Gateway for Industrial Cyber-Physical Systems with Hardware-based Trust Anchors

Diethelm Bienhaus, Lukas Jäger, Roland Rieke, and Christoph Krauß

Abstract Industrial Cyber-Physical Systems require appropriate security mechanisms to provide protection against cyber attackers. In this paper, we propose a security architecture for a gateway connecting production and cloud systems. A Trusted Platform Module 2.0 is used for protecting the cryptographic keys used in secure communication protocols and to provide protection against illegitimate firmware manipulation. As proof of concept, we implemented the key protection functionality with a TPM 2.0 for the OPC UA protocol.

1 Introduction

Industrial Cyber-Physical Systems (ICPS) are becoming more and more important in manufacturing, sales, and logistics. Networked smart production systems exchange data using different protocols such as Message Queuing Telemetry Transport (MQTT) or OPC Unified Architecture (OPC UA) with each other and with large distributed cloud systems like Manufacturing Execution Systems (MES) or data analytic tools. To protect against cyber attackers, appropriate security mechanisms are required.

In this paper, we propose a security architecture for a gateway connecting production systems such as sensors and actors with cloud systems. Our security architecture is

Diethelm Bienhaus
Institute of Technical-Informatics, Department of Mathematics, Natural Sciences and Computer Science, University of Applied Sciences Mittelhessen, Giessen, Germany
e-mail: diethelm.bienhaus@mni.thm.de

Lukas Jäger, Roland Rieke, Christoph Krauß
Fraunhofer Institute for Secure Information Technology, Darmstadt, Germany
e-mail: lukas.jaeger|roland.rieke|christoph.krauss@sit.fraunhofer.de

based on a Trusted Platform Module (TPM) 2.0 and has two main features. First, it provides secure storage and usage of cryptographic keys used for securing communication protocols such as OPC UA or MQTT within Transport Layer Security (TLS). Second, the integrity of the platform is secured, i.e., an attacker cannot successfully manipulate the firmware of the gateway without being detected. To show the feasibility of our approach, we integrated TPM usage into the open source OPC UA implementation open62541 [3]. The private keys used by OPC UA security are protected by the TPM.

2 Background and Related Work

A wide range of communication protocols are common in CPS. In the industrial domain specialized protocols have evolved. A short overview is given in table 1.

Table 1 Typical IoT protocols [12], [15]

Abbr.	Name	Standardization body
AMQP	Advanced Message Queuing Protocol	ISO/IEC 19464
CoAP	Constrained Application Protocol	Internet Engineering Task Force (IETF)
DDS	Data Distribution Service	Object Management Group
MQTT	Message Queuing Telemetry Transport	OASIS

The most profound protocol for Internet connectivity in the industrial domain is the Open Platform Communications (OPC) Unified Architecture (OPC-UA). OPC UA provides the infrastructure for interoperability across different hard- and software platforms, such as PC hardware, Cloud-based servers, PLCs, micro-controllers running Microsoft Windows, Apple OSX, Android, or Linux. OPC UA comprises a generic object model including a type system and an information model [17]. OPC UA has an elaborated security specification as part of the standard body [7]. The German Federal Office for Information Security (Bundesamt für Sicherheit in der Informationstechnik, BSI) carried out a comprehensive safety analysis of OPC UA in 2015 [10]. The study included the following security tests: Certificate tests, static code analysis, fuzzing and dynamic code analysis. The study came to the conclusion that OPC UA, unlike most others industrial protocols, offers a high level of security. No systematic errors could be detected. BSI recommends OPC UA as the preferred protocol for ICPS.

Elleithy et al. evaluate the security gain of OPC UA in [9]. They also regard it as significant even without any further hardware security measures for key protection.

In [8] the authors propose to use an SDN gateway as a distributed means to monitor traffic from IoT based devices and detect anomalous behavior. This gateway can also initiate countermeasures such as blocking traffic.

Although the OPC Foundation recommends to use TPMs as secure key storages for OPC UA [14], no related work on the specifics of this approach could be found.

Trust establishment in ICPS should - in addition to securing the gateway - also take into account trusted event reporting for critical event sources, so that the authenticity of the security related events can be verified [16].

3 Gateway Architecture

The aim of this work is to provide a platform for secure vertical integration from the sensor level to Cloud-based systems like MES or analytics allowing productivity assessment and predictive maintenance. A special emphasis is put on a combined hard- and software based security approach. We use a TPM [13] as a hardware anchor for cryptography in combination with signed code procedures as a software-side security solution. In order to achieve the approach, the most important task is interfacing open62541 with the open source TCG TPM Software Stack (TSS) developed by Intel, Infineon and Fraunhofer SIT [6] In order to support a large variety of application scenarios not only one protocol has to be included. The proposed architecture aims on multi-connectivity. On a simple level lightweight protocols such as MQTT and CoAP are implemented. To give receiving applications more domain specific, semantic data PPMP and OCF IoTivity are supported. Retrofit of older machines is a challenge especially for small and medium enterprises (SMEs). Hence integration of various sensors is intended. Besides already installed sensors data acquisition is additionally carried out by means of recently developed industrial wireless sensor technologies. Interfaces to standard field-bus systems are designed.

To achieve integration of many field bus protocols an abstraction of them is necessary. Hence a generic data model was developed comprising the entities "device", "resource" and "property". A device can have one or more resources which can have one or many properties. Each of those entities have data fields like IDs, names or time-stamps.

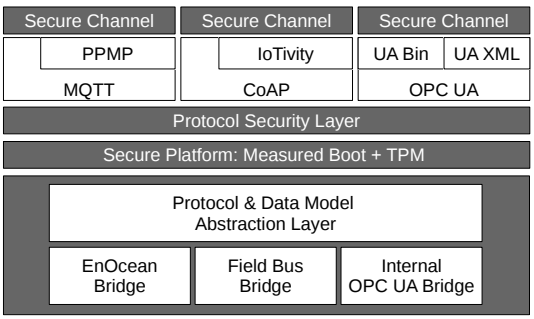

Fig. 1 Layered architecture with protocol specific I/O modules and integration of TPM / measured boot

An import design decision had to be made: should the data model be represented in an application specific implementation or is one of the addressed protocols suitable to represent the generic data model. Since OPC UA consists of a data model and features easy integration of the other protocols, our model is adapted to OPC UA. The OPC information model framework enables domain specific extendibility [11]. Hence our generic data model is managed in an OPC UA server which works as the core of the abstraction layer in Fig. 1. Import modules for the different data sources are implemented in such a way that they all use a common interface of the core server. The export modules work similar: data transfer is coordinated by the core server application. TPM based security is integrated on each level of the layered architecture as explained below.

4 Security Concept

In order to explain the security concept of our solution, we first define the attacker model and evaluate the threats it has to face. From this we derive a security concept consisting of three building blocks: Measured Boot in combination with Remote Attestation, OPC UA with secure communication using the TPM as a hardware anchor and binding of the OPC UA keys to the firmware integrity of the device.

4.1 Attacker Model

We assume that the biggest threat to the Industrial Internet of Things (IIoT) comes from remote attacks. This is due to the exposure of data and services to the internet. Attacks on any IIoT system most likely have the goal of gaining control over the system and modifying it. This class of attacks can be attributed to a wide range of black-hat hackers with very different motivations and skill sets. For IIoT-solutions in small-scale businesses, we focus on attackers that have medium skills and are able to gain remote access to the gateway and start manipulating the firmware.

With the data produced by IIoT devices and machines becoming a valuable asset, attacks that try to siphon this data off will become more common and dangerous. A basic approach for such an attacker is to listen to the gateway's plaintext communication. A more sophisticated approach might gain access to the gateway and try to extract the keys to eavesdrop on future encrypted communication. Extracting keys can also serve the forging of authentications.

4.2 Measured Boot and Remote Attestation

Measured boot is established as an effective countermeasure against modifications of firmware. The TPM facilitates this technique with several Platform Configuration Registers (PCR). These are special registers that can be written only via the *Extend*-operation that concatenates the data to write to the old value of the PCR and computes a hash of the result:

$$PCR_{new} = Hash(PCR_{old}||data)$$

Integrity measurement solutions such as the Integrity Measurement Architecture (IMA) [1] now compute a hash of every file that is opened and every process that is executed. This results in a list of all opened files and executed processes and a compound PCR value that results from hashing all these files and processes to a PCR. These are used for remote attestation to a verifier. For this, the PCR is signed digitally by the TPM and the signature and the list of files and processes is sent with the PCR value to the verifier. The verifier needs to check the list for faults, compare the final PCR of the list with the signed PCR value and verify the signature itself. This technique is used to detect firmware manipulations quickly.

4.3 OPC UA with a TPM-based Secure Key Management

OPC UA uses established network standards such as TLS and HTTPS for secure communication. TLS-based standards enable cryptography primitives such as authentication, authorization, encryption and integrity protection for communication. This effectively blocks all passive eavesdropping attacks. However an intruder could still extract cryptographic keys and decrypt the communication or forge authentications. Therefore it is important to protect the keys. The TPM was designed for the secure management of cryptographic keys. It is able to create, manage and store keys securely. Therefore, integrating the TPM into TLS and OPC UA should be adequate to prevent attackers from extracting keys. The TPM 2.0 features two different hierarchies that user keys can be created in: The Storage Hierarchy for keys that are not used to identify the TPM (like disk encryption keys) and the Endorsement Hierarchy that stores keys derived from the TPM's unique Endorsement Key and therefore allow to identify the TPM. When the TPM is used as a key storage for OPC UA, the OPC UA keys must be included in a suiting hierarchy. Encryption keys are mapped to the Storage Hierarchy. Signature keys for authentication can technically be mapped to both hierarchies. Using the Endorsement Hierarchy is the obvious choice and prevents misuse because it only allows the creation of signature keys. However the binding to the Endorsement Key may have unintended side effects. For example it does not allow export of the key if the TPM is to be exchanged.

The use of the Storage Hierarchy is unrestricted and does not come with any drawbacks. Therefore, we map all keys to the storage hierarchy of the TPM. With the secure TPM-based mechanisms of creating and managing cryptographic keys, OPC UA is secured against most key extraction attacks.

4.4 Binding TPM Keys to Firmware Integrity

In order to grant authorization to use a key, a TPM can either use password authentication or Enhanced Authorization Policies. The latter are a flexible way of binding the authorization to use a key to certain preconditions including the availability of a digital signature, time, password or a certain digest within a PCR. All policies can be combined to enforce any combination of preconditions.

In order to fulfill policies, an authorization session is started in the TPM. Each session has a session digest that is changed after each policy command using a hash function. The change occurs in a way that cryptographically includes the enforced precondition. Usage of a key is only allowed if the session digest is the same as the specified session digest that is bound to a key. The OPC UA keys are bound to the resulting digest of the TPM policy command *PolicyAuthorize*. This allows usage of a key, if a previous set of policies lead to a session digest that was signed with a certain authorization key outside the TPM. With this, the gateway attests its firmware to a remote verifier by sending the PCR value and measurement list. After verifying both, the verifier computes the expected session digest after executing the policy command *PolicyPCR* with it and signs that digest with the private authorization key. This signature is sent to the gateway. The gateway executes *PolicyPCR* and subsequently *PolicyAuthorize* with the received signature. *PolicyAuthorize* unlocks the key only if the session digest produced by *PolicyPCR* is the same that was signed by the verifier. This is a more flexible way to react to changes in a PCR. Both implementations are suited to lock down sensitive keys in case the firmware was tampered with.

Figure 2 shows an exemplary authentication session with a remote verifier for an OPC UA key. The session and PCR digests are 32-bit hexadecimal values for demonstration purposes. In real applications, the length of these digests is determined by the selected hash algorithm. The OPC UA key is bound to a session digest of $D34DC0D3_{hex}$ which is achieved by the *PolicyPCR*-command that evaluates to $02AB1D00_{hex}$ if the PCR contains $C0D3B332_{hex}$. The session digest after *PolicyPCR* is signed by the remote verifier and sent to the TPM to fulfill *PolicyAuthorize*, which will set the session digest to the desired value if the signature is valid and the session digest before the command is equal to the session digest in the signature.

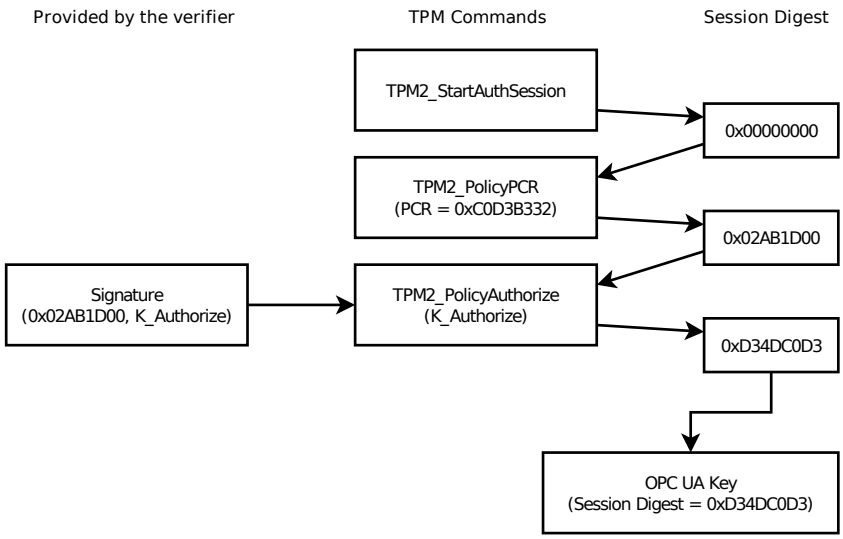

Fig. 2 Example of Enhanced Authorization Policies binding the key to the firmware integrity

5 Implementation

The open62541 OPC UA implementation is written in C and follows a modular approach. Hardware or platform specific functionality is replaceably encapsulated by defining interfaces. This allows open62541 to use either openSSL [4] or mbedTLS [2] as a cryptography backend. A TPM integration for openSSL was already implemented [5]. In the context of the Internet of Things a more lightweight cryptography library is needed. Therefore for the proposed concept TPM functionality is included into the mbedTLS library. The openSSL integration can be used for OPC UA on less resource-constrained devices. The implementation of this TPM integration into mbedTLS is currently in progress. It will map all asymmetric keys to storage hierarchy keys as described in subsection 4.3.

6 Conclusion and Future Work

The concept and rationale for a secure gateway that enables unidirectional access to sensor data and other data from the field-bus level is introduced. Trusted Platform Modules and software encryption were integrated for secure connectivity of the gateway with Internet and Cloud-based applications.

In an evaluation phase already installed vacuum machines of the industry partner will be equipped with the developed gateway to send a variety of machine data to Cloud-based tools like predictive maintenance applications.

Acknowledgements This project (HA project no. 574/17-56) is funded in the framework of Hessen ModellProjekte, financed with funds of LOEWE – Landes-Offensive zur Entwicklung Wissenschaftlich-ökonomischer Exzellenz, Förderlinie 3: KMU-Verbundvorhaben (State Offensive for the Development of Scientific and Economic Excellence).

References

1. Integrity Measurement Architecture (IMA). https://sourceforge.net/p/linux-ima/wiki/Home/. Accessed: 2019-06-18
2. MbedTLS. https://github.com/ARMmbed/mbedtls. Accessed: 2019-06-18
3. open62541. https://github.com/open62541/open62541. Accessed: 2019-06-18
4. OpenSSL. https://www.openssl.org/. Accessed: 2019-06-18
5. OpenSSL engine for TPM2 devices. https://github.com/tpm2-software/tpm2-tss-engine. Accessed: 2019-04-25
6. OSS implementation of the TCG TPM2 software stack (TSS2). https://github.com/tpm2-software/tpm2-tss. Accessed: 2019-04-25
7. Unified Architecture Part 2: Security Model. https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-2-security-model. Accessed: 2019-06-18
8. Bull, P., Austin, R., Popov, E., Sharma, M., Watson, R.: Flow based security for iot devices using an sdn gateway. In: 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), pp. 157–163 (2016). DOI 10.1109/FiCloud.2016.30
9. Elleithy, K., Sobh, T., Iskander, M., Kapila, V., Karim, M., Mahmood, A.: Technological Developments in Networking, Education and Automation. Springer Netherlands (2010)
10. Federal Office for Information Security: OPC UA Security Analysis. Tech. rep. (2017). https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/OPCUA/OPCUA.html
11. Gaj, P., Kwiecień, A., Sawicki, M.: Computer Networks: 24th International Conference, CN 2017, Lądek Zdrój, Poland, June 20–23, 2017, Proceedings. Communications in Computer and Information Science. Springer International Publishing (2017)
12. Mala, D.: Integrating the Internet of Things Into Software Engineering Practices. Advances in Systems Analysis, Software Engineering, and High Performance Computing (2327-3453). IGI Global (2019). URL https://books.google.de/books?id=GPGCDwAAQBAJ
13. Mitchell, C.: Trusted Computing. Institution of Electrical Engineers (2005)
14. OPC Foundation: Practical Security Recommendations for Building OPC UA applications. Tech. rep., OPC Foundation (2018). URL https://opcfoundation.org/wp-content/uploads/2017/11/OPC-UA-Security-Advise-EN.pdf
15. Raj, P., Raman, A.: The Internet of Things: Enabling Technologies, Platforms, and Use Cases. CRC Press (2017). URL https://books.google.de/books?id=cLI0DgAAQBAJ
16. Rein, A., Rieke, R., Jäger, M., Kuntze, N., Coppolino, L.: Trust Establishment in Cooperating Cyber-Physical Systems, *Lecture Notes in Computer Science*, vol. 9588, pp. 31–47. Springer International Publishing, Cham (2016). DOI 10.1007/978-3-319-40385-4_3
17. Rinaldi, J.: OPC UA Unified Architecture: The Everyman's Guide to the Most Important Information Technology in Industrial Automation. CreateSpace Independent Publishing Platform (2016)