Integration of a Security Gateway for Critical Infrastructure into Existing PKI Systems

Münch, Andreas; Frauenschläger, Tobias; Mottok, Jürgen
Ostbayerische Technische Hochschule Regensburg
https://doi.org/10.57688/365

Abstract

The Transport Layer Security (TLS) protocol is used to cryptographically secure network connections. To ensure authenticity, TLS uses certificates that are exchanged at the beginning of each new connection. Due to expiration or early revocation of certificates, the deployment of new certificates to devices in the field is necessary. In addition, a device must identify revoked certificates during connection establishment to abort the connection. This paper presents the implementation of these two functionalities within a Security Gateway for the power grid. The nature of embedded systems with their limited resources and requirements regarding dependability impact the device-specific implementation. With these features, the Security Gateway can be integrated into an existing Public-Key Infrastructure System.

1. Introduction

Cyberattacks on critical infrastructures are currently on the rise [1]. In addition to water supply, critical infrastructures also include the power grid. Besides the private consumer and the industry, the power grid also provides energy for hospitals, traffic lights, and water pumps. Since the power grid is an important part of critical infrastructure, a successful cyberattack would have devastating consequences. The power grid mostly contains devices that do not have protective functions against such attacks, mainly due to the presence of legacy components from a time without awareness for security, leading to a dangerous security gap. The most vulnerable communication in the power grid takes place between a central control station and several substations, as the transmission lines of this wide-area connection are easily accessible for an attacker. Through this connection, data is exchanged to control and monitor the components necessary for grid stability.

The Energy Safe and Secure System Module (ES³M) project was launched to secure this communication path in the power grid. The main goal of this research project is to develop a module that can be inserted into the described wide-area connection, protecting it against Man-in-the-Middle attacks. A Man-in-the-Middle attacker infiltrates the communication and can read the messages between two partners or even manipulate them. To protect the communication path from that, two ES³Ms are integrated into the network connection between the controlling station and a substation, each next to one

legacy component. The two ES³Ms establish a secure Transport Layer Security (TLS) connection between each other with the protective measures. More information about the ES³M research project can be found in [2].

TLS uses certificates to ensure authenticity in a secure communication link. In the initial attempt of the device, the certificates that TLS requires for a secure connection must have been manually transferred to the ES³M via an adapter. This manuscript presents how this functionality can be automated. In addition to this, the ES³M cannot check whether a certificate received by the other ES³M in the communication path has been revoked or not. The implementation for that functionality shall also be described in this work. In the following, the contributions that are fulfilled in this work are summarized:

Contribution 1

Implementation of a request functionality on the ES³M to apply for a new certificate over the network.

Contribution 2

Implementation of a verification functionality on the ES³M to check the revocation status of a certificate received from the Peer-ES³M at the beginning of every new TLS connection.

In Chapter 2, the technical background of this manuscript is provided. The realization of Contribution 1 is presented in Chapter 3. Chapter 4 shows the implementation of Contribution 2. The manuscript is concluded by Chapter 5.

2. Technical Background

In this chapter, the technical background of this manuscript is introduced. Firstly, information about the context of the work is provided in Section 2.1. This context consists of a short overview of the ES³M and a brief description of how TLS is used on the ES³M. Section 2.2 gives a short introduction about certificates that are also used by the ES³M to prove its authenticity. In Section 2.3, certificate management based on a Public-Key Infrastructure is presented.

2.1. Context

Transport Layer Security (TLS) is a protocol to secure the communication in computer networks. It is also used for the secure connection between two ES³Ms. To establish the TLS connection between the controlling station and the substation, one ES³M is placed in the controlling station and the other one in the substation, as shown in Fig. 1. The ES³M in the controlling station is acting as TLS client and the other one as TLS server [2].

Fig. 1: Establishing a secure communication channel between a controlling station and a substation with the help of two ES³Ms. For this purpose, one of the ES³Ms is placed in the Controlling Station next to the server and the other one next to the gateway of the substation [3].

When establishing a secure connection, a so-called TLS handshake is conducted. During this TLS handshake, both communication partners must prove their authenticity. This is also referred to as mutual authentication. In TLS, this authenticity is ensured by certificates.

The ES³M internally consists of four microcontrollers to fulfill the *Separation of Concerns* principle [2]. Three of them are used for handling network connections. The fourth microcontroller is responsible for the implementation of TLS on the ES³M. This also includes all functionalities regarding certificate handling. For the implementation of TLS on the ES³M, the WolfSSL library is used. In addition to the four microcontrollers, the ES³M also contains a smartcard storing the certificates and their related private keys in a highly secure manner. More information about the system architecture of the ES³M can be found in [2].

2.2. Certificates

As already mentioned in Section 2.1, certificates are needed to provide the authenticity of a TLS connection. For this purpose, the certificate contains a signature created by the private key of an Issuer-Certificate on a higher authority level. This signature can be verified by every entity possessing the Issuer-Certificate. The certificate on this higher authority level is also referred to as *Root-Certificate*. The Root-Certificate must be stored on every device in a communication network as a trust anchor, so each one can verify the authenticity of a communication partner. The Root-Certificate is self-signed, meaning its signature is created by its associated private key. All this gives the Root-Certificate a very important role. With the public key of the Root-Certificate, one can verify signatures created with the associated private key. For the ES³M, another certificate was used in addition to the Root-Certificate and the certificate at the lowest authority level, referred to as *Device-Certificate*. This additional certificate is denoted as *Intermediate-Certificate* in the following, as it is on an authority level between the Device-Certificate and the Root-Certificate. It is used to not require the Root-Certificate and its private key for every issuance of a Device-Certificate. Therefore, it is less likely that the critical trust anchor of the system will be compromised.

The validity period of a certificate is determined by a start-date and an end-date. However, it is possible to invalidate a certificate prior to its expiration. This process is called *revocation*. Reasons for such a revocation can be the compromise of the corresponding private key or the withdrawal of privileges of the entity using the certificate. With a revoked certificate, a connection must not be established successfully. However, it is the responsibility of a device to check the revocation status of the received certificate during the TLS handshake. Based on this description of certificates, the next section presents a concept for managing them.

2.3. Public-Key Infrastructure

A Public-Key Infrastructure (PKI) is a system to generate, distribute, verify, and revoke digital certificates [4]. To be able to perform all these tasks, the PKI consists of all the parts listed in the following:

Certificate Authority (CA)

The CA is responsible for the issuing of new certificates. Using the private key of an authorized certificate, the CA signs a certificate and starts its lifecycle.

Validation Authority (VA)

The VA provides services to check the validity of a certificate. These are described in more detail in Section 4.1.

Registration Authority (RA)

The RA supervises the parties that connect to the CA to request a new certificate. It is an intermediary between the CA and a client applying for a new certificate.

All the mentioned components of the PKI are separated for security reasons. Detailed information about a PKI and its components can be found in [4]. These components form the common authority needed for key management in a secure communication network. In an early stage of the device, the ES³M could not interact with the mentioned components of the PKI. It could neither apply for a new certificate independently nor check if a received peer-certificate has been revoked. For this reason, mechanisms have been implemented connecting the ES³M with the relevant parts of the PKI. In the ES³M project, the PKI is provided by the power transmission company. All the mechanisms used in this work are also recommended by the IEC 62351-9 [5] standard. This standard defines requirements for the key management of devices used in the power grid. The implementation of the request functionality of a new Device-Certificate as the first of the two features is presented in the next chapter.

3. Request of a new Device-Certificate

To implement the functionality presented in Contribution 1, the first section defines the procedure to request a new Device-Certificate at the PKI. Thereafter in the second section, the actual implementation for the ES³M is presented.

3.1. Procedure of the Certificate-Request

The seven steps illustrated in Fig. 2 show the procedure executed by the ES³M to obtain a new Device-Certificate. To apply for a new certificate, a *Key-Pair* consisting of a private key and a public key must be generated first. After the generation of the Key-Pair, a *Certificate Signing Request* (CSR) is formed of the public key and other necessary metadata like the proposed name of the certificate.

This CSR is then transferred to the CA via an authentic channel. The CA verifies the CSR and generates a new Device-Certificate by creating a signature with the help of the Intermediate-Certificate. The certificate received by the ES³M is base-64 encoded and is in the PKCS #7 format [6]. It must be decoded and converted first before storing it on the Smartcard. After a reboot, it can be used as a new Device-Certificate.

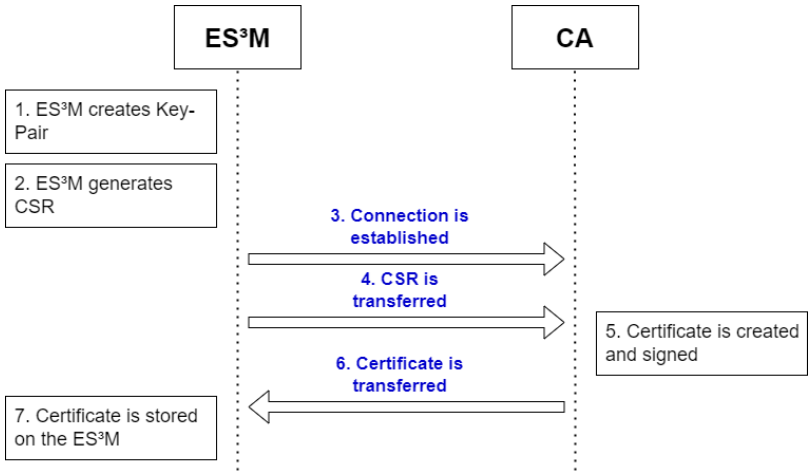

Fig. 2: Representation of the complete application process for a new certificate. All messages exchanged between the ES³M and the CA are highlighted in blue.

The CSR must comply with the PKCS #10 standard. In addition to the public key, it also includes a name and other additional metadata. This information is stored on the ES³M so it can be added to the CSR. The complete CSR is then signed by the corresponding private key to ensure its integrity.

Besides the integrity of the CSR, the authenticity of the ES³M applying for a new Device-Certificate must also be ensured. It must not be possible for any device to request a new certificate that is not part of the secure network. For this purpose, the EST protocol, which is based on TLS, is used for the communication with the RA and the CA of the PKI, providing the authenticity of the communication partners [6].

3.2. Implementation

Before considering the actual steps to request a new Device-Certificate, it must be defined how to start this process at all. For the current prototype setup, it can be triggered by a client software tool connected to the ES³M. However, it is conceivable to trigger the process automatically once the Device-Certificate is considered invalid by the device itself (e.g. through periodic revocation and expiration checks).

For the integration of the request procedure into the firmware, the existing software infrastructure for establishing TLS connections is re-used. The actual procedure is separated into two new tasks, denoted *CertificateTask* (CT) and *NetworkTask* (NT), in the following. The CT is responsible for the internal key and certificate management, whereas NT implements the endpoint for the EST protocol. All tasks possess a message queue and an internal communication interface to exchange data in a synchronized and efficient manner. The sequence of operation is depicted in Fig. 3.

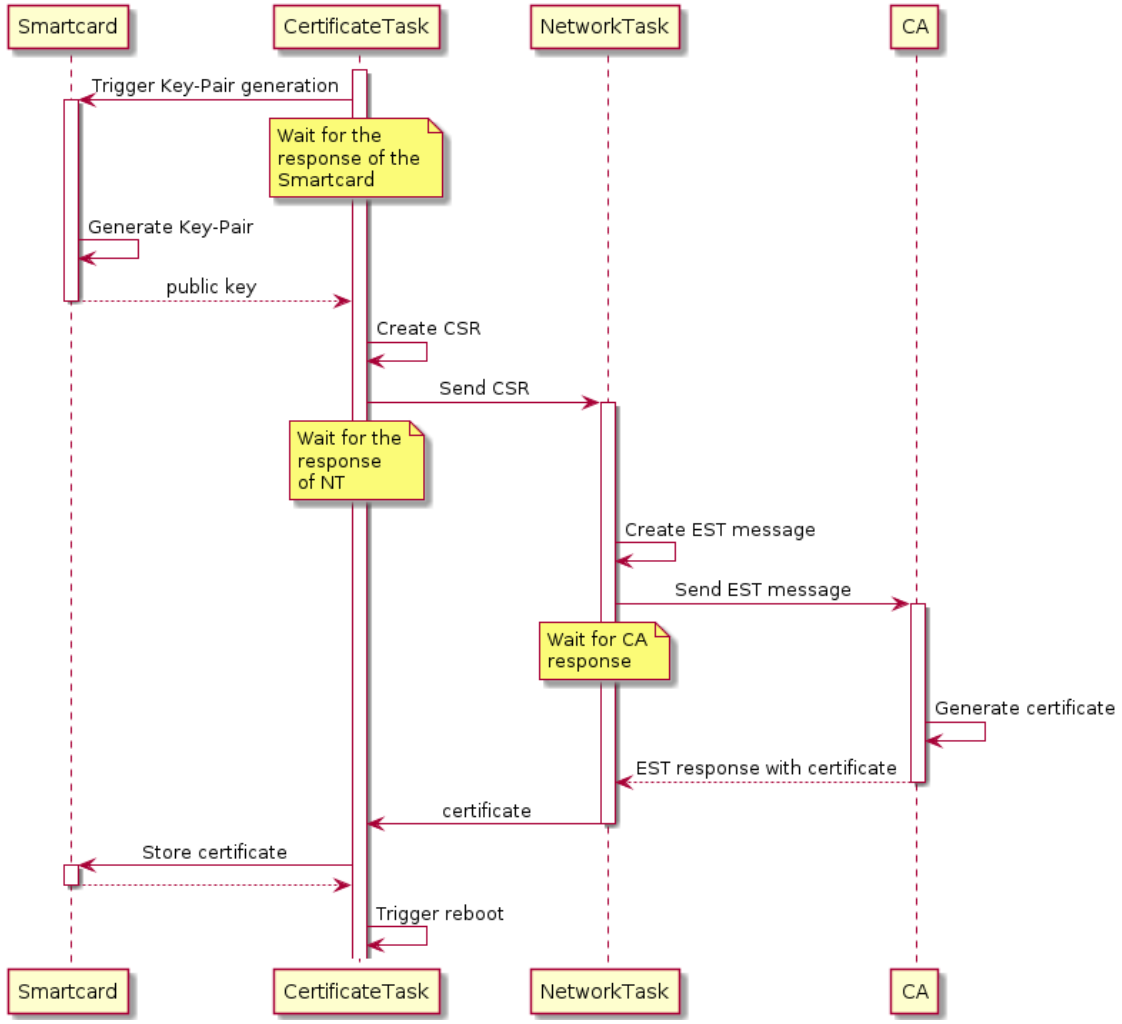

Fig. 3: The two new tasks, NetworkTask and CertificateTask, implement the certificate request functionality. To simplify the figure, the software parts in between the NetworkTask and the CA for the TLS functionality have been omitted.

Once the procedure is started through the external client software tool, the CT triggers the Key-Pair generation on the smartcard and then creates the CSR. The generation of the appropriate EST message and its transfer to the PKI is taken over by the NT (with support of the existing TLS functionality). Once the certificate is generated by the CA and sent back to the ES³M, the NT decodes the EST response and forwards the certificate to the CT. Finally, the CT stores the new certificate on the smartcard and triggers a reboot, so the new certificate is loaded to be used for the next TLS connection. This concludes the description of Contribution 1, leading to the discussion of Contribution 2 in the next chapter.

4. Revocation Status Verification of Peer-Certificates

At the beginning of every new TLS connection, both communication partners exchange their certificates. A certificate can lose its validity after it has been issued prior to its expiration through revocation. The ES³M must be able to identify a revoked certificate during the TLS handshake to abort the connection establishment. This chapter starts with an introduction of different mechanisms to identify a revoked certificate in Section 4.1. After this, the realization of Contribution 2 is described in Section 4.2.

4.1. Conceptual-Analysis

In this section, two different mechanisms are introduced to check if a received certificate has been revoked. All the concepts presented in the following are also recommended in the IEC 62351-9 [5] standard.

Certificate Revocation List (CRL):

The first approach to identify revoked certificates is to put all of them on a list. These so-called *Certificate Revocation Lists* (CRL) are stored locally on a device. A CRL lists every certificate revoked by the CA of the PKI. To also identify even recently revoked certificates, the lists must be updated periodically, as certificates not on the list are considered valid. The CRLs are distributed by the VA of the PKI, from where every entity in the system must actively download them [7].

Online Certificate Status Protocol (OCSP)

The second option to verify the revocation status of a Peer-Certificate is to contact the VA during the TLS handshake. The verification of the revocation status is performed by sending a request message from the device to the VA using the *Online Certificate Status Protocol* (OCSP). In contrast to CRL, OCSP provides a positive response that clearly shows that the received certificate has not been revoked. In addition to this, the positive response of the VA is always up to date. Disadvantages of the OCSP method are the additional network connection to the VA during the handshake and the dependence on the availability of the VA [8].

4.2. Implementation

As already mentioned in the previous section, CRL and OCSP have both advantages and disadvantages. For this reason, a combination of both mechanisms is used within the ES³M. The collaboration of the CRL and OCSP mechanisms is displayed in Fig. 4.

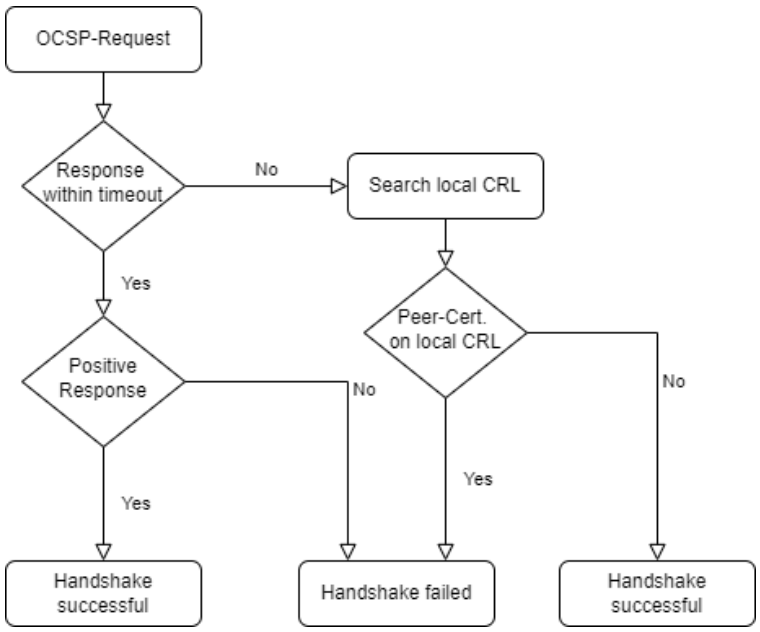

Fig. 4: Illustration of the complete verification process, which consists of an OCSP-Request and the optional verification with a local CRL. If there is no status response ("good" or "revoked") to the OCSP-Request or an error occurs, the ES³M will check if the Peer-Certificate is listed as revoked on the local CRL.

To ensure that the ES³M always has the latest information about the status of the received certificate, the OCSP procedure is used first. If there is no response from the VA within a specified timeout or an error occurs, a local CRL is searched for the received certificate. Through the local CRL, the ES³M can be protected against denial-of-service attacks caused by preventing the communication between the device and the VA.

The implementation of the two protocols is done in a similar way as of Contribution 1. However, a large part of the functionality is already supplied by WolfSSL. The library supports the OCSP protocol and can handle CRLs automatically. Furthermore, the sequence of Figure 4 has been integrated into WolfSSL by customizing the error handling of the library. For the OCSP functionality, a single task has been created similar to the NetworkTask of Section 3.2. The task gets the OCSP-Request message from WolfSSL, wraps it in an HTTP message, which is used as a transport mechanism, and sends it to the PKI. When the response is received, the timeout expires, or another error occurs, the task hands the response (or an error message) back to WolfSSL. Regarding CRLs, WolfSSL automatically triggers the download of a new list once the

current list is considered outdated. This trigger is handled by an additional new task, which downloads the new CRL file from the PKI using HTTP. The obtained new list (or an error message) is handed back to WolfSSL and, in addition, stored persistently on the smartcard. During system startup, the ES³M loads the stored CRLs from the smartcard to always have a valid CRL available. Both new tasks use the existing network functionality of the ES³M to implement the underlying network connections for HTTP. With the description of the functionality to verify the revocation status of a certificate to fulfill Contribution 2, this manuscript is finalized by a conclusion in the next chapter.

5. Conclusion

Two contributions have been defined for this work. Contribution 1 implements a functionality on the ES³M to obtain a new Device-Certificate from the PKI. In Contribution 2, a functionality is created to check the revocation status of a certificate received from the Peer-ES³M during the TLS handshake. By integrating both contributions, the ES³M can now be successfully integrated into a PKI system.

This work described how different protocols and mechanisms can be used to support the administration of certificates on the ES³M. However, it has not yet been considered how the first Device-Certificate is transferred to the ES³M after its production. The goal of future work should therefore be to analyze the implementation of an automatic bootstrapping procedure for the first certificates of the ES³M.

Literatur

[1] Stringer, David; Lee, Heesu: Why Global Power Grids Are Still Vulnerable to Cyber Attacks. Online, last accessed on November 28, 2021, https://www.bloomberg.com/news/articles/2021-03-03/why-global-power-grids-are-still-so-vulnerable-to-cyber-attacks.

[2] Frauenschläger, Tobias; Dentgen, Manuel; Mottok, Jürgen: Systemarchitektur eines Sicherheitsmoduls im Energiesektor. Tagungsband 2. Symposium Elektronik und Systemintegration, Landshut, April 2020.

[3] Frauenschläger, Tobias: Design and Development of the Payload Processing Pipeline in a Multi-MCU Network Security Gateway. Master's thesis, September 2019.

[4] Weise, Joel: Public Key Infrastructure Overview. Online, last accessed: November 28, 2021, http://highsecu.free.fr/db/outils_de_securite/cryptographie/pki/publickey.pdf.

[5] IEC 62351:2017: Power systems management and associated information exchange - Data and communications security, 2017.

[6] Pritikin, M; Yee, P; Harkins, D: Enrollment over Secure Transport. Online, last accessed on February 15, 2021, https://tools.ietf.org/html/rfc7030.

[7] Cooper, D; Santesson, S; Farrell, S; Boyen, S; Housley, R; Polk, W: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Online, last accessed on December 15, 2021, https://datatracker.ietf.org/doc/html/rfc5280.

[8] Santesson, S; Myers, M; Ankney, R; Malpani, A; Galperin, S; Adams, C: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP. Online, last accessed on December 15, 2021, https://datatracker.ietf.org/doc/html/rfc6960.

Kontakt

Andreas Münch, B. Eng.
Ostbayerische Technische Hochschule Regensburg
Laboratory for Safe and Secure Systems LaS³
Seybothstraße 2
93053 Regensburg
E-Mail: andreas.muench@oth-regensburg.de