Even if subsequent attacks are possible, the only effect is that individual data records are not saved or are saved late. However, the integrity of the stored data is not violated, due to the system sorting out malicious data. Meaningful protection between sensor and microcontroller can only be achieved by locking and sealing the two components.

Man in the middle attacks are not a major threat to this architecture. All data is encrypted and therefore cannot be viewed. In addition, the signature of each recipient is checked, which means that messages that have been smuggled in or changed are not accepted. In addition, local buffering of data ensures that data is not lost if it does not end up in the blockchain.

Sustained **data manipulation** is no longer possible after sensor data has entered the microcontroller. This means that either false data must be injected between sensor and microcontroller or the sensor itself must be manipulated. Both can be prevented by sealing the two components.

6 Summary

In this paper, a Hyperledger Fabric-based approach to data provisioning was tested for security and immutability. For this purpose, it was shown which gaps the proposed architecture closes. Then, based on state-of-the-art techniques, an analysis of remaining risks was performed, classified and explained.

In conclusion, it can be said that even the proposed architecture cannot exclude all threats, but the risk of occurrence is significantly lower than with commercially available solutions. The hardware-based signature of the blockchain transactions prevents manipulation of data within the downstream systems and thus offers a clear advantage over the rest of the solutions.

Nevertheless, a security gap of this approach has become clearly visible. The connection of sensor and microcontroller is the weak point of the system. However, without developing sensor specifics or having to customize each sensor, there are only physical ways to improve the connection of the two components. Using the technique of sealing both components in a Blackbox, an industrial usage is not yet given. By not using this technique, the system can be used with every sensor that can deliver its data via GPIO.

References

1. Korb, T., Michel, D., Riedel, O., Lechler, A.: Securing the data flow for blockchain technology in a production environment. IFAC-PapersOnLine **52**(10), 125–130 (2019). ISSN: 2405-8963. https://doi.org/10.1016/j.ifacol.2019.10.012
2. Uygun, Y.: A multi-agent-based communication prototype for cross-company capacity exchange in manufacturing networks, 1 April 2016. https://doi.org/10.2139/ssrn.3909338
3. Ruf, P., Stodt, J., Reich, C.: Security threats of a blockchain-based platform for industry ecosystems in the cloud. In: 2021 Fifth World Conference on Smart Trends in Systems Security and Sustainability (WorldS4), pp. 192–199 (2021). https://doi.org/10.1109/WorldS451998.2021.9514058
4. KOSMoS. https://www.kosmos-bmbf.de/kosmos-system/. Accessed 04 July 2022
5. Polge, J., Robert, J., Le Traon, Y.: Permissioned blockchain frameworks in the industry: a comparison. ICT Express **7**(2), 229–233 (2021). ISSN: 2405-9595. https://doi.org/10.1016/j.icte.2020.09.002
6. Bux, T., Riedel, O., Lechler, A.: Blockchain based approach on gathering manufacturing information focused on data integrity. In: Liewald, M., Verl, A., Bauernhansl, T., Möhring, H.C. (eds.) WGP 2022. LNPE, pp. 473–483. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-18318-8_48
7. Shevchenko, N., Chick, T.A., O'Riordan, P., Scanlon, T.P., Woody, C.: Threat modeling: a summary of available methods. Technical report 2018-07-01,Defense Technical Information Center. https://apps.dtic.mil/sti/citations/AD1084024. Accessed 04 July 2022
8. Kohnfelder, L., Garg, P.: The threats to our producs. Microsoft Security Blog. https://adam.shostack.org/microsoft/The-Threats-To-Our-Products.docx. Accessed 04 July 2022

9. Microsoft Threat Modelling Tool. https://docs.microsoft.com/de-de/azure/security/develop/threat-modeling-tool-threats. Accessed 04 July 2022
10. Threat Dragon Github. https://github.com/OWASP/threat-dragon/. Accessed 04 July 2022