SCTE·ISBE CABLE-TEC EXPO 2018 ATLANTA, GA OCTOBER 22-25

2018 Fall Technical Forum SCTE·ISBE • NCTA • CABLELABS

Internet of Things Security: Implement a Strong, Simple & Massively Scalable Solution

A Technical Paper prepared for SCTE•ISBE by

Ron Ih
Director of Business Development, Kyrio Security Solutions
Kyrio
r.ih@kyrio.com

Table of Contents

List of Figures

1. Executive Summary

The industry has been chasing its tail for the past 5 to 7 years on the issue of Internet of Things (IoT) security, and it seems every week brings a new article about the need for device security or details about yet another security vulnerability exploit. The Internet is teeming with articles about issues and after-the-fact bandages, but very few of them get to the heart of the problem, which is how to secure network ecosystems that include interoperable autonomous devices.

IoT adoption continues to grow—but at the expense of good network and cybersecurity practices. Industrial and commercial IoT had previously been characterized by isolated networks that allowed devices within the network to communicate, but there was no connection to the outside world. In these use cases, it was possible to get away with weak security because it was more difficult to execute a wide-scale attack from the Internet. However, there is growing demand by utilities and builders to enable external communication and control of commercial devices to improve energy efficiency and provide better power grid management. This requires that commercial, industrial and even residential IoT devices be connected to the Internet so that they can be reached by utilities and state energy regulators. These would include lighting control systems, smart meters, solar inverters and home appliances. In fact, network connectivity is already starting to be mandated in some states (e.g., California Rule 21).

However, as critical electric power infrastructure is being network-connected, there needs to be an economical solution that adequately addresses security concerns as well as the logistics surrounding its implementation.

Companies that can provide strong security at scale will be able to use that as a key differentiator for their products, protect their brand and future-proof their products— which can have lifespans of 10 to 20 years or more— as calls for stricter requirements regarding device security loom on the horizon. Even as more wired control systems get connected, wirelessly connected devices are seeing exploding growth. Wireless devices are much easier to install and often reduce deployment time from several weeks to just a few days— or even hours. Easier installation reduces the amount of time installers need to spend on a job, thus reducing costs and increasing revenue by enabling them to do more jobs in the same amount of time. However, expanded wired and wireless connectivity accelerates the need for a more scalable security solution for these networked devices.

This paper covers the fundamentals of security architecture, best practices and new processes that can vastly simplify the implementation of strong, enterprise-grade security into small resource-constrained IoT devices. The goal is to enable deployment of security on the massive scale needed for IoT, while not sacrificing security robustness, and provide a workflow that can be implemented in hardware across a highly fragmented, embedded system.

This paper will not cover any hacks or exploits; those have been covered quite sufficiently to date. What is needed are more articles that cover the “how” of IoT security, not just further descriptions of new problems. In the new IoT reality, users need to know how to apply security that is strong, simple and massively scalable to tens of billions of hardware devices.

2. Scope of the Problem

The first question that is often asked is: "Why bother? A small IoT device possesses limited data and limited capabilities, so why is it worth securing?" Consider the situation from another perspective. When you are issued an access badge at your company, is the company securing you as a person? No, the company is securing access to its assets. This security measure is a way to control access based on the verification of your identity. The same concept applies to the IoT device: It should be verified before it is permitted access to the network. The device itself is not nearly as important as what the device potentially has access to. Just as unverified people should not be allowed to wander through secure buildings, unverified IoT devices should not be allowed access to your networks. With advances in technology and new logistical processes, it is now possible to provide manageable secure identities to IoT devices by default on a massive scale.

Resolving the issue of IoT security has been a complex problem to address. The reason the problem remains is that people have been trying to address it within the limited scope of their own market position and place in the value chain. The importance of IoT security spans multiple interrelated but very different market constituents. Before we can understand how to solve the overall problem, we need to understand each part of the IoT value chain and the respective concerns and issues surrounding cybersecurity implementation.

2.1 Device Manufacturers

General-purpose devices such as PCs, mobile phones and servers are what consumers are accustomed to thinking about when it comes to computer hardware. These devices have large processors, contain lots of memory and storage, can execute many types of software applications and can perform many different tasks.

Embedded systems, by contrast, are purpose-built to perform a specific task. Examples of embedded systems are lighting control systems, Wi-Fi-enabled thermostats and networked security cameras. Each of these systems has a microprocessor, memory and storage, but they are generally much smaller and optimized to perform specific tasks and no more.

Most of the IoT devices hitting the market are small embedded systems that have microcontroller-class processors with far more limited compute power and resources available to them. These devices include temperature sensors, light switches, cameras and so on.

For the most part, manufacturers of these devices use network connectivity in their products to improve functionality, features and ease-of-use/installation. Network connectivity allows manufacturers to build value with improved functionality, and wireless network connectivity allows further enhancement by simplifying deployment.

Cost reductions and advanced miniaturization have made it technically possible and economically feasible to network-connect very small devices and even put wireless radios in the smallest, simplest devices. However, device network connectivity (as well as Internet connectivity in itself) requires greater emphasis on security to control access to the networks and ecosystems those devices are connected to. Wireless-enabled IoT further compounds the need for scalable security because the ease of connectivity and deployment of such devices makes them not only abundant but easily accessible.

The problem is that most of these companies typically do not have or cannot afford a dedicated team of cybersecurity specialists. So far, companies have not been held accountable for producing IoT products with inadequate security, but that is changing with greater government scrutiny.

The final link in the device manufacturer chain consists of those involved with the installation of these devices. Whether they are professional installers on commercial construction sites or end users in their homes, IoT devices are deployed by the thousands every day. Device authentication to services needs to move beyond the username-and-password paradigm that assumes every machine has a human being behind it. In addition to being a weak form of security, usernames and passwords are not scalable. They work adequately for 10 devices, but they do not work for a company that requires stronger cybersecurity and will ship 10 million devices to a global sales and installation channel.

Thus, IoT device makers need a security solution that is inexpensive on a per-unit basis, uses minimal computational resources in the device and does not require a cybersecurity specialist to implement. Most IoT devices will use small microcontrollers that do not have a lot of compute power, and it does not make economic sense to put a large System on a Chip (SoC) in place merely to crunch cryptographic math for an operation that is only used to establish the authenticated secure session. How can you accomplish strong cryptographic security using a small, inexpensive microcontroller that doesn't require a cybersecurity expert to implement and doesn't resort to using weak aftermarket network security?

It turns out that there are solutions on the market that can provide strong, scalable device security for a broad range of devices with a price point of less than $1 in moderate volumes.

2.2 Semiconductor Manufacturers

The use of secure element chips is becoming more prevalent in the market. These application-optimized cryptographic chips provide a pre-packaged solution for securely storing private keys and they also provide crypto-math acceleration, which is very useful when used in conjunction with small, low-power microcontrollers.

However, an issue arises when promoting these chips to the embedded systems companies that design and manufacture the devices. When the time comes to complete the sale, the companies face the issue of setting up Public Key Infrastructure (PKI). Specifically, the challenge is establishing the chain of authority for the certificates and the associated cryptographic keys that are provisioned into the secure elements. Today, it is up to the device manufacturer to find a PKI provider and to define the security requirements around the company's security domain and its policies. It requires the chip manufacturer's customer to be quite knowledgeable about cybersecurity and cryptography. On top of that, it is up to the customer to coordinate the deployment of PKI with the manufacturing flow of the secure elements. This all adds complexity and friction to the sales and design processes for a secure chip manufacturer.

Chip manufacturers need a security infrastructure that is well integrated into their production flow and that also abstracts from their customers most or all of the technical complexities behind establishing a certificate chain. Their customers need something that is as simple as adding a component to their bill of materials (BOM).

2.3 Cloud Service Providers

Due to the massive scale of device management for IoT product lines, most device vendors will use some form of IoT cloud-based management service (e.g., Microsoft Azure, Google Cloud, Amazon Web Services IoT). However, cloud service providers need a scalable provisioning process so that:

- Devices authenticate with the cloud provider's servers to prove that a given device originates from an authorized supplier and establish secure Transport Layer Security (TLS) communication sessions.

- Devices are automatically assigned to the proper company accounts. This must be able to work for thousands of manufacturers, each with many product lines, and must happen in a way that is seamless to the end user.

Cloud service providers have a similar problem as chip manufacturers because their customers are device manufacturers who need a simple, scalable way to strongly authenticate devices from many different manufacturers to their online services. These providers need a security solution that coordinates between the device makers, chip manufacturers and the cloud provider's online authentication processes.

2.4 Security Infrastructure Providers

Elliptic Curve Cryptography (ECC) is rapidly becoming the algorithm of choice for IoT because of the smaller key sizes compared with RSA. A 256-bit key in ECC is of roughly equivalent cryptographic strength to a 2048-bit RSA key. This makes ECC a more practical way to implement strong asymmetric authentication in small networked devices. Asymmetric authentication— arranged in a hierarchy known as PKI— is the framework on which managed secure infrastructure is constructed. As discussed in more detail later, authentication is the cornerstone of network security because it must be able to identify whom you are communicating with. If you cannot verify who you are sending data to, nothing else really matters.

PKI is the cornerstone of most enterprise cybersecurity plans, but for IoT it needs to be repackaged so that it fits seamlessly into the IoT hardware supply chain. In addition, the packaging needs to make PKI simple enough for implementation by device manufacturers who may have limited or no knowledge of cryptography, while not weakening its underlying secure authentication processes.

PKI has been in use for about 30 years in various forms, beginning with RSA in the 1980s and recently migrating to ECC. PKI has many advantages:

- It is highly scalable because credentials/certificates are digitally signed by an authority that has very rigidly controlled access. Any credential signed by an authority's private key can be validated using the authority's public key. As a result, millions of device credentials can be validated using a single key, which makes key management very simple.

- Even though a single public key is used to validate many credentials, each credential is unique. This allows for very granular management of devices because each one can be uniquely identified.

- PKI has been rigorously tested over the decades and is the basis of security used in most enterprise datacenters and servers that house sensitive data. It is not perfect, but it is robust enough for the vast majority of applications including IoT.

For all of its benefits, PKI's main weakness is the cost and complexity of its deployment. PKI was originally deployed in enterprise software environments where customers typically had security teams that understood network security and how to properly employ it. In addition, because deployments were in server and browser software applications, there has been substantially more flexibility in making changes or applying updates and patches, compared with deployments in embedded systems.

Finally, in most web browser applications, there was no need to authenticate the computer/device itself; users needed only to authenticate the server. As a result, most network sessions were only authenticated one way because the device didn't matter. It was the user that mattered. So, users authenticated themselves in the web application using their username and password to verify their identity.

For IoT, the identity of the device matters and the deployment model needs to adapt to that.

2.5 Scope of the Problem—Summary

The challenge is that for IoT, typically no active user is behind the device, unlike with PCs and mobile phones. The device logs in on its own and sends data on its own. For all intents and purposes, today an IoT device is a user on the network and now the task of authenticating the device itself becomes a concern.

The situation demands a practical and economical way to deliver private keys and certificates that belong to hundreds of PKI domains and thousands of manufacturers making billions of devices. This is something that PKI can do in theory but is not something that has been done scalably in practice. Therefore, we need to consider designing in a verifiably authentic digital identity as a basic capability in networked devices.

3. Understanding the Components of the Solution

Properly addressing the IoT security issue requires addressing the needs of each of the constituents in the supply chain. Otherwise, the solution will likely not be adopted or fit in the deployment flow. If a solution addresses only a specific constituent of the value chain, it is easier to deploy because it can be done independently— but is also likely to not be as effective because the other parts of the value chain are not cooperating in the solution. To properly address the IoT security problem, the different parts of the value chain must collaborate and coordinate with a solution that is continuous throughout the entire design and production flow.

3.1 Foundations of Cybersecurity

For many, particularly those not involved directly with cybersecurity, the popular term that comes to mind when the word "security" is mentioned is encryption. Although encryption is certainly necessary, it is by itself *insufficient* to provide a proper level of meaningful security in network communications. If encryption does not provide sufficient security, what does?

3.1.1 Encryption vs. Authentication

First, we need to understand that there are several components comprising proper security protocol and each of those components plays a key function in the process.

Encryption is an important part of any security solution, but it has a specific purpose: to prevent eavesdropping on transmissions. If you send data from one place to another, you do not want unauthorized people to intercept and read that data in transit. Modern encryption schemes, known as ciphers, are quite strong and effective at preventing interception of data transmissions. However, encryption by itself lacks a critical capability: It is unable to verify *whom* you are communicating with.

If you cannot verify whom you are sending data to, encryption by itself becomes less meaningful. Even though people cannot eavesdrop on your transmissions, you may be communicating with one of the entities you were trying to avoid in the first place because their identity cannot be established with reasonable certainty.

The ability to exchange verifiable credentials and validate them is known as *authentication*. The two most common forms of cryptographic authentication are symmetric and asymmetric authentication.

Symmetric authentication is fairly simple: The sender and receiver share the same key, and you can do a relatively simple random number challenge (or *nonce*) to determine whether the entity you are communicating with has the same key without actually transmitting any keys between you. Thus, for every entity you want to authenticate with, you have a unique symmetric key.

The issue with symmetric authentication is that scalability becomes an issue when the size of the ecosystem becomes very large. If an ecosystem contains millions of members, you will need something that manages millions of keys, and soon key management becomes a complex and costly system to maintain. In addition, key security must be maintained by both the sending and receiving parties. This is further complicated by the problem of securely provisioning keys on both sides— especially if devices are introduced to the ecosystem at different times and you need to authenticate new devices with older ones that are already deployed. If either side leaks its private key, the identity is compromised. If you add the complexity of a multi-vendor open ecosystem, the issue gets even more difficult.

Asymmetric authentication, as the name implies, involves a system in which the sender and receiver have different keys. The two keys are mathematically related to each other and are generated in pairs. The private key is protected and is used to "sign" digital data. The public key is often included in what is called a digital certificate and is used to verify the signature on its certificate. In the case of asymmetric authentication, only the signing (private) key needs to be protected. This accomplishes authentication in which the public key verifies that the device holding the paired private key sent the message. The strength of a public key cryptography system depends on the impractical amount of computation required to derive the private key from its paired public key. This means that effective security only requires keeping the private key private; the public key can be openly distributed without compromising security.

Figure 1 - Asymmetric Cryptography

This scheme also has the critical benefit that devices can easily be added at different times by different suppliers. Because the public keys can be openly distributed, enabling new devices to authenticate with devices that have already been deployed is easy.

The operation of asymmetric authentication involves three main operations: hash, sign and verify. A cryptographic hash is a one-way algorithm that produces a 256-bit number (in the case of SHA-256) that is consistent but not reversible. If you put in the same data, it always comes out with the same number. However, it is computationally infeasible to work backwards to determine the original data from the hash alone. So, to prove that data has not been altered, you provide someone with both the data and its hash. That person re-runs the hash with the data, and the results should match if the data has not been altered.

The sign operation is performed with the private key. Assuming that the private key is protected and has restricted access, data/hashes signed by a private key can originate only from an authorized source with access to that private key. So, a successful verification by the public key associated with that private key ensures the origin of that data. The combination of hashing and signing data forms the certificate. The creation and verification process for a digital certificate are shown below.

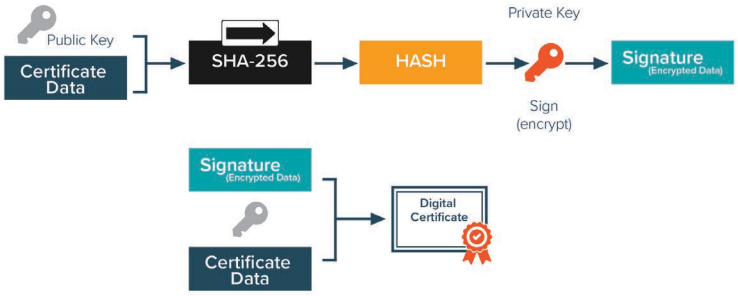

Figure 2 - Creation and Verification Process for a Digital Certificate

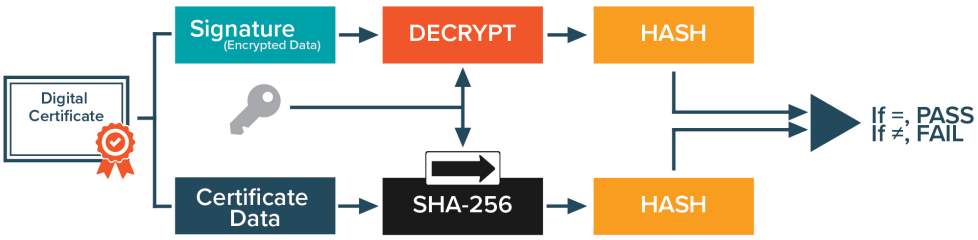

Figure 3 - Digital Certificate Authentication

A common scenario is to use asymmetric private keys as signing or "certificate authorities" (CAs) and arrange them into a hierarchy that generates, signs and organizes digital certificates. This security hierarchical structure is referred to as PKI. The CA certifies ownership of the key pairs and provides proof that a certain public key is authentic, belongs to the entity claimed and has not been tampered with.

3.1.2 Public Key Infrastructure (PKI)

As mentioned in the previous section, asymmetric authentication is often arranged in a hierarchy of CAs that sign and issue digital certificates/credentials. An example of a PKI hierarchy is shown below:

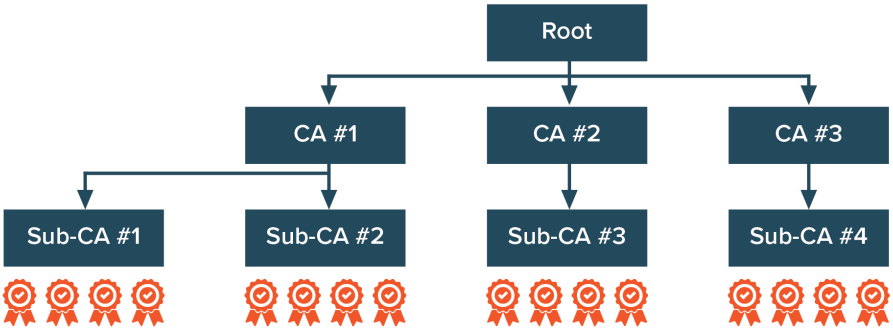

Figure 4 - Example Public Key Infrastructure Hierarchy

Each CA grants authority to sub-CAs, which then sign digital certificates for devices. The digital certificates at the bottom are carried by end devices and are authorized by the sub-CA above them that generated and signed them. These are generally called *device certificates*.

The sub-CAs that generate the device certificates possess their own certificate, authorized by the digital signature of CA above them, and so on. PKI eventually terminates at the root, which is the foundation on which this particular PKI ecosystem domain is constructed.

The reason PKI ecosystems are arranged in hierarchies is that it allows for selective levels of revocation or access denial if it is later determined that a leak or compromise of a private key in the ecosystem has occurred, as shown below:

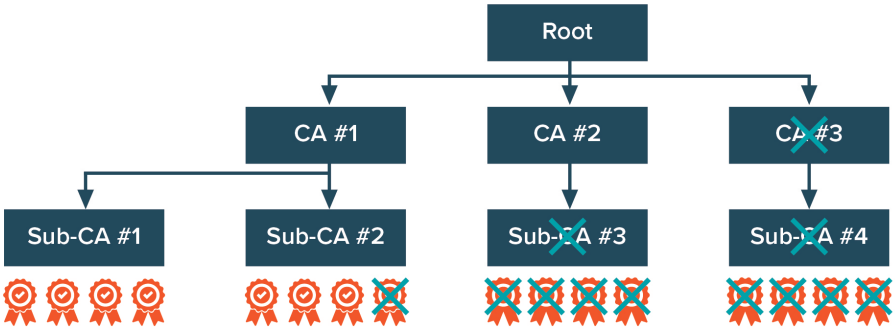

Figure 5 - Example Revocation in a PKI Ecosystem

As the figure shows, you can revoke the certificate of any element within PKI from the device up to a high-level CA, depending on the nature of the security compromise. However, it should be noted that if you revoke a certificate, you also revoke anything below that element in the hierarchy. It is possible to specify the revocation to be effective after a specific date noted in a PKI certificate, or be fully retroactive. It should also be noted that revocations cannot be undone— so they should not be initiated lightly.

This illustrates why PKI implementations are set up in tree-like hierarchies: This design allows for the ecosystem owner to perform selective damage control in the event of a compromise. It is also the reason why it is not a good practice to issue any device certificates off the root CA, which limits flexibility. If something goes wrong in that case, you may need to invalidate and revoke the entire PKI and all deployed devices in the field, which is generally not a good option. This is why device certificates are almost always issued by sub-CAs below the root.

Using this hierarchical structure enables the recipient of a digital device certificate to validate its pedigree or "chain of trust." To do this, you use the public keys recursively up the chain to verify each signing authority above the device certificate to validate the device's origin and the validity of its certificate. This is known as a chain validation. To avoid excessive computational requirements and time, it is recommended to not make PKI hierarchies too deep, because each level of validation requires additional certificate storage as well as computation.

The final and potentially largest benefit is that with a single sub-CA generating potentially millions of device certificates, you can authenticate millions of devices with one sub-CA public key, making key management far more scalable and manageable than other options.

If properly designed, PKI can be a compelling solution for IoT security. PKI is highly scalable and allows management of credentials and access control, so why hasn't it been adopted more widely for IoT?

3.2 Considerations for PKI

Although PKI certainly has many attractive aspects, it still has some considerations that need to be properly addressed for it to be an acceptable security solution for IoT devices. Below are a few complications with PKI as it relates to IoT applications in hardware and large-scale device deployments.

3.2.1 Infrastructure Complexity

One of the main drawbacks of PKI and asymmetric cryptography is the solution's complexity and the relatively intensive mathematical computations needed to perform operations.

When PKI was first developed, its target applications were servers, datacenters and web hosting. In these environments, security was implemented in software and companies that used PKI typically had a dedicated team of experts.

Setting up a PKI requires that the customer understands how to define the architecture needed based on the use cases. In addition, the customer needs to define the format of its certificates and the security policy around the management and protection of the PKI ecosystem. Next, the customer needs to implement the PKI hierarchy (or hire contractors to do it) and make sure that all the proper security protocols and cipher suite modules are used. Finally, because the private signing keys are the authorities that enable access to the ecosystem, they need to be securely hosted and periodically audited for policy compliance. Even with a sophisticated customer audience, the complexity of PKI makes implementation from scratch non-trivial and costly in both time and money.

3.2.2 Private Key Generation and Storage

Although asymmetric cryptography requires the protection of only the signing private key, not the verifying public key, any entity that needs to prove (authenticate) its identity needs to hold a private key.

A digital certificate is similar to a passport in that they both include two critical authentication components:

- Something that proves the credential originated from an authorized source and was not altered
- Something that proves the credential actually belongs to the bearer

On a passport, the embedded holograms, graphics and other physical security features make it very difficult to fake and alter the document. Examining the physical security features helps prove that the document is authentic and originated from the Government. The picture on the passport proves that the bearer of the passport is the owner.

For digital certificates, the asymmetric cryptographic signature on the certificate allows the recipient to use available public keys to verify the origin of the certificate and also prove that the data within it has not been modified. It proves authenticity and is akin to a passport's holographic security features.

The digital certificate contains a public key in addition to its certificate data. The public key is mathematically related to a specific private key. So, when a digital certificate is presented, the public/private key pair relationship proves that the device that presented the certificate actually possesses the unique private key associated with that certificate. By doing what's called a random number challenge, the recipient of the certificate gives the sender a number to hash and sign with its private key and send it back. If that private key corresponds to the public key in its certificate, then the recipient can decrypt the response using the public key and compare it to its own hashed number. If the two hashes match, the authentication passes. If not, the authentication fails.

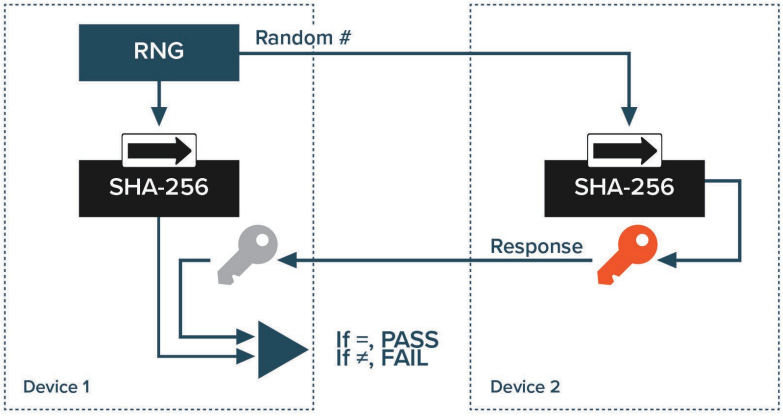

Figure 6 - Public/Private Key Authentication Example

Through this example, you can see why protecting any private key is vitally important— because it is used to prove rightful ownership of a digital certificate. Every member of an ecosystem needs a certificate to gain access, and therefore every member of an ecosystem needs a private key that is well protected. Consider the signature stamp of the company CEO: If that gets out, anyone can start signing things on behalf of the CEO and you could not tell which signatures are real or fake because you would not be able to determine the rightful owner.

Furthermore, the generation/creation of the public/private key pairs needs to be tightly controlled to prevent unauthorized access or leakage of private keys. In the best case, private keys are generated within a secure storage chip and are never exposed outside that device.

In the days when only servers hosted in protected datacenters needed to authenticate themselves, key generation and storage was typically performed in software because those systems were generally not physically accessible and (if properly secured) did not permit access to protected key storage.

In addition, the scale was quite different: The number of websites at the time was relatively small and easier to manage. Even today, there are approximately 1.8 billion websites in total around the world, although only about 250 million of them are active. That number may seem large, but the scale of IoT and small embedded systems is measured in billions of microcontrollers sold per year. The IoT future will involve generating and properly storing private keys on a scale at least an order of magnitude larger than web server certificates.

The current technology of PKI can theoretically support that future, but the logistics of making it happen at that scale is a whole new problem. IoT devices are meant to be deployed in the field and in places that are physically accessible to the public. Yet, the devices must hold a private key so that they can authenticate themselves to gateways, servers and cloud service software. For PKI to work for IoT, it needs a method to provide strong protection for private keys stored in small devices that have limited compute power— and be able to do it economically on a massive scale.

4. Repackaging as a Scalable Security Solution for IoT

The industry needs a solution that can provide the security robustness of PKI but that abstracts away the technical and logistical complexities. Most previous attempts at this were done by companies trying to address the security issue from within their own market vertical, with limited collaboration across the separate parts of the supply chain. As a result, these solutions are only partially successful at best.

4.1 Reducing PKI Implementation Complexity

Although a typical PKI implementation has its own certificate policy, profile and other security specifications unique to the requirements of that ecosystem, the reality is that most IoT applications do not require this level of customization.

Enterprise PKI solutions tend to provide the “Cadillac” version of software and systems, which is highly customized and tuned to a specific client’s needs. When you are dealing with a multi-billion dollar bank or global corporation, that makes sense. However, for IoT, that approach is neither economically or technically feasible. Small devices are highly cost-sensitive and require any security solution to fit into their manufacturing flow— not vice versa.

As such, a pre-established PKI that encompasses a multi-vendor ecosystem is more appropriate. In this case, a standard is established that covers the needs of a broad range of adopters. A PKI implementation is created based upon this standard, and all adopting manufacturers simply need to use certificates issued from that PKI. This creates much greater efficiencies and economies of scale because the costs of creating the PKI implementation are spread out among many companies who share the benefits of the common ecosystem PKI.

4.2 Certificates in Secure Chips

Advances in semiconductor miniaturization have made it possible to create chips specialized in cryptographic key storage and mathematical operations. In addition to being physically very small to save

circuit board space, they include physical and electronic protection mechanisms to prevent unauthorized access to private keys. They are commonly referred to as secure elements, although the term is something of a misnomer because they often perform many functions and are more like a cryptographic co-processor.

Private keys stored in secure elements are literally inaccessible once locked. You cannot electronically access the memory slots, and the chips themselves include features such as extra metal layers that prevent chip de-capping and micro-probing of memory cells to extract keys.

The co-processing capability is critical for small IoT devices because the math behind cryptographic authentication can be computationally intensive. By offloading the cryptographic operations to the secure element, the use of a very simple main microcontroller will save cost and simplify the design. In addition to the ability to perform advanced crypto functions in a few tens of milliseconds (orders of magnitude faster than if performed in firmware), this technique also saves code space and power. The crypto functions are hard-coded in the secure element, which means those functions do not need to be included in the main firmware code, thereby reducing memory requirements. Because the hardware is optimized for performing cryptographic math, it completes these functions far faster. Less compute time equals fewer clock cycles, and fewer clock cycles equal less power used.

Once the authentication and crypto operations are complete, the secure element goes to sleep and draws current only on the scale of nanoamperes.

Drawing upon the pre-established PKI ecosystems referenced in the previous section, it is now possible to pre-provision keys and certificates into these secure elements. By doing so, you have now reduced the implementation of PKI and digital certificates in a small device to a line item in a BOM. The cryptographic math is baked in, as are securely stored keys and the digital certificate. You add the secure element with its digital certificate on the I^2C bus next to your host microcontroller, add a small library/SDK to your firmware, and you are done.

Once the implementation is complete, you have now placed enterprise-grade security into an IoT device. Each device has a unique, cryptographically verifiable identity. You can verify that the certificate is authentic because the digital signature prevents the data from being altered. The securely stored private key proves that the certificate corresponds to the device that sent it. On top of that, this is a method that can be rapidly scaled to billions of units, if needed.

4.3 Designing Products for Security

The pre-packaged certificate-in-a-chip can now be used by the device manufacturer or system integrator. The key to implementing a trusted ecosystem is to treat IoT devices like users on a network. An IoT device logs in by itself and it sends data by itself. Certificates-in-a-chip allow for the ecosystem owner to embed verifiable unique identities in devices so that they can be managed almost like human users. The difference is that the device identity is authenticated with the ecosystem certificate as opposed to a username and password.

With the turn-key certificate-in-a-chip solution, different product lines can very easily adopt a common authentication method that is based on strong cryptography and security methodologies, yet simple enough to be implemented by engineering teams that are not specialized in cybersecurity. Implementation is a matter of adding one additional component on the device/system BOM and integrating a small software module into the main firmware to call the secure element when it is needed.

Now that there is a cryptographic identity embedded in every device, what can you do with it?

4.3.1 Automated Secure Device Commissioning

One of the trickiest aspects of IoT is adding a new device to a network or ecosystem. How do you know that the device is authorized to join? For IoT, a device's physical location typically matters because it is controlling something or taking data in a specific area. How do you assign a device to a location on a large scale?

With embedded certificates securely stored in IoT devices, you can use that unique identifier with mobile device software to rapidly deploy devices and assign physical locations to them and do it securely. A technician's mobile devices, IoT devices, gateways and servers will all have ecosystem certificates, and anything that wants to connect to them must authenticate using a certificate and private key.

With pre-provisioned keys and credentials embedded in secure chips that are manufactured into the products (as opposed to added in after manufacture), a cryptographically verifiable, securely stored, unique device identity is an integral part of each device. As such, it is able to authenticate locally in the case of isolated networks, or authenticate over a WAN to public or private cloud services and other network resources right out of the box.

Although this all sounds complicated, it can be completely transparent to the installer and end user, if done properly. The mutual authentications occur as part of standard protocols such as TLS. Assuming that all certificates are valid, the secure connection, device commissioning and other data transfer happens without any additional user intervention. It just happens, and the installer can move onto the next installation.

4.3.2 Access Control

With a verifiable identity and certificate data, you can now effectively control who has access to what on the network. Because each device can be individually identified by its certificate, you can put devices in groups and determine which devices can access certain parts of the network and which cannot.

Certificates can be used for server-to-server authentication:

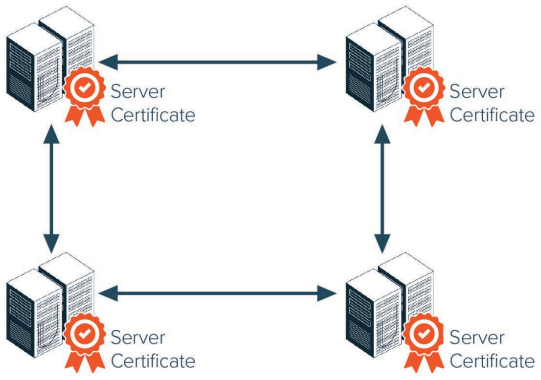

Figure 7 - Server-to-Server Authentication

They can also be used for device-to-server/cloud authentication:

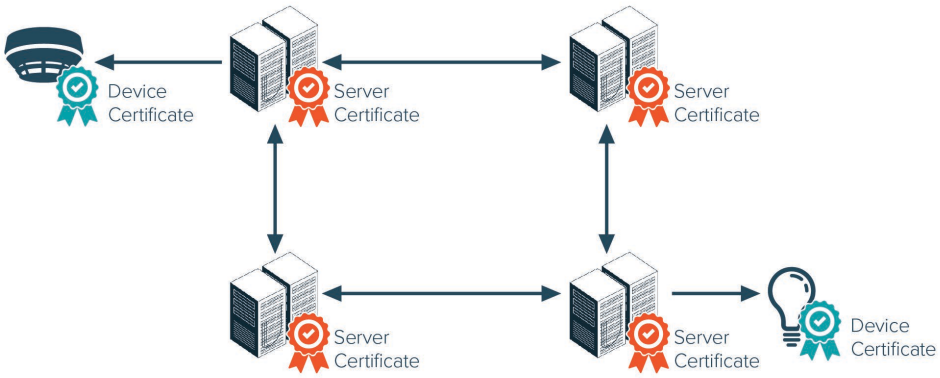

Figure 8 - Device-to-Server/Cloud Authentication

And they can be used for device-to-device authentication:

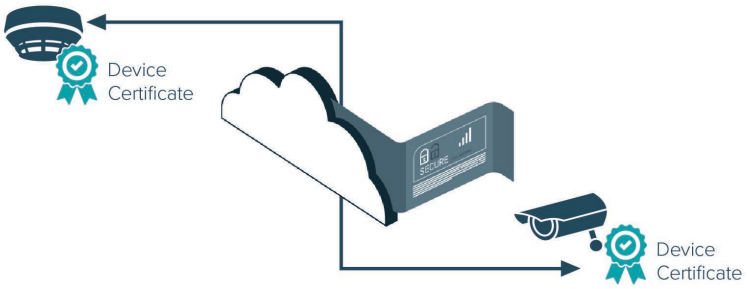

Figure 9 - Device-to-Device Authentication

Most access breaches occur not as a result of breaking the encryption but as a result of private key leakage or identity spoofing. An example of identity spoofing is when something like a serial number or MAC address is used as the device identity. If the intruder knows the proper format of that number and presents one that fits the profile, there are very limited (and often cumbersome) ways to verify whether the number is legitimate. Spoof attacks do not actually break any encryption or security. Spoofing is essentially identity theft and allows the unauthorized entity to pretend that it is someone/something that has legitimate access to the network. By spoofing the identity, it gains the same access as that identity.

Providing access control for devices is similar to providing access privileges to human users, but certificates are used to provide their identity. If the certificate is issued from a well-managed PKI and private keys are securely stored in the device, a digital certificate identity is very difficult to spoof because the signature prevents alteration and the private key proves ownership. Even in the case of the compromise of a private key, the use of a managed PKI allows for the revocation of a certificate. This prevents use of the compromised certificate even if all of the cryptographic math works out and helps provide layered security.

4.3.3 Access Denial

Although a digital certificate is difficult to spoof, it is not impossible. It is possible to potentially steal chips or compromise PKIs that are not well managed or implemented. In this case, it is possible to revoke a certificate, as described in Section 0. Revocation allows the ecosystem owner to disallow access by

someone/something bearing a certificate even if the cryptographic authentication succeeds. This can happen when a private key is leaked along with its associated certificate. Because the private key is what proves legitimate ownership, legitimate ownership can no longer be proven once they private key is stolen.

As shown in Section 0, you can revoke a single device certificate or an entire branch or chain of a PKI implementation. These revocations are logged in what is called a Certificate Revocation List (CRL). These are posted on servers that are called Online Certificate Status Protocol (OCSP) responders. Some techniques, such as OCSP stapling, help improve response time and efficiency, but details about those are beyond the scope of this white paper. Suffice it to say that when something wants to authenticate and sends its certificate, you can look it up to see whether that certificate has been revoked.

4.3.4 Executable Code Verification

Another benefit of having a managed PKI is that you can create a separate sub-CA branch that can be used to sign code.

Figure 10 - Using Managed PKI to Create Separate Sub-CA Branch to Sign Code

This is often referred to as a Code Verification Certificate (CVC) sub-CA. Authorized developers use the CVC sub-CA to sign code so that it carries a signature from the ecosystem PKI that can be verified. For example:

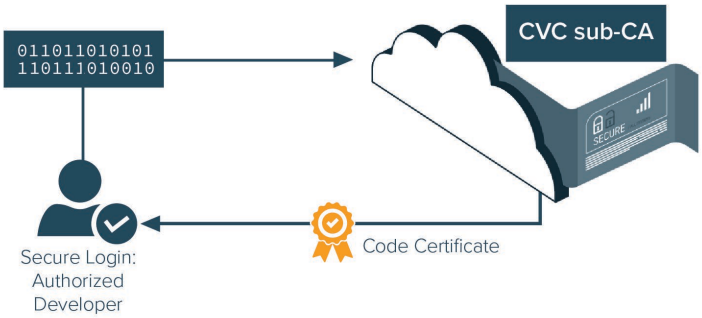

Figure 11 - Code Signing with Managed PKI

With code signing, you can now secure the trust in your ecosystem even further by programming your devices to only permit execution of code that has a verified signature. When combined with a Trusted Execution Environment (TEE), available in many of today's microcontrollers, you use PKI, digital certificates and secure semiconductors to provide a trusted ecosystem not just within a single device but across a global network of devices.

4.4 Bringing It All Together

For all entities in communication with one another within a trusted ecosystem, you need to:

- Verify the identity of the device
- Verify the identity of the server
- Verify the signature of any executed code

This means that in the ideal case, any communications over the network must be mutually authenticated using each endpoint's certificate.

The best approach is to make it so that implementing good ecosystem security meets the following requirements:

- **It must be economical in terms of cost and time.** The large benefit of ecosystem security must be bolstered by requiring only an incremental increase in BOM and development time cost. It must be inexpensive and simple enough so that there is no reason not to do it.
- **The solution must not require in-depth cybersecurity expertise to implement.** Security engineers are not cheap, and not every company can afford to have its own. The solution must be simple enough for a non-security person to implement, but not compromise the security itself. Storing keys and credentials in secure silicon provides a pre-packaged solution that offers both secure key storage and strong cryptography.
- **It must fit within the existing manufacturing flow for hardware devices.** If a solution requires a substantial change to the established hardware design and manufacturing flow, the probability that it gets adopted decreases dramatically.

To address these points, the security infrastructure must be part of the design and manufacturing flow—not bolted on after the fact. If security is implemented from the outset, it can be fed into the manufacturing flow for the chips and subsequently the IoT devices themselves. From there, once it is known that new devices are carrying certificates, you can start to apply the methods outlined in Section 0.

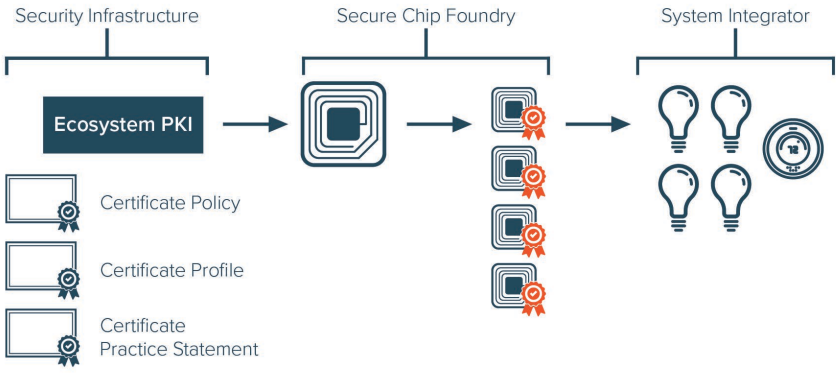

Figure 12 – Example Security Infrastructure in Design and Manufacturing Workflows

5. Conclusion

It is possible to implement highly scalable device security for IoT without compromising the strength of that security. If security is worked into the front-end of the design process, it makes it far easier to implement later on. With a managed PKI implementation backed by a trusted CA providing the rigor and process behind issuing and revoking authentic digital certificates, this secure back-end can feed directly into the manufacturing flow for semiconductor components used in everything from larger devices (e.g., gateways) to small IoT devices (e.g., lighting, sensors). With a cryptographically verifiable identity securely embedded in each device, you can maintain the integrity of the ecosystem and enable secure software updates by providing control over network access and code execution throughout the lifetime of your devices.

The value of IoT security lies in the protection of the greater ecosystem, not in individual devices. However, it is the sum of all the individual devices that collectively contribute to the security of that ecosystem which allows the value to be realized.