Implementing Amazon Web Services (AWS) IoT Cloud Connectivity with OPTIGA™ Trust X

Dr. Christian Lesjak
Dr. Josef Haid

www.infineon.com/optiga-trust-x

Implementing Amazon Web Services (AWS) IoT Cloud Connectivity with OPTIGA™ Trust X

Connectivity to cloud-based services, which promise tremendous business value, is a major enabler for the Internet of Things (IoT). To realize this value, however, secured IoT devices and secured connectivity from IoT devices to cloud services is essential, especially in light of recent events where insecure IoT devices caused service outages on a massive scale.

This white paper outlines a major cloud service (AWS) and its IoT-related security architecture. It also looks at the advantages of hardware-based security for IoT devices, going on to illustrate how an Infineon hardware security product can be integrated into an IoT device. In summary, this white paper shows that Infineon OPTIGA™ Trust X can be easily integrated with IoT devices in order to securely interact with AWS IoT.

Contents

1. Introduction

The Internet of Things (IoT) connects a diverse array of devices with each other or with cloud-based services. The ability to exchange data with other devices, services and humans enables IoT solution providers to develop new use cases and business models.

Today, communication technologies provide connectivity almost anywhere around the world, at ever-increasing data speeds. The proliferation of cloud-based storage and computing infrastructure has led to a multitude of cloud-based technology offerings from both large players such as Amazon, Google and Microsoft, but also from many newcomers. And the broad availability of hardware and software for embedded connected devices has broadened significantly. Affordable single-board computers such as Raspberry Pi or microcontroller platforms such as Arduino empower millions of developers to engineer new applications.

However, as more and more devices connect to the Internet, they are also becoming increasingly attractive targets for deliberate attacks. One notable recent example was the Mirai botnet in 2016. This virus used malware that “enslaved” IoT devices [5] and used the resulting botnet to:

- take down parts of a domain name system (DNS) via a distributed denial-of-service (DDoS) attack and cause massive disruption to Internet services;
- conduct click fraud in online advertising and generate fraudulent profit and
- promote a DDoS mitigation business by targeting organizations with DDoS attacks.

All of these attacks were enabled by infected IoT devices. It is thus the fundamental duty and responsibility of IoT device vendors to properly protect their devices and services so that they are not vulnerable to such attacks and cannot be targeted in this way. Governments and regulators have started to increase the obligations on IoT device manufacturers, mandating that they implement proper technical security measures in their products and services.

This white paper focuses on how to securely connect IoT devices to a popular cloud service, Amazon Web Services (AWS). It reviews the architecture and security model offered by AWS IoT, going on to look at the benefits of hardware security modules and the integration effort they entail. In summary, this white paper shows that hardware security provides robust protection and works seamless with cloud services such as AWS IoT.

2. AWS IoT – A High-Level View

Cloud services provide remote infrastructure, services and solutions on a pay-per-use basis and do not require a large upfront investment. The core responsibility of the cloud service provider lies in operating its data centers and provisioning its infrastructure and services to cloud customers. Cloud customers, such as IoT solution providers, can thus benefit from high availability and scalability, coupled with the freedom to focus on their core business.

A popular public cloud platform is Amazon Web Services (AWS) [1]. AWS comprises more than 100 services including computing, storage, networking, database, analytics, application services and tools for the Internet of Things.

AWS IoT is a set of IoT-related tools and services running on AWS. AWS IoT enables bi-directional communication between IoT devices and the AWS cloud. The core of AWS IoT is a managed MQTT message broker which dispatches messages between IoT devices and other cloud services.

MQTT is an ISO-standardized, publish-subscribe message exchange protocol operating on top of TCP/IP. IoT devices that want to share data publish messages to an MQTT broker. IoT devices or services interested in receiving data subscribe to the MQTT broker. To receive relevant messages only, a subscriber tells the broker what message topics it is interested in.

Almost any kind of input device, also known as a “Thing” in the AWS IoT context, can be used. Amazon offers device SDKs for different programming languages and device platforms, including Python and embedded C SDKs [3], [4]. Messages dispatched via the MQTT broker can be distributed not only to other AWS services, but also to any other device or service that subscribes to the broker.
For example, a weather station might periodically publish temperature and relative humidity using the topics

- weatherstation/germany/munich/campeon/building1/temperature
- weatherstation/germany/munich/campeon/building1/humidity

Figure 1 summarizes the potential components of an AWS IoT solution. In the center, AWS IoT hosts a managed MQTT broker. Furthermore, AWS IoT provides authentication and authorization services to restrict access to the MQTT broker. A “things” registry organizes the resources associated with each thing. Device shadows provide persistent presentations even when IoT devices are temporarily disconnected. The rules engine supports integration with other services on AWS.

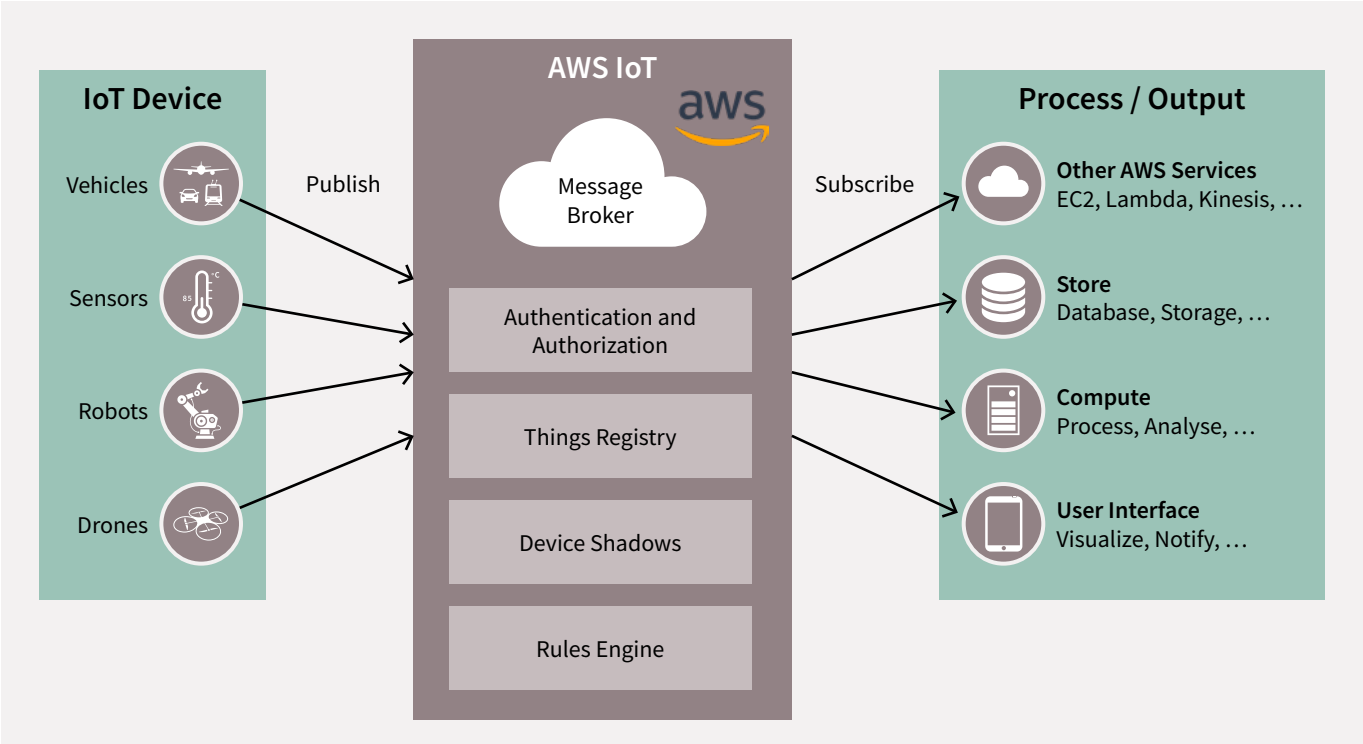

Figure 1: AWS IoT lets connected devices interact with other devices and services

3. AWS IoT Security

Amazon has put a lot of effort into securing the AWS infrastructure and its services. In this section, we will look at the AWS IoT security measures.

3.1 Shared Security Responsibility Model

When working with AWS, security tasks are shared between a solution provider and AWS; AWS secures the underlying infrastructure that supports the cloud services but the solution provider is responsible for securing anything they put on the cloud, or for securing any IoT device they connect to the cloud. This model is called the Shared Security Responsibility Model [6].

3.2 Transport Layer Security (TLS)

Transport Layer Security (TLS) is an Internet Engineering Task Force (IETF) standard for a secured communication protocol over a computer network [2].

This protocol cryptographically protects the integrity and confidentiality of messages that are exchanged among two peers during a TLS session (the TLS Record Protocol). Furthermore, TLS offers mutual authentication between two peers. Thus, before a TLS session is established between two peers, they have to complete what is known as a “handshake” (the TLS Handshake Protocol). During the handshake, both peers authenticate using public key cryptography and public key certificates.

As a prerequisite for mutual authentication, each peer has a long-term X.509 public key certificate issued by a trusted certification authority (CA). Such a certificate includes the peer’s public key, which corresponds to the peer’s private key, which is secret to the respective peer. Each peer must have the respective trusted CA certificates preconfigured in order to validate the derived peer certificate during the handshake. Such a trusted CA certificate is called the trust anchor.

3.3 AWS Public Key Infrastructure (PKI)

In AWS IoT, each IoT device is represented by a thing, a certificate and one or more policies. The thing can be administrated in the AWS IoT device registry. A device certificate is required for device authentication during the TLS handshake with AWS IoT. AWS IoT requires that the IoT device authenticate itself during the TLS handshake using a device-specific X.509 public key certificate. The policies are used to authorize IoT operations such as publishing or subscribing to MQTT topics.

Figure 2 summarizes the TLS-based security architecture in AWS IoT by illustrating a simple example with one IoT device and the AWS IoT endpoint. The AWS IoT cloud service endpoint is the solution provider's AWS IoT cloud service instance hosted by AWS.

The IoT device in Figure 2 has a device certificate and a corresponding device private key. Furthermore, a trust anchor to verify the AWS IoT endpoint certificate during the handshake is stored on the IoT device. As the TLS channel between the IoT device and the AWS IoT endpoint is mutually authenticated, the AWS IoT endpoint has a private key and public key certificate, as well as a trust anchor for the device certificate.

3.4 Enrollment and IoT Device Certificate Generation

Enrollment is the process of provisioning IoT devices with a long-term identity and linking this identify with an AWS IoT cloud service instance. The long-term identity is composed of the X.509 certificate and a corresponding private key. To link these devices with the AWS IoT cloud service endpoint, the trust relationship, thing representation and policies need to be set up in the AWS IoT cloud service instance. From a security perspective, the provisioning of device certificates and private keys is a critical step in the enrollment procedure. AWS IoT offers three methods to provision the X.509 device certificates and device private keys:

1. AWS-generated private key and AWS-issued device certificate

With the "one-click certificate creation", the AWS IoT console generates a device key pair and corresponding public key certificate, using the AWS IoT certificate authority. In this case, AWS (theoretically) knows the private key that was generated, and this key needs to be transferred from the AWS IoT console to the device. Since the certificate is issued by the AWS IoT certificate authority, this CA certificate becomes the trust anchor that has to be configured on the IoT device.
The disadvantage of this device certificate creation method is that the key is created outside the IoT device. Furthermore, this sensitive key material needs to be transferred from AWS IoT to the device.

2. Device-generated private key and AWS-issued device certificate

With the option "create with CSR", a certificate signing request (CSR) is uploaded from the IoT device to the AWS IoT console. With a CSR, the IoT device "applies" to the AWS CA for a digital certificate. The CSR is created with the private key of the IoT device. AWS IoT verifies that the CSR includes a public key that matches the CSR signature created with the private key. If validation is successful, AWS IoT issues a device certificate using the AWS IoT CA. With this option, generation and storage of the device private key is controlled by the solution developer.

3. Device-generated private key and public key-infrastructure (PKI) from the solution provider

The option "use my certificate" allows the solution provider to issue device certificates with its own certification authority. This solution provider CA is registered once with the AWS IoT endpoint, and subsequently all device certificates issued by this registered CA are accepted by AWS IoT when registering IoT devices.

Figure 2 depicts an IoT device provisioned with a trust anchor, device certificate and device private key. The IoT device uses the trust anchor to verify the AWS IoT endpoint certificate during authentication of the AWS IoT endpoint. The device private key and the device certificate are used to authenticate the IoT device to the AWS IoT endpoint. In a nutshell, all three credentials are used to mutually authenticate and secure the TLS link over which MQTT publishing and subscribing to AWS IoT takes place.

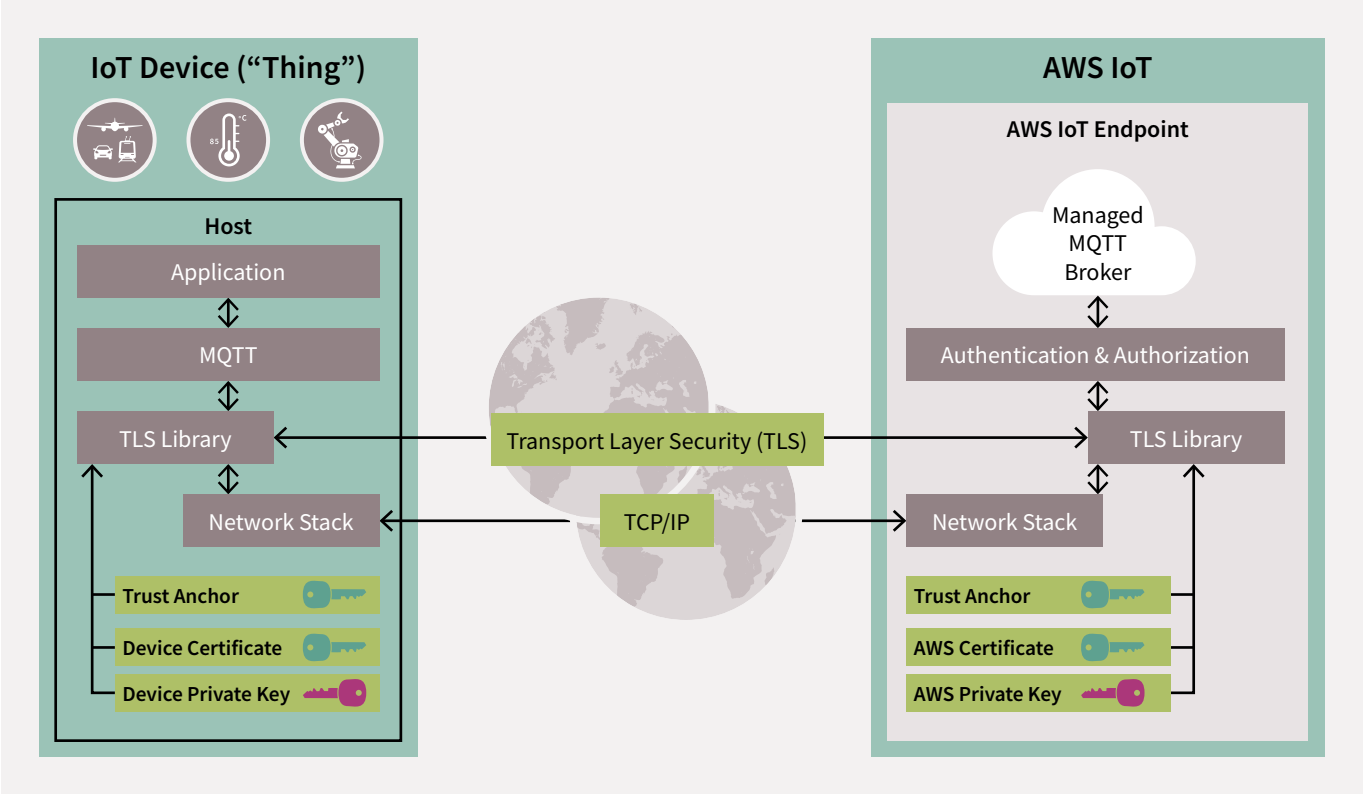

Figure 2: An IoT device uses a private key and a corresponding public key certificate to authenticate to the AWS IoT cloud service endpoint.

4. Securing IoT Devices Using Infineon OPTIGA™ Trust X

Depending on the IoT device and its application, the main processing component is usually a CPU or microcontroller, referred to as the host controller in the following. The IoT device's host controller typically handles the device's functionality, data collection and processing and remote data communication to the AWS IoT cloud, for instance.

The AWS Shared Security Responsibility Model requires that the IoT solution provider properly protect their IoT devices. This section explains how hardware-based security adds strong protection capabilities to IoT devices.

4.1 Hardware-based security

With hardware-based security, a discrete hardware security module (HSM) adds an especially hardened element to the IoT device host processor. Such an HSM safeguards the most sensitive data objects, the most critical cryptographic credentials and the most important processing steps by isolating data storage and processing on a dedicated integrated circuit.

In the baseline security recommendations for critical information infrastructures, the European Union Agency for Network and Information Security (ENISA) recommends hardware-based security to overcome potential vulnerabilities in IoT devices [7].

4.2 Infineon OPTIGA™ Trust X

Infineon OPTIGA™ Trust X is a turnkey hardware security module that offers the right set of security functions to protect IoT devices.

The function set of OPTIGA™ Trust X includes

- Operations to manage data objects such as certificates and trust anchors
- Authentication functions for one-way authentication and brand protection
- Authentication functions for TLS Handshake Protocol
- Complete DTLS protocol handling, including Handshake and Record Protocol
- Cryptographically secured random number generation
- A toolbox with cryptographic primitives and functions

The toolbox is a collection of cryptographic functions that enable or enhance cryptographic protocols that are not natively implemented inside the Infineon OPTIGA™ Trust X. Such use cases include platform integrity or secured communication and can be implemented using the following toolbox components:

- TLS-specific cryptographic primitives such as the calculation of the shared secret or key derivation
- Cryptographic key pair generation
- Cryptographic message digest calculation (so-called hashes)
- Digital signature computation using cryptographic keys stored securely inside the Trust X
- Digital signature verification using digital certificates stored securely inside the Trust X

An IoT solution provider can decide to what extent they wish to utilize the security features provided by Infineon OPTIGA™ Trust X. From a security perspective, the most sensitive and security-critical operations are those that use long-term cryptographic material, such as device authentication private keys.

4.3 Integration with an AWS-Connected IoT Device

Of the comprehensive set of functions offered by Trust X, those related to key and certificate management and those supporting the TLS protocol are the most important ones for secured AWS IoT connectivity.

Figure 3 depicts the system architecture of an IoT solution with an IoT device secured by OPTIGA™ Trust X. In this architecture, OPTIGA™ Trust X protects the long-term cryptographic material used for the TLS-based authentication with the remote AWS IoT endpoint. Trust X thus safeguards, stores and provides access to three important data objects:

- the long-term public key certificate for the IoT device ("device certificate"),
- the IoT device's long-term private key, which corresponds to the certificate ("device private key"), and
- the long-term server root CA certificate used to verify the remote endpoint ("AWS Trust Anchor")

Compared with the software-only security solution outlined in Section 3, the hardware-secured solution depicted in Figure 3 only implements a minor portion of the TLS library. This "host library for TLS" delegates the security-sensitive asymmetric key operations required during the TLS handshake to OPTIGA™ Trust X.

Electrically, Trust X is connected via the I2C bus to the IoT device host controller. The Infineon I2C protocol stack library enables communication with Infineon OPTIGA™ Trust X products. The protocol stack consists of multiple layers that relate to the ISO OSI (Open Systems Interconnection) model. A host controller-specific hardware abstraction layer (HAL), which interfaces with a host's I2C driver or I2C peripheral, enables flexible host controller support.

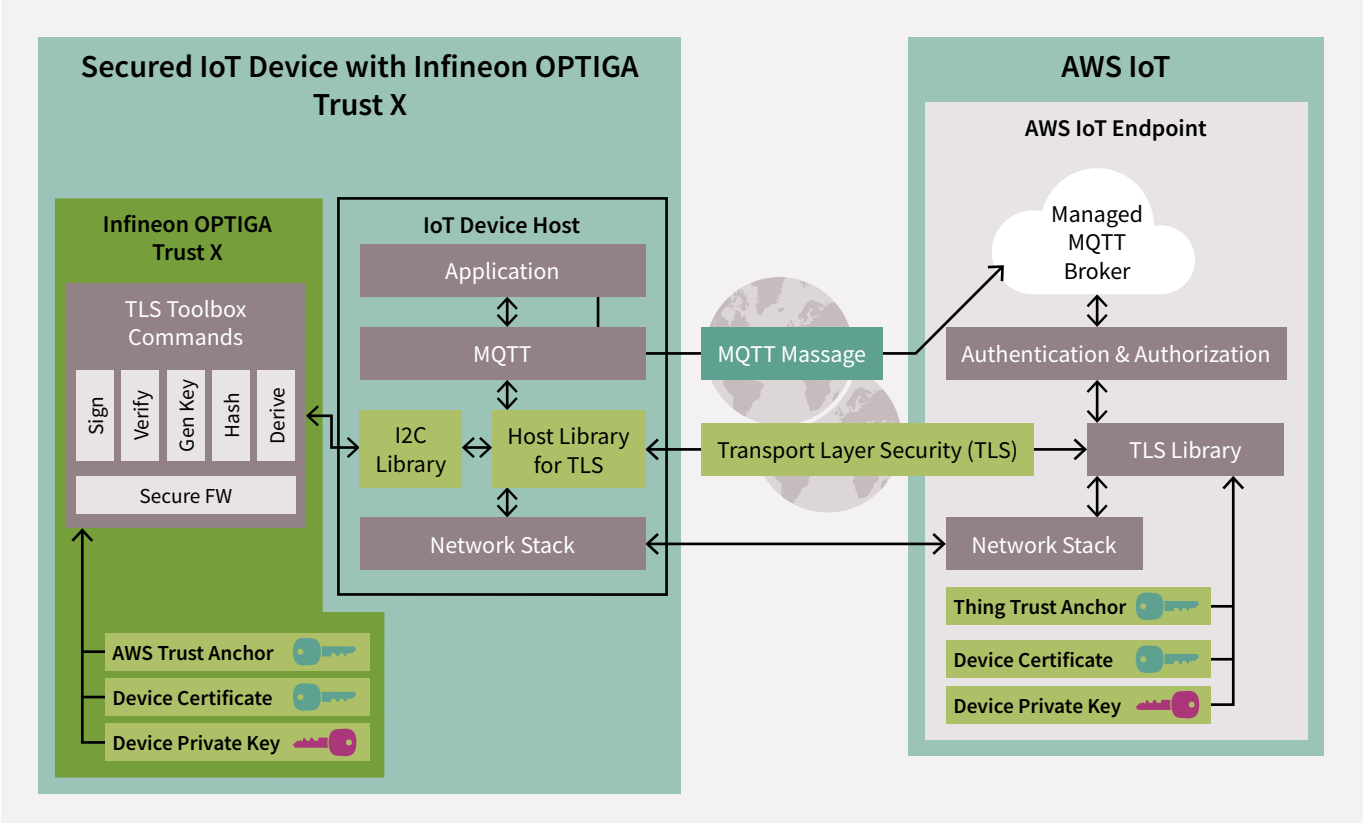

Figure 3: Integration of a hardware security module (Infineon OPTIGA™ Trust X) with an IoT device that securely publishes MQTT messages through a secured TLS channel over the Internet to the AWS IoT message broker.

4.4 Trust Provisioning

Trust provisioning is the early phase in which the IoT device credentials are initially created and installed.

The fundamental provisioning process is composed of multiple steps, which are depicted in Figure 4. The process steps are conducted with dedicated personalization equipment and within a secured personalization environment:

- First, the private key is generated on the Trust X device (using the “generate key” command), and is never exported from the Trust X.
- Second, the public key is exported to personalization equipment, on which a certificate signing request is generated. A CSR must be signed by the corresponding private key, using the Trust X signature calculation command.
- Third, a certificate authority verifies the CSR. If the CSR satisfies the CA policy, and the public key matches the signature of the CSR, the CA issues a device certificate.
- Fourth and finally, the device certificate is installed on Trust X by the personalization equipment, and its lifecycle status is correctly set.

OPTIGA™ Trust X supports different scenarios depending on the IoT solution provider requirements. Personalization can be performed by Infineon or an Infineon distributor at a secured premises. Alternatively, the IoT solution provider can provision IoT device certificates at its own facilities, using a public key infrastructure fully managed by the solution provider.

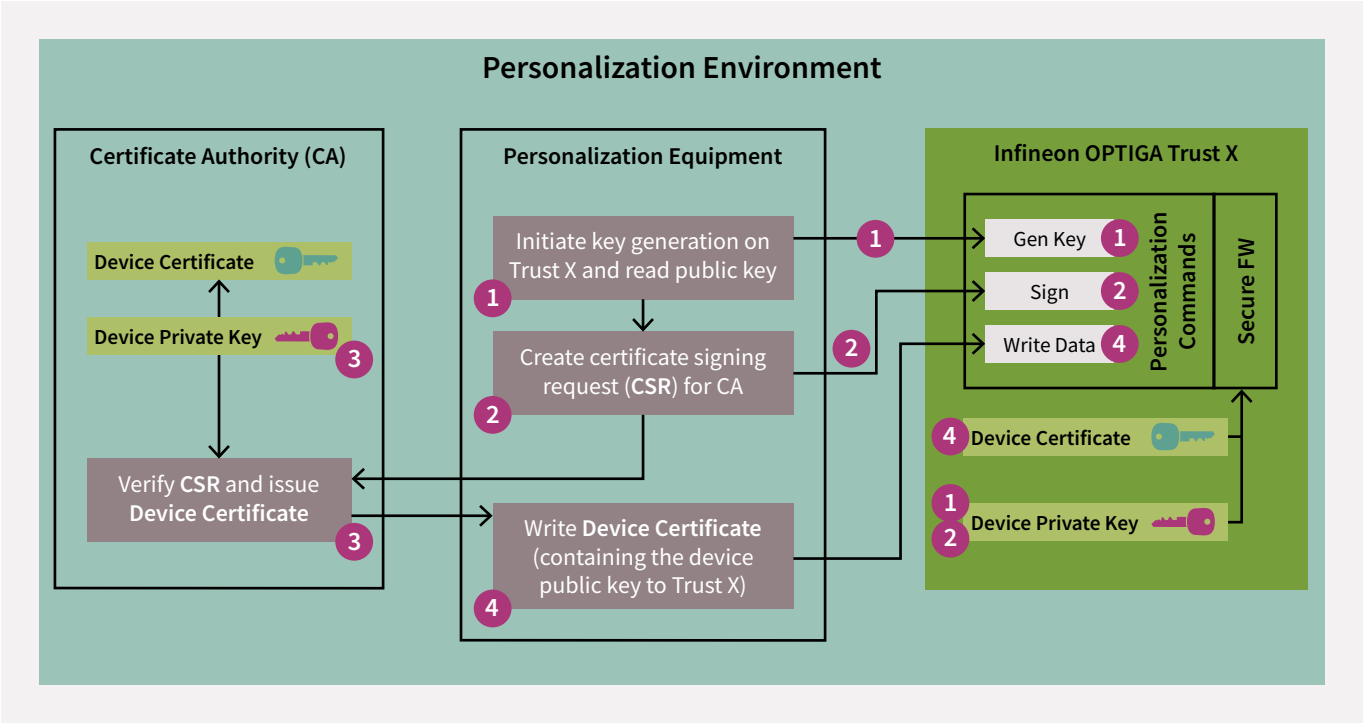

Figure 4: During the personalization process, the device private key is generated on Trust X, and the device certificate installed on Trust X.

4.5 Host Implementation Aspects

IoT solution developers can select the host controller integration method best suited to their performance, security and other requirements.

For TLS-based mutual authentication, the developer can choose from the following options:

- Signature calculation operation using the long-term private key
- Further cryptographic operations during the handshake, including ephemeral key generation, shared secret calculation using Diffie-Hellman and key derivation.

This flexibility also allows for a wide choice of potential host-side libraries to perform the TLS handshake and Record Protocol. On a Linux-based host controller, an OpenSSL engine allows a developer to easily delegate digital signature operations to hardware security modules such as OPTIGA™ Trust X. On an embedded platform, a TLS library with a smaller footprint such as mbed TLS could be used.

Alternatively, an IoT solution developer can also implement a proprietary host library for TLS that integrates with a subset or all of the OPTIGA™ Trust X toolbox commands.

Thus, the Trust X toolbox command scope can range from solely delegating the security-critical digital signature operation to conducting most TLS handshake operations on Trust X.

Depending on the specific host controller of the IoT device, OPTIGA™ Trust X not only enhances the security but also the performance of IoT devices. The ability to offload certain operations can reduce host controller memory requirements or increase the speed of cryptographic operations.

4.6 Device Enrollment Process with AWS IoT

The provisioning and enrollment process with AWS IoT can be conducted in different ways.

For explanatory purposes, the web-based AWS IoT console [7] can be used. However, this method does not scale well to hundreds or thousands of IoT devices. Hence, automated procedures are available:

- Programmatic provisioning and enrollment allows new devices to be registered via the AWS RegisterThings REST API. For scripting purposes, the AWS Command Line Interface (CLI) can be used on both Windows and Linux. This tool also accesses the AWS REST API.
- Alternatively, with just-in-time provisioning, IoT devices are provisioned when they first connect to AWS IoT. The solution developer needs to enable automatic registration and configure an associated provisioning template. Furthermore, the CA certificate of the to-be-provisioned devices has to be installed on AWS IoT.

Infineon OPTIGA™ Trust X supports both enrollment methods, because it allows the IoT solution developer to also use their own trust anchor. But it is also possible to realize a "zero-touch" enrollment process with just-in-time provisioning:

1. The IoT solution provider integrates pre-provisioned OPTIGA™ Trust X with their IoT device platform.
2. The IoT solution provider registers the CA certificate of pre-provisioned OPTIGA™ Trust X devices with their AWS IoT cloud service endpoint.
3. The first time each IoT device with OPTIGA™ Trust X connects to the AWS IoT cloud service in the field, it is securely enrolled into the AWS IoT cloud service endpoint and can subsequently be managed by the IoT solution provider via AWS IoT.

In such a zero-touch scenario, the solution provider does not need to manage their own certificate authority and does not have to operate their own CA.

4.7 Hardware-Security Beyond TLS and Secure Communication

This white paper has shown how the long-term cryptographic credentials for authentication and secured connectivity to AWS IoT can be secured.

Looking beyond authentication and secured connectivity, however, Infineon OPTIGA™ Trust X also offers a wide range of cryptographic services that protect other important security-relevant processes that are needed for IoT devices:

- Secured device firmware updates and secured software updates
- Secured boot, measured boot and platform integrity
- Native Datagram Transport Layer Security (DTLS) including Record Protocol layer implemented on Trust X

Please visit https://www.infineon.com/OPTIGA-Trust-X to get more information, white papers and application notes on OPTIGA™ Trust X.

5. Conclusion

Cloud services are an essential enabler for the IoT. Both protected communication and device authentication are of utmost importance to build trust in cloud services. These capabilities must be implemented from the very beginning and cover the complete product lifecycle.

This paper provided a comprehensive overview of an MQTT-based managed cloud service and its security features. It has also shown that hardware-based security significantly improves security while simultaneously offloading processing tasks from an IoT device's host processor.

AWS IoT provides a managed MQTT broker hosted on Amazon Web Services. With OPTIGA™ Trust X, Infineon protects AWS-connected IoT devices with hardware-based authentication and communication features.

OPTIGA™ Trust X is a feature-rich and flexible solution designed for ease of integration. Highlights include:

- Scope – from pure authentication primitives via TLS support functions to a complete DTLS protocol
- Flexibility – OPTIGA™ Trust X can be integrated with different host platforms and TLS libraries – from proprietary customer to proven open-source solutions
- Ease-of-use – Infineon provides I2C protocol, authentication and DTLS implementations to enable turnkey deployment

In conclusion, Infineon's OPTIGA™ Trust X enables the implementation of secured IoT devices offering tamper-resistant hardware combined with a tailored set of ready-to-use security functions.

6. Glossary

Amazon Web Services (AWS)	Collection of public cloud services offered by Amazon
Application Programming Interface (API)	Set of protocols and programmatic methods/functions to interface with a software or library
Certificate Authority (CA)	Entity that issues digital certificates
Certificate Signing Request (CSR)	Message sent from an applicant to a certificate authority in order to apply for a digital certificate
Datagram Transport Layer Security (DTLS)	Datagram-based version of TLS that runs on top of UDP
Hardware Security Module (HSM)	Discrete computing device that securely stores and processes digital keys and certificates
mbed TLS	Software library containing an open-source implementation of TLS and DTLS
Message Queuing Telemetry Transport (MQTT)	Publish-subscribe-based messaging protocol
OpenSSL	Software library containing an open-source implementation of TLS and DTLS
Software Development Kit (SDK)	Set of software development tools to create an application
toolbox	In the context of Infineon OPTIGA™ Trust X, toolbox refers to a set of cryptographic functions that can be used to implement TLS
Transport Layer Security (TLS)	Cryptographic protocol for secure Internet communication that runs on top of TCP
Trust anchor	In an X.509 architecture, a trust anchor is a root CA certificate, from which trust for derived certificates can be validated
User Datagram Protocol (UDP)	Transport-layer Internet protocol that provides no guarantees to the upper layer protocol for message delivery
Transmission Control Protocol (TCP)	Transport-layer Internet protocol that provides reliable, ordered, and error-checked delivery of a stream of octets between applications running on hosts communicating over an IP network
X.509	Standard that defines the format of public key certificates

7. References

[1] Amazon Still Leads Cloud Rankings, But Competition Is Coming On Strong. http://fortune.com/2017/06/15/gartner-cloud-rankings/ 2017-12-22

[2] The Transport Layer Security (TLS) Protocol Version 1.2. Dierks & Rescorla, IETF. https://tools.ietf.org/html/rfc5246 2017-12-27

[3] AWS IoT Device SDK for Python. https://github.com/aws/aws-iot-device-sdk-python 2017-12-27

[4] AWS IoT Device SDK for embedded C. https://github.com/aws/aws-iot-device-sdk-embedded-C 2018-01-22

[5] Mirai IoT Botnet Co-Authors Plead Guilty. https://krebsonsecurity.com/2017/12/mirai-iot-botnet-co-authors-plead-guilty/ 2018-01-22

[6] AWS Security Best Practices. August 2016. http://aws.amazon.com/security/ and https://d0.awsstatic.com/whitepapers/aws-security-best-practices.pdf 2018-01-22

[7] AWS IoT Console. https://console.aws.amazon.com/iot 2018-01-22