What is secure provisioning of an IoT device and why is it important?

Version 1.0

CRYPTO QUANTIQUE

Introduction

Within seconds of searching for “secure provisioning” on the web, you are presented with a myriad of descriptions covering multiple technologies, few of which seem to relate to the problems placed upon your classic embedded development engineer. So why should an embedded engineer be interested in secure provisioning as a service?

With the ever-increasing investment in engineering resources required to develop a connected product, it’s no surprise that managers are under pressure to find ways of protecting this investment. This pressure flows down to development engineers who are asked to come up with solutions to ensure companies’ valuable software is protected. There are many ways to protect software but a key factor, that is often overlooked, is the security of the manufacturing process that programs the software into the microcontroller. Can the process be trusted? How does “secure provisioning” come into the equation?

Some companies claim to have a secure process for programming semiconductors. They’re mostly similar and invariabley costly. In this paper we will highlight the approach taken by Crypto Quantique, a company that adopts a “zero trust” philosophy to crack the challenges of secure provisioning.

The role of encryption in IoT security

This is a complex subject but a key element in ensuring your software is protected and not visible to prying eyes. Encryption is often the first thing that comes to mind when security is raised. By encrypting software, you can transport it securely. For example, it is common for software images to be sent to remote programming houses to turn microcontrollers into useful products. But this is one point at which software is vulnerable to attack.

Without getting too bogged down in math, what does an embedded development engineer need to know? Well, there are two popular public key cryptographic techniques: Rivest, Shamir, & Adleman (RSA) and Elliptic Curve Cryptography (ECC).

If you're an embedded design engineer, you need to understand the microcontroller resources needed to implement these techniques. ECC is the clear winner because it features:

- Fast key generation
- Fast signatures
- Smaller key sizes for equivalent RSA security levels
- Good standards support
- Good key exchange mechanism

So, for ECC, it comes down to being fast and small, with short computation bursts and only small flash memory needed for key storage and transmission. As an example, a 256-bit ECC public key has approximately the same level of security as a 3074-bit RSA public key.

Ideally, any secure provisioning service should support the latest cryptographic technologies so as to maintain compatibility with the security architecture choices made by development engineers. There is often a requirement for a secure provisioning service to generate keys at the programming head in order to support injecting keys into microcontrollers.

But what threat to security does quantum computing pose? Quantum computing is coming and in theory, it will be able to crack today's cryptographic algorithms in an uncomfortably short time. No-one knows when quantum computers will become practical, it may be 5, 10 or 15 years yet. It's a little like nuclear fusion – every year the experts tell us it will be working in 20 years' time!

Unsurprisingly, the security industry is not standing still. The National Institute of Standards and Technology (NIST) has been working to ensure that post-quantum computing cryptographic algorithms are going to be available when needed. At the time of writing, round three of the submissions for algorithms is being evaluated. (see: CQ **Article**). The bottom line is that the security industry will be prepared when quantum computers become practical and the internet has to update all its algorithms. Some companies, like Crypto Quantique, are thinking ahead and providing technology that is algorithm-agnostic. The company has combined a novel method of generating a random identity in semiconductor hardware that is utilized by a security platform designed ready for the post-quantum era.

IoT devices can only be secured after a root-of-trust is established

A hardware root-of-trust (RoT) is the basis for ensuring security in an embedded system. It is based on the idea that an intelligent system has an immutable (non-interruptible) and repeatable sequence that must be executed during initialization. The RoT must include, but will not necessarily be limited to providing:

- A Trusted Execution Environment (TEE)
- Some cryptographic functions (AES, True Random Number Generator, PUF, key generation engine)
- Secure services available to the main application (signing, verification)

The TEE can be expanded to include hardware security measures specific to the microcontroller. Examples of these are Arm's TrustZone™, Renesas' Secure Crypto Engine (SCE) and Trusted Security IP (TSIP), Silicon Labs' Secure Vault, and STMicroelectronics' SFI & Firewall. With these technologies, cryptographic keys, along with other secret information, can be stored securely within the microcontroller. These form an important part of the RoT.

When an intelligent system includes a RoT, it can be guaranteed that the microcontroller or other semiconductor at the heart of the system will follow a set sequence of instructions on being released from reset (see Figure 1). For example, starting up a simple ROM-based boot loader that carries out a signature check on the main application prior to execution. There are many RoT implementations but a fundamental consideration is how the RoT gets programmed into the microcontroller? How secure is that process?

This is where secure provisioning comes to the fore. There are many security firms that purport to be able to secure your software, but they often depend on a RoT. This is typical in the mobile telecoms arena, where the RoT is usually based on a hardware TEE. Secure provisioning ensures that, for an embedded microcontroller-based system, the RoT is programmed (provisioned) into the microcontroller itself in a secure manner. This ensures the integrity of security functions used in the upper layers of the system that are dependent on the hardware RoT.

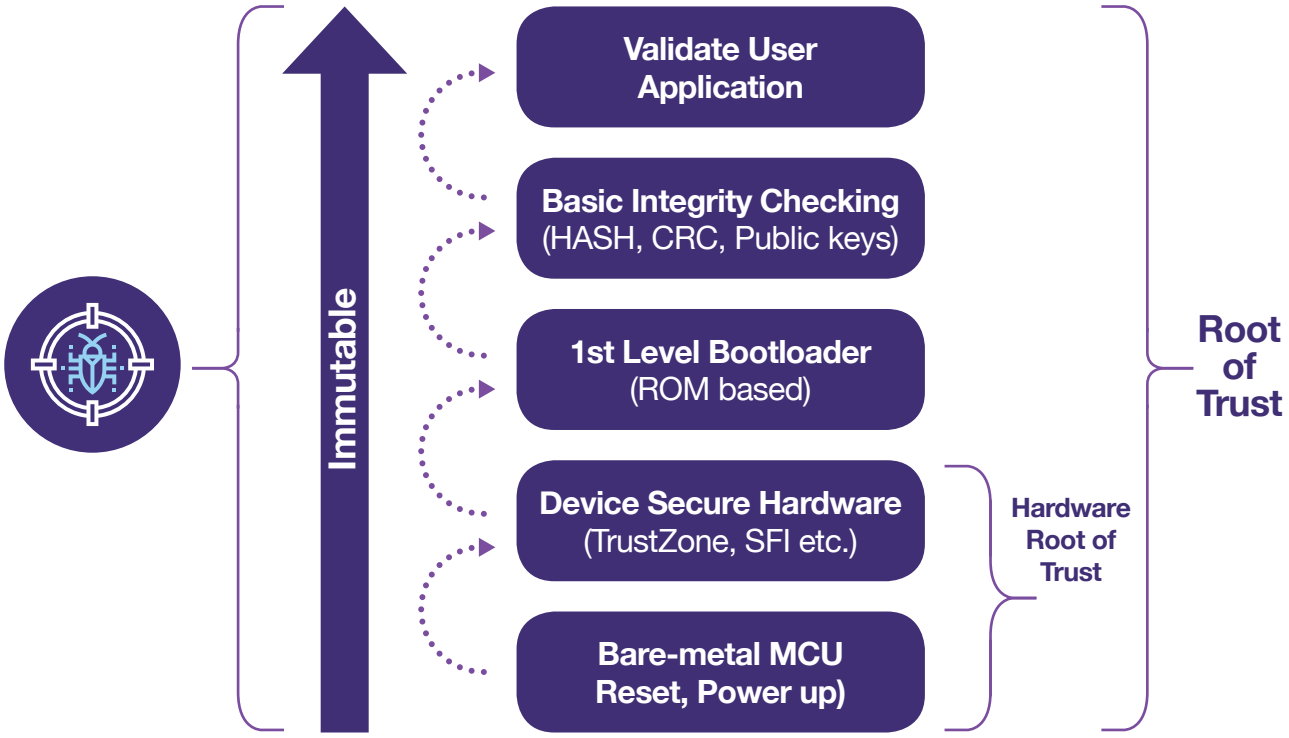


Figure 1: Root of Trust

The importance of identity, or fingerprinting

Public Key Infrastructure (PKI) is a standards-based security technology that has been used to secure network connected devices for some years. It covers many use cases and provides secure encrypted communications and mutual authentication between devices, services and users. Through the use of digital certificates every connected “thing” can be bound, by the use of public keys, to entities such as people and organizations. The use of these certificates creates a chain of trust between the connected device and the certificate authority that issued the certificates. However, in a similar way that the RoT is the basis of the security of a microcontroller system, a unique identity is the key to a valid PKI.

Here’s the problem. You can’t simply use the Unique ID, or even the UUID (Universally Unique ID) that is readable from most microcontrollers available today. This is not an identity, it’s only an identifier. An identifier cannot “prove” that it owns the device in question.

This is where cryptography and PKI has a role. For PKI to work you need a randomly generated cryptographic key pair to be created for every device manufactured. This key pair becomes the basis of each device’s identity. Typically, the key pair must be provisioned into the RoT of the microcontroller. The private key must be protected by the hardware RoT because this key is used to “prove” that the device is who it says it is. The public part is held in a certificate and both keys are provisioned during manufacturing.

There are some issues with this current approach:

1) In order to ensure that the key pair is generated from a truly random seed, a certified Hardware Security Module (HSM) is required.
2) An HSM is expensive.
3) An HSM is outside the MCU, therefore the key pair must be transported between the HSM and the MCU securely.
4) HSMs are typically not designed to be compatible with general programming hardware.

Let's revisit the issue of uniqueness and quantum computers. We have mentioned post-quantum algorithms - those algorithms that cannot be broken by a quantum computer, but there is also the issue of generating the "random seed".

Earlier, we said we needed a randomly generated key pair, but to be precise, we need to generate a random seed that is then used to generate a key pair. It turns out that creating a truly random seed is difficult. Typically, it is generated using an electrical circuit like a ring oscillator. The jitter in the circuit is the source of randomness, however, since jitter is the result of physical and electrical phenomena such as temperature, circuit thresholds etc., a quantum computer could theoretically predict the resultant random seed output and also the derived keys.

Therefore, the semiconductor industry needs a technology that generates a quantum-computer-safe random seed that is internal to the semiconductor device i.e. never leaves the silicon and has very high entropy (unpredictability). That's Crypto Quantique's QDID (Quantum-Driven IDentity).

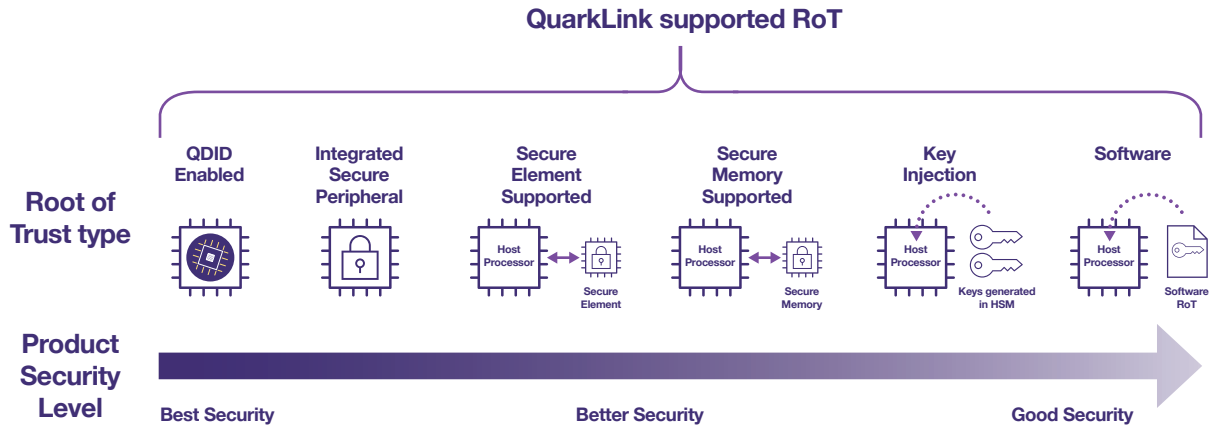
QDID: the quantum-driven hardware RoT

Crypto Quantique's QDID is a hard macro IP block that can be easily integrated into a standard CMOS process. The technology works on process nodes that are less than 90nm and has been ported to 55nm (Global Foundries) and 22nm (TSMC) processes. It can be used in microcontrollers, ASICs and SoCs. QDID takes advantage of the random nature of atomic positions and imperfections of nanostructures within the silicon oxide layer of a CMOS transistor. This has a direct effect on the level of quantum tunneling current within each transistor and measuring it produces a unique fingerprint. This fingerprint is baked hard into the structure of the silicon during manufacture and is there for the life of the device. It can be read at any time and does not need to be stored in secure memory. Also, the fingerprint never leaves the host device. It's the holy grail of security i.e., the fingerprint is truly random in nature (unique), it's hard-coded and can be used at anytime to generate a cryptographic key pair. As a bonus, because it's based on a quantum effect, its unforgeable, cannot be counterfeited and is immune to side channel attack.

So, for a QDID-enabled device you have no need for an HSM in the provisioning process since the keys are already in the silicon and they are safe.

Secure chip-to-cloud connectivity: the QuarkLink Security Platform

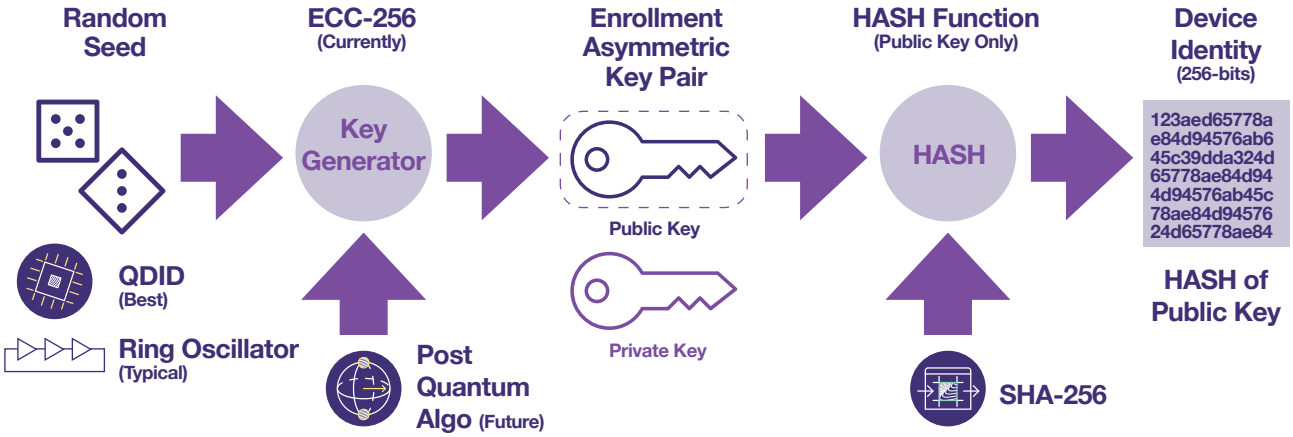
As mentioned previously, security platforms require a RoT to set up a secure communications channel with a networkconnected device. It is over this secure communications channel that device credentials, certificates and other security functions are executed. Crypto Quantique's security platform – QuarkLink - is a software-based IoT platform that securely connects devices to server-hosted apps on-premises or in the cloud. QuarkLink uses advanced cryptography to integrate with any RoT for end-to-end security across every IoT device. QuarkLink effectively secures any IoT system where devices have RoTs but provides the highest levels of security when it's combined with QDID.



Let's examine the requirements for the RoT to enable a connected device to communicate securely with QuarkLink.

When a device connects to a cloud service provider or web application, it will initially reach out to its assigned QuarkLink to receive the credentials needed to complete the onboarding process.

The process starts with the device connecting to a local network and initiating an HTTP connection request to its associated QuarkLink. The device will offer its identity to QuarkLink and will be challenged to determine whether it is the true owner of the identity submitted. But how is the identity bound to cryptographic keys in such a way that it can be proven? The diagram below shows how the identity is represented by a 256-bit HASH of the public key of an asymmetric key pair (called the enrollment key pair). The algorithm used to generate the enrollment key pair is agnostic to the process. Currently the keys are generated using an ECC-256 algorithm, but this will be updated with a NIST-compliant, quantum-safe algorithm after the standard has been agreed.



With this scheme, the receiver of the 256-bit identity value (the QuarkLink) can cryptographically verify it by running a HASH function on the public part of the enrollment key pair. This cryptographically links the identity value with the private key held by the device, which will be used in the challenge.

Once the identity has been proven, QuarkLink will set up a secure TLS communications link with the device. With the secure link running, QuarkLink can send all the required credentials, i.e. certificates and endpoint information, to the device. The device can then disconnect from the QuarkLink and onboard to its cloud service provider or web application using the credentials provided. In the meantime, the QuarkLink registers the device with the cloud service in the background. The onboarding process is shown in the diagram below.

QuarkLink Onboarding Process

The provisioning process in detail

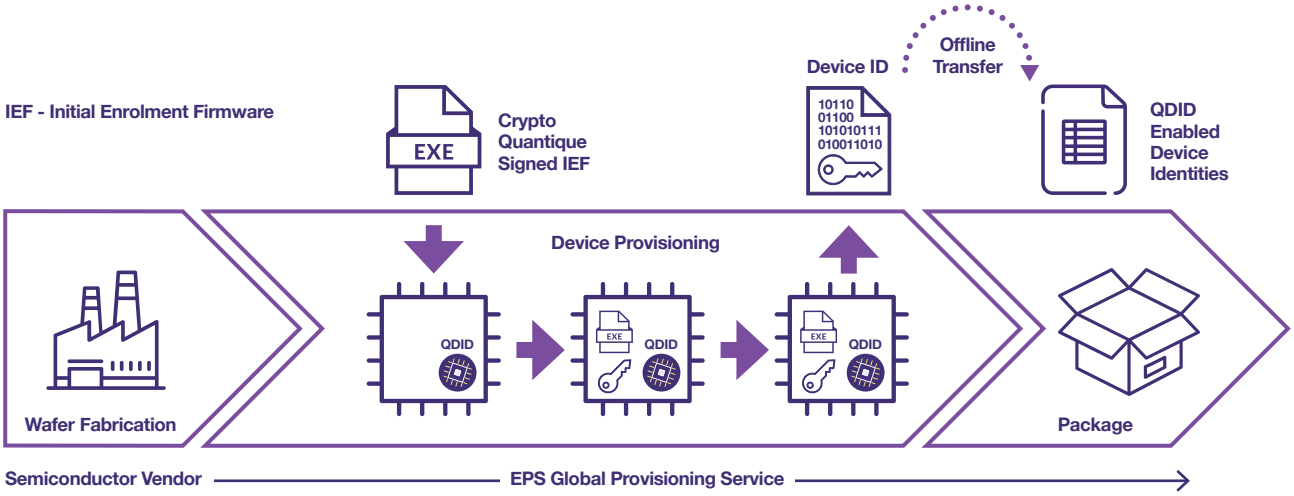
Now that we have seen the onboarding flow provided by QuarkLink, let's examine how a connected device transforms from an anonymous semiconductor to a client with a provable identity.

As discussed previously, the RoT is the foundation of security. Typically, a RoT is established during provisioning where a semiconductor device is programmed with cryptographic keys that have been generated by an HSM. The keys are programmed into secure memory, such as One Time Programmable (OTP) or other memory protected by a hardware security function e.g., TrustZone or SRAM PUF. Injecting or programming keys has security and cost implications due to the need for HSMs and a secure programming environment. Since QuarkLink is capable of utilizing many different types of RoT, we will only examine two primary examples where the provisioning process differs substantially.

QDID-enabled device provisioning

As mentioned previously, a semiconductor that has an integrated QDID IP block does not require the use of an HSM to generate keys and does not need to be provisioned in a secure facility since the RoT keys are already included in the QDID.

In the case of a QDID-enabled device, the keys associated with identity (enrolment key pair) can be generated, on demand, from a seed within the QDID, making it exceptionally secure. However, firmware must be programmed into the device to instruct QDID to generate the key pair and provide the public part to the outside world. This firmware is called the Initial Enrolment Firmware (IEF). This firmware is signed by Crypto Quantique and cryptographically verified by the device prior to execution. The IEF is programmed into the device using an approved programmer. The IEF is then executed by the device to allow the extraction of the 256-bit identity (see above). The private enrolment key is never exposed as it is baked into QDID during manufacture. This provisioning service does not require a secure facility since the IEF does not include sensitive information, it cannot be corrupted (since it is signed), and it only extracts public information such as the identity and public enrolment key from the device. The provisioning workflow is shown below.



Once the identities have been extracted, the devices can be repackaged and sent to the end customer for software programming. Alternatively, the end customer software can be programmed during the provisioning process alongside the IEF. The QDID-enabled device includes additional security features to allow programming of customer software in an encrypted form if the customer needs to protect their IP.

The QDID identities are later used by QuarkLink to verify that the connected device is a registered QDID-enabled semiconductor.

Non-QDID Enabled Device Provisioning

For this category we will look at two typical semiconductor types. Those devices that include integrated cryptographic key generation hardware/software and those that do not.

Where a Key Generation Function available

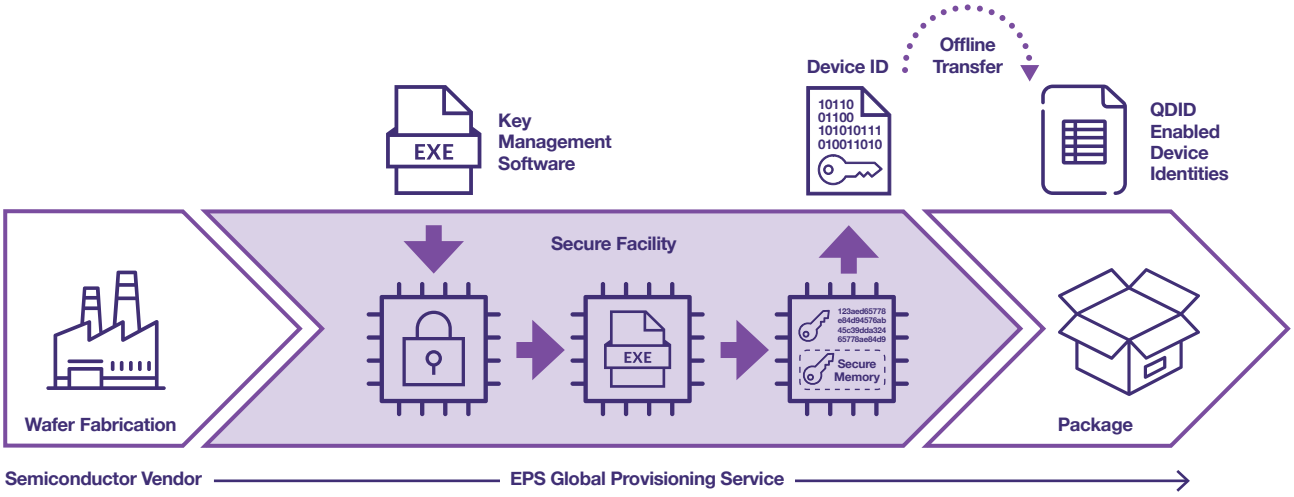
This type of device is the more secure of the two types discussed here. It's similar to the QDID-enabled device in that the private key never leaves the semiconductor. This is possible if the semiconductor includes key generation functions based on an alternative seed generation mechanism e.g., a ring oscillator. This is less secure than a QDID-enabled device for the following reasons:

1) The seed used to generate the keys is not quantum-safe.
2) The seed generation hardware/software is susceptible to side channel attack.
3) There is usually only one seed available for key generation.
4) Identity extraction firmware (equivalent to IEF) requires programming in a secure facility.

Firmware needs to be programmed into the semiconductor during provisioning to instruct the security functions to generate cryptographic keys and provide an identity. Secure microcontrollers such as the Renesas RX (Trusted Secure IP) and RA (Secure Crypto Engine) devices and Silicon Labs' (Secure Vault) microcontrollers include key generation hardware. For these devices, a secure facility is used to program the initial provisioning software.

The software is then used to:

1) Command the secure hardware to generate cryptographic keys.
2) Configure any secure hardware to protect the cryptographic keys using available memory protection mechanisms e.g., TrustZone, Memory Protection Unit (MPU), Flash Access Window (FAW).
3) Secure Bootloader management.

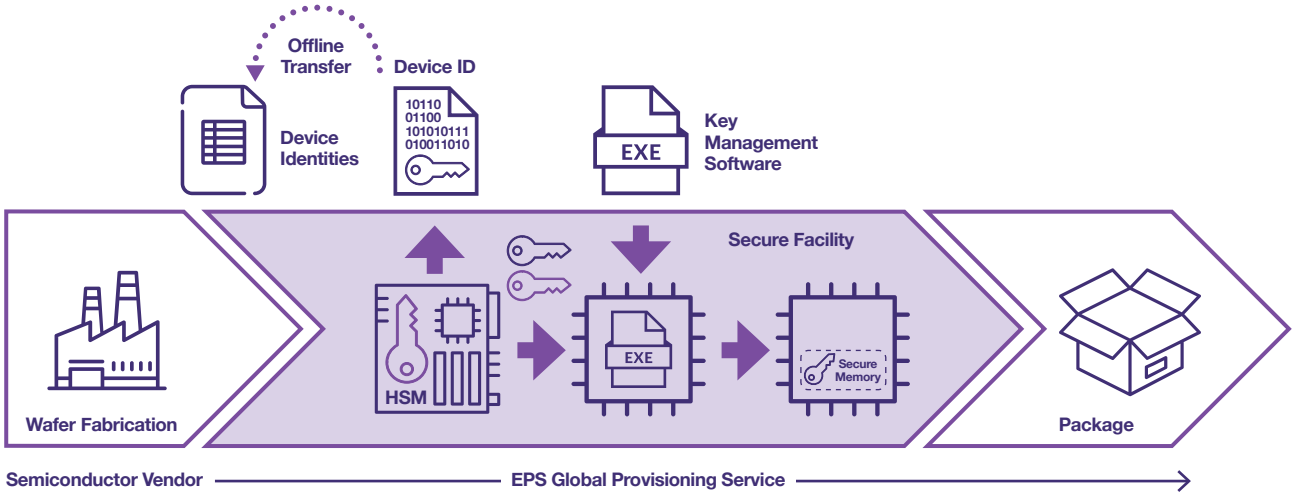


Where a Key Generation Function is not available

This type of semiconductor is less secure. It requires the cryptographic keys to be generated externally and be injected into secure memory. A major weakness is that private keys generated externally are transferred over the device programming interface, which is usually not encrypted. There are newer secure microcontrollers that have resolved this issue by providing encrypted bootloader interfaces and this improve security considerably. However, these devices suffer from the following disadvantages:

1) An external HSM and associated control software is required for key generation.
2) Typically, all keys are transferred from the HSM to the device via a non-encrypted programming interface e.g., JTAG, Single Wire Debug (SWD)
3) Additional costs are incurred due to the need for an HSM and compatible programmer.
4) A secure facility is needed in to house the HSM and programmer.

Firmware and keys need to be programmed into this type of semiconductor during provisioning. The firmware will instruct the security functions to store the keys securely and protect them from unauthorized access. Identity values can be generated in the HSM prior to injection of the asymmetric key pairs.



Conclusion: QDID + QuarkLink means greater security, scalability and cost-effectiveness

Today's microcontrollers are slowly improving their security capabilities. Legislation is helping drive the adoption of secure best practices and this is filtering down to semiconductor vendors, driving them to provide engineers with advanced security peripherals. What's more, the need for quantum-safe security functions is likely to arise within the operating life of devices being deployed today.

Adopting advanced cryptographic hardware such as QDID greatly simplifies and lowers the costs of device provisioning. Crucially, QDID also provides OEMs with devices that can be updated to be quantum-safe via Over-the-Air (OTA) software updates to replace existing cryptographic algorithms with NIST compliant post-quantum algorithms.

Both QDID-enabled devices and those that have advanced key generation hardware, greatly improve product security. Both ensure that private keys never leave the device and eliminate the need for an external HSM.

Secure provisioning service providers, such as EPS Global, have a wealth of knowledge when it comes to dealing with the programming of secure devices. They have invested in technologies, both software and hardware, that fully utilize the capabilities of secure devices. This provides product developers with confidence that their software IP and RoTs are secure and that the latest key provisioning techniques are implemented to ensure minimal exposure of cryptographic keys. A combination of EPS Global's expertise in secure provisioning, Crypto Quantique's quantum-safe hardware IP and the QuarkLink security platform provides customers with easy-to-use, scalable IoT security from semiconductor device to cloud service.

Contact us

For more information please see **www.cryptoquantique.com**
or contact **info@cryptoquantique.com**