Establishing a Root of Trust (RoT) in EtherNet/IP Devices Implementing CIP Security

John S Rinaldi
President
Real Time Automation

Presented at the ODVA
2020 Industry Conference & 20th Annual Meeting
March 4, 2020
Palm Harbor, Florida, USA

Abstract

CIP Security™ for EtherNet/IP can assist manufacturers in preventing these sort of security breaches. The CIP Security standard requires authentication and integrity of EtherNet/IP messages. It requires both Scanners and Adapters to be authenticated to ensure they are the devices they claim to be, and message encryption to ensure integrity and privacy. CIP Security provides confidential communications between trusted entities, and disallows communication between untrusted entities, on an EtherNet/IP network.

However, no EtherNet/IP device can be secure without a mechanism for establishing trust. Key to any embedded security system, such as CIP Security, is the establishment of a Root of Trust (RoT) and effective protection of the certificates, passwords and keys of the Public Key Infrastructure (PKI). Failing to provide a RoT in a CIP Security device can compromise the security of an entire manufacturing system. This paper examines the various mechanisms for establishing that trust in a secure EtherNet/IP device.

Definition of terms

AES The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data created by the U.S. National Institute of Standards and Technology (NIST).

Attack Surface A term for measuring all the ways an attacker might compromise a device. Devices with smaller attack surfaces are more resilient to intrusion than devices with larger attack surfaces.

Authentication The process of identifying trusted entities, and accepting messages only from those trusted entities and rejecting messages from untrusted entities.

Authorization The process of allowing an entity to perform a requested function.

CIP The Common Industrial Protocol. CIP encompasses a comprehensive suite of messages and services for the collection of industrial automation applications and four industrial networks: EtherNet/IP™; DeviceNet™; ControlNet™; and CompNet™.

CIP Security CIP Security™ is the set of security-related requirements and capabilities for CIP devices, specifically EtherNet/IP devices.

Data Integrity Messages have data integrity when messages are transferred from a sender or a receiver without modification.

DEK A Data Encryption Key (DEK) is one of several terms used for the master key in a secure system.

DID Defense in Depth (DID) is an approach to cybersecurity in which defense mechanisms against cybersecurity events are layered such that an intruder must defeat a consecutive series of mechanisms to execute an attack.

DTLS Datagram Transport Layer Security (DTLS) is a modified version of Transport Layer Security (TLS) for securing UDP protocol messages. DTLS secures EtherNet/IP cyclic messages (implicit messaging) in CIP Security for EtherNet/IP communications.

EtherNet/IP EtherNet/IP is the Ethernet implementation of the Common Industrial Protocol (CIP).

KEK A Key Encryption Key (KEK) is one of several terms used for the master key in a secure system.

HSM A Hardware Security Module (HSM) is a standalone silicon component of an embedded device that is responsible for securely managing secrets and protected information.

NIST The National Institute of Standards and Technology (NIST), founded in 1901, and now part of the U.S. Department of Commerce.

PKI A Public Key Infrastructure (PKI) is the entire set of policies, processes and procedures for authenticating, authorizing and communicating securely on a network.

PUF A Physically Unclonable Function (PUF) is a process of using the static silicon features of a hardware component to generate a non-clonable encryption key.

RoT The Root of Trust (RoT) is the core of a secure CIP Security device. The RoT is the basis for performing critical security operations such as generating digital signatures and encrypting and decrypting messages.

Secrets All protected information including digital authentication credentials, passwords, keys, APIs, and tokens for use in applications, services, privileged accounts and other sensitive parts of a microprocessor’s data.

TCP The Transport Control Protocol (TCP) is a core component of the TCP/IP protocol suite. TCP is a connection oriented protocol in which a confirmation is provided when a message is received by a receiver. EtherNet/IP uses TCP for acyclic (explicit messaging) communications.

TEE A Trusted Execution Environment (TEE) is a secure execution area of a microcomputer system.

TLS Transport Layer Security (TLS) is the successor to the Internet SSL (Secure Sockets Layer) protocol, and is used to secure web traffic (HTTPS://). TLS secures EtherNet/IP acyclic messages in CIP Security for EtherNet/IP communications.

TPM A Trusted Platform Module (TPM) is a trade name for a silicon component of an embedded system that securely manages secrets and other protected information.

UDP The User Datagram Protocol (UDP) is a core component of the TCP/IP protocol suite. UDP is a connectionless protocol in which no confirmation is provided when a message is received by a receiver. EtherNet/IP uses UDP for cyclic (implicit messaging) communications.

FACTORY FLOOR DEVICE SECURITY

The requirement to keep confidential information confidential is hardly new. The Spartans of ancient Greece encoded messages vertically on leather wrapped in a helix around a wooden rod. Without the right key – the correct sized rod – the message couldn’t be decoded.

While we are now a long way from wrapping wooden rods with strips of leather, the need for confidentially exchanging messages hasn't changed. Today we rely on factory floor Ethernet systems to exchange confidential messages between controllers and end devices. In recent years, more and more of these systems have been extended to link enterprise and cloud applications to the factory floor.

Extending connectivity beyond the factory floor has increased the vulnerability of those systems to attacks. Attackers, sensing an opportunity, have shifted their attention from personal and business computers to the world of factory automation. Because the majority of these attacks are not publicized, no one knows for certain how many plants have had their servers locked, important data stolen, messages altered, and programmable controllers hijacked.

In the early years of Internet connectivity, it wasn't uncommon to have insecure controllers directly connected to the Internet. Over the years, these controllers have been removed, updated or replaced with newer versions that are more cybersecure. Most manufacturing installations have also added Defense in Depth (DiD) strategies that make it much more difficult to get to controllers and I/O networks from the outside.

What's often still open and vulnerable, though – if you can get to it – is the inside, the I/O network side of programmable controllers.

If you can get access to a manufacturing Ethernet network, you can often have free reign to create all kinds of havoc. There's generally nothing stopping you from accessing the controller tags over that network: turning pumps on or off; increasing motor speeds; or opening and closing valves.

In the past, there were a lot of barriers to getting access to these networks. You'd have to get past security gates and through guard houses and locked doors before you could plug into a control network. For those among us prone to mischief, it's now a lot easier with all the cloud connectivity and Internet of Things (IoT) devices (sometimes installed with little planning or forethought). Much of that IoT infrastructure has access to I/O networks, and sometimes isn't as rigorously protected by Defense in Depth strategies. Once an attacker gets in, they can attack the PLC from that soft underbelly – its I/O network communications – as well as play havoc with the I/O devices.

Even when strong cybersecurity protection is in place from the outside, factory floor systems can be compromised from the inside. Most facilities have an army of IoT vendors, automation vendors, technicians, system integrators, and corporate engineers who come onsite and knowingly or unknowingly bring viruses, malware, time bombs and worse into your plant and onto your critical I/O networks.

EtherNet/IP, the Ethernet implementation of CIP (the Common Industrial Protocol), was never designed as a secure communications transport. It is designed for ease of use and flexibility. Anyone can make connections to an EtherNet/IP Adapter and execute any operation, including a reset of the device. This makes EtherNet/IP a very insecure communications protocol.

In light of this relatively new environment, ODVA developed CIP Security for EtherNet/IP. CIP Security is a secure standard for the transportation of EtherNet/IP messages. It allows communication between trusted entities, and disallows communication between untrusted entities, on an EtherNet/IP network.

WHAT IS CIP SECURITY

CIP Security defines the security related requirements and capabilities of CIP devices and specifically for EtherNet/IP. It provides three benefits to a manufacturing system using EtherNet/IP:

1. Data integrity - It rejects data that has been modified during transmission.
2. Authentication – It rejects messages transmitted by untrusted entities.
3. Authorization – It rejects actions that an entity is allowed to perform.

To accomplish these objectives, CIP Security employs two standard IT cryptographic protocols: Transport Layer Security (TLS); and Datagram Transport Layer Security (DTLS). TLS is the standard cryptographic protocol used to secure internet communications and online traffic. CIP Security uses TLS to secure EtherNet/IP acyclic messages (explicit messages). DTLS is a version of TLS designed to secure UDP (User Datagram Protocol) messages. It is used by CIP Security to secure EtherNet/IP cyclic traffic (implicit messages).

But secure TLS and DTLS traffic is only possible if two entities trust one another. CIP Security for EtherNet/IP supports two mechanisms for entities to trust another: Pre-Shared Key (PSK) and X.509 certificates.

PRE-SHARED KEY (PSK) – Pre-Shared Key is an uncomplicated and simple system that works well in small systems. A private key is known and shared by all the devices in a network. The key is used to encrypt messages. Any device that knows the private key is authenticated and can encrypt and decrypt messages. For added protection, the key is typically changed at some set interval, sometimes as part of a maintenance cycle.

X.509 CERTIFICATES – X.509 certificates are a standard way for two devices to securely communicate. The devices share their certificates. Each certificate identifies the entity authenticating the certificate. That entity can be the device itself (self-signed certificate), the vendor who manufactures the device, or some outside authority that is trusted by all the devices in a network. The public key in a certificate is used to send encrypted messages to the certificates owner who uses his private key to decrypt the message. A private key associated with the certificate should never be disclosed.

A fundamental design tenet of CIP Security is that not all devices on an EtherNet/IP network need the same level of protection. Some devices are less critical and some are more critical to an automation system. CIP Security defines two security profiles to provide that different level of protection:

1. The EtherNet/IP Confidentiality Profile provides secure communications by requiring authentication and data integrity for all EtherNet/IP messages. Devices that are not authenticated are unable to make a secure connection. Messages that fail the integrity check are rejected.
2. The EtherNet/IP Authorization Profile goes one step further than the Confidentiality Profile. It provides User Authorization. With the Authorization profile, an application requesting an action like opening or closing a valve would have to be authorized to take that action. (It should be noted, however, that the EtherNet/IP Authorization profile is not currently part of CIP Security, and the specification describing how this is to be accomplished isn't available at the time of this writing.)

Devices that do not support CIP Security can coexist with devices that support the Confidentiality or Authorization Profiles.

CIP SECURITY KEY MANAGEMENT

> **"The proper management of cryptographic keys is essential to the effective use of cryptography for security. Keys are analogous to the combination of a safe. If a safe combination is known to an adversary, the strongest safe provides no security against penetration.**
>
> **Similarly, poor key management may easily compromise strong algorithms. Ultimately, the security of information protected by cryptography directly depends on the strength of the keys, the effectiveness of mechanisms, and protocols associated with keys, and the protection afforded to the keys."**
>
> **NIST Special Publication 800-57 Part 1, Revision 3**

CIP Security and all other key management strategies are completely undermined without an effective key management system. Attackers can exploit under-secured systems to steal intellectual property, gain access to proprietary information about processes and products, utilize them as platforms to propagate further attacks, and even cause real world physical damage and harm to humans. Proper key management techniques must be practiced to make a CIP Security device difficult to penetrate.

Unfortunately, there is no perfect security system and no flawless mechanism to completely secure the confidential information and security infrastructure in a CIP Security device or in any electronic device. All that can be done is to raise the bar for what is required by an attacker to penetrate a device and access any protected information and the cryptographic security keys. With upfront planning during product requirements and follow through during implementation, it is possible to raise the bar enough to dissuade most attackers.

Securing a CIP Security device, or any electronic device, begins with following basic security fundamentals. Some of the general techniques (which may or may not be practical for a specific device) include:

- Utilizing a tamper-resistant enclosure. Adding deadman switches or anti-tamper meshes that detect enclosure openings or other intrusions such as drilling. In devices where “high bar” security is required, the device would react to these intrusions with countermeasures, such as erasing cryptographic keys and other critical information.
- Implementing tamper-resistant circuitry to detect circuit board intrusions.
- Implementing a secure boot to authenticate firmware prior to execution.
- Using a microprocessor with a Trusted Execution Environment (TEE)[1].
- Encrypting and storing critical information in secure storage memories (if available).
- Implementing hardware-based random number generators to ensure truly random data encryption keys.
- Offloading cryptographic processing to hardware accelerators which are far more efficient than general-purpose CPUs and enable the generation of longer keys.
- Securing your remote update process to ensure that only factory-authorized code can execute.

All these practices are helpful, but the protection of your private key is paramount. Just as you would never leave home with the front door locked and the key in the lock, you wouldn’t create a strong private key and not protect it.

CIP Security depends on the use of strong encryption keys and the protection of those keys. CIP Security uses both symmetric and asymmetric encryption. Both encryption techniques are effective as long as the keys remains private. Encryption keys that are weak, or stored where an attacker can easily discover them, are likely to provide little real protection.

Common poor key management practices include:

- Using encryption keys that are too short. An encryption key is simply an input to a mathematical algorithm that generates an encrypted message. It’s possible (but not easy) for a hacker with adequate hardware to reverse the algorithm to generate the encryption key. Reversing the algorithmic process is more feasible with shorter keys[2].
- Using a flawed, non-standard, low entropy, random number generator to create a key.
- Storing an encryption key in application code, file or table.
- Storing an encryption key in a file or table.
- Storing an encryption key with poor protection (XOR with weak data, etc.).
- Storing an encryption key with password-based encryption protection.
- Failing to secure debug ports so that an attacker can monitor program execution.
- Failing to provide an audit trail of who accessed what data and when.
- Failing to provide a tamper-proof, secure clock.

These poor practices make your device hackable and, if your device is subject to audit, may result in compliance failure. Devices requiring advanced protection and subject to audits should use an effective key management system that is designed for that purpose and which meets an appropriate NIST standard like FIPS 140-2.

Beyond developing a device that isn’t subject to these poor practices, a well-designed CIP Security device must provide a resilient and effective Root of Trust. The RoT is the basis for performing critical

[1] A trusted execution environment (TEE) is secure area of a processor that is protected with respect to confidentiality and integrity. A TEE offers isolated execution, application integrity and confidentiality.

security operations such as generating digital signatures and encrypting and decrypting messages. A Root of Trust can be as simple and inexpensive as a master key stored in the device's non-volatile memory, or as complex and costly as a Hardware Security Module (HSM) whose sole purpose is to protect the Root of Trust from attackers.

Every CIP Security device vendor must decide what Root of Trust is appropriate for their device and how much additional cost in design, development and operating expenses are tolerable for their device. Security organized around a software RoT is less expensive than a hardware RoT, which adds cost to the bill of materials, consumes precious circuit board real estate and increases complexity in development, testing, support and operations.

SOFTWARE ROOT OF TRUST

Using a Hardware Security Module (HSM), some type of protected memory or other hardware add-on is sometimes infeasible in manufacturing devices. Adding hardware to a device can be impractical, cost prohibitive or unnecessary. In these cases, cryptographic keys and other critical information can be secured using a software Roof of Trust.

Note that a software Root of Trust (RoT) is always going to be problematic. There is no perfect key security and no mechanism that is foolproof.

There are any number of approaches an embedded design might use to implement a software Root of Trust. Four common approaches follow.

Method 1 – Hide the Key in Plain Sight

Description: Store security keys as clear text in standard, non-volatile RAM. This mechanism relies on the inability of most attackers to 1) penetrate the device internals, 2) access non-volatile RAM, and 3) identify which bytes are the device keys.

Why Use it? Devices where vendors believe that consequences of an attack are insignificant can rely on this approach. Devices of this sort include simple input only sensors and other input only devices with largely inconsequential data. This approach is generally not appropriate for an actuator where an attacker can affect outputs that might control a process.

Limitations: This approach provides only minimal key protection and should be limited to devices that the vendor is certain will never be used for anything critical.

A determined attacker may be able to run the executable image in a VM, dump the RAM, sniff network traffic, decompile the executables…etc. and easily compromise the device. Access to the pre-shared key means the device has the key for all devices in the CIP Security zone.

THIS APPROACH SHOULD NEVER BE USED ON A CIP SECURITY NETWORK USING PRE-SHARED KEYS; A COMPROMISED DEVICE COMPROMISES THE ENTIRE NETWORK!

Attack Surface: Large

Cost: Minimal

Method 2 – Let the OS Worry About It

Description: Shift the burden of key management to the operating system and the OS vendor. Use the key management facility available in your embedded operating system. Many operating systems, including Linux and Windows, now provide secure key management.

Why Use it? Many operating systems now provide key management based on the Advanced Encryption Standard (AES), a specification for the encryption of electronic data created by the U.S. National Institute of Standards and Technology. Vendors are cautioned to thoroughly vet the key protection mechanisms of an operating system. A “name” OS may not offer any additional protection (and sometimes less) than other vendor self-implemented key protection strategies.

Limitations: The security mechanism for well-known open source operating systems are well-known. Attackers know the methodologies, vulnerabilities and weaknesses of these operating systems.

Some of these standard mechanisms require an Initialization Vector (IV) which must be secured. Having to secure an IV to secure secrets is no less of a problem.

Attack Surface: Varies with the selected operating system

Cost: Minimal

Method 3 – Multi-level Key Storage

Description: A master key known as a DEK (Data Encryption Key) or a KEK (Key Encryption Key) is used to secure the CIP Security private key or the entire certificate chain.

Why Use it? A DEK or a KEK master key is longer and stronger than the private key it encrypts. An attacker must first find the DEK or KEK and then use it to decrypt the private key.

Either just the private key or the entire certificate chain can be secured with the master key.

Limitations: How to secure the DEK or KEK master key is an issue. Some vendors will likely choose to avoid the indirection of a DEK or KEK master key and protection of that additional key. They may instead choose to just focus on protection of the master key.

Attack Surface: Moderate

Cost: Minimal

Method 4 – No Key Storage

Description: Using Physical Unclonable Function (PUF) technology, security keys and unique identifiers are extracted from the innate characteristics of a semiconductor component. Like biometrics measures, these identifiers cannot be cloned, guessed, stolen or shared. Keys are generated only when required and don’t remain stored on the system, hence providing a very high level of protection for the DEK or a KEK master key.

Why Use it? PUF keys are guaranteed to be unique and unclonable since they utilize the inherent randomness of the manufacturing process. No master key is ever stored. Low cost, easy to integrate and ultra-secure key protection.

PUF is supported by ARM, Renessas and other silicon manufacturers.

Limitations: PUF technology must be licensed and maintained.

Attack Surface: Small

Cost: Minimal when included with your processor development system.

HARDWARE ROOT OF TRUST

A Root of Trust in hardware often lends itself to a more secure solution for CIP Security devices than a software Root of Trust. A hardware Root of Trust strengthens the attack surface that must be overcome to compromise a device. However, it does that at increased cost and complexity over a software RoT

A hardware Root of Trust can be implemented as simply as performing key management in protected processor memory, using some isolated off-processor memory component for key storage, or implementing a complete HSM using a commercial Trusted Platform Module (TPM).

The advantages of HSMs in general include:

- Inaccessibility by systems outside the microprocessor ecosystem.
- Hardware accelerators for processor-intensive cryptographic functions.
- True random number generation.
- Secure clocks for applications where time is critical.
- Protected key management.
- Intrusion protection.

Of these advantages, hardware acceleration and true random number generation provide significant value. With software RoT, the processing of cryptographic algorithms can introduce latency into device communications and validating that a random number generator is producing truly random numbers can be challenging.

Method 1 – Non-volatile Memory

Description: Encryption keys are stored alongside other protected information in a special non-volatile memory implemented solely for secrets. Secrets are deemed secure because the device firmware provides no mechanism for unauthorized access.

Why Use it? Implementation simplicity and lower cost development, production and support. It can be implemented with a firmware update and requires no modification to hardware. There is no change to software tool chain and it provides the ability to add security to legacy devices in the field.

Limitations: Requires extremely high-quality source code, open source and third-party libraries, as any flaw may compromise secrets. An attacker with physical access to the device can unsolder the non-volatile memory to discover its secrets.

Attack Surface: Large

Cost: Minimal

Method 2 – External Hardware Security Module

Description: A dedicated cryptographic module specifically designed for the protection of the embedded device secrets. Located off device, the module can be networked, attached, embedded in a PC server, or attached via USB.

These types of modules address a broader scope of security needs but can provide the key storage and data encryption of an embedded system.

Why Use it? Cryptography is managed by an external provider.

Limitations: May be impractical in the embedded environment due to cost and latency issues.

Attack Surface: Small

Cost: Significant, although one module can support many embedded modules.

Method 3 – Onboard Hardware Root of Trust Device

Description: There are various types of hardware RoT devices offered by semiconductor companies marketed as hardware RoT devices. Some vendors offer a class of device known as Trusted Platform Modules (TPMs). Others provide similar devices under various trade and generic names. These modules are typically microcontrollers that securely store secrets and platform measurements to ensure that the platform remains trustworthy. Secrets are confined within a security-hardened, tamper-resistant IC, and thus hard to get at directly, even with an unlimited budget and expertise.

Why Use it? Hardware RoT devices significantly increase the resources and technical expertise necessary to compromise a device. They can simplify aspects of firmware development through a well-documented and well-supported common API.

Limitations: Unit cost and footprint can be problematic for some simple devices. May lead to complacency from over reliance on the device; the “I don’t need to worry about it” affect.

Attack Surface: Small

Cost: Varies with the manufacturer

This discussion should not conclude without discussion of bootloader and firmware managers. These devices, such as the STM X-CUBE-SBSFU, provide for secure transfer for microcontroller firmware. The update process is performed in a secure way to prevent unauthorized updates and access to confidential on-device data. Many robust off-the-shelf solutions are available. They validate the integrity of the firmware running the CIP Security device but can be cumbersome and intimidating for field personnel.

SUMMARY AND CONCLUSION

The number of well-documented attacks on cities, transportation systems and other public infrastructure systems continues to grow. The level of concern regarding securing manufacturing systems has now reached the C-suite. It is becoming apparent to executives and investors in manufacturing organizations that the growing deployment of Ethernet, cloud communications and IoT applications is increasing their vulnerability to attack.

Unlike with enterprise devices, there aren’t a few dominant suppliers that can be relied upon to secure the factory floor. The nature of manufacturing is such that manufacturing devices are designed to accomplish widely disparate tasks and have vastly different computing platforms, operating systems, code complexity and functionality. These devices are now beginning to be equipped with PKI-based security features like CIP Security, but the encryption and authentication underlying all these systems can be completely undermined if proper key management at the device level isn’t enforced.

Unfortunately, there is no perfect security system and no perfect mechanism to secure the confidential information and security infrastructure in manufacturing devices, or in any device. All that can be done is to raise the bar for what is required by an attacker to penetrate a device and access protected information and security keys.

Further complicating all this is that embedded devices are mass produced. There may be thousands to millions of identical devices. If a hacker can build a successful attack against one of these devices, the attack can be replicated across devices at many manufacturing sites in different geographies and industries.

The key question is “how high to raise the bar?” The bar can be raised so high that cost and complexity rise to a point where the device becomes essentially unusable. Conversely, the bar can be so low that attackers come to know the weakest link in every manufacturing system.

There are many ways to think about how high to raise the bar for a specific device, but two questions predominate:

1. What are the consequences of compromised security? It may be a nuisance if a temperature sensor is compromised but *devastating* if the device is integral to critical infrastructure in applications like nuclear, medicine or transportation.
2. Does the device contain critical and highly valuable data?

When the stakes are high enough, attacks are multi-phased, multi-year efforts carried out by large, well-funded teams of attackers. In these situations, it's not about protecting a device from malformed IP packets and DoS packet floods. Cyberterrorists will invest significant resources in gathering information on the device or devices prior to the attack, and then mount sophisticated and complex attacks.

Unfortunately, most embedded design engineers won't know for certain where and how their products are used, and the level of protection appropriate for a specific application. Devices tend to be incorporated in diverse industries, across geographic boundaries and in unusual applications.

In these circumstances, designers should opt for selecting a Root of Trust model that is affordable yet offering the most comprehensive and highest level of protection appropriate to the typical application of their device.

****

The ideas, opinions, and recommendations expressed herein are intended to describe concepts of the author(s) for the possible use of ODVA technologies and do not reflect the ideas, opinions, and recommendation of ODVA per se. Because ODVA technologies may be applied in many diverse situations and in conjunction with products and systems from multiple vendors, the reader and those responsible for specifying ODVA networks must determine for themselves the suitability and the suitability of ideas, opinions, and recommendations expressed herein for intended use. Copyright ©2020 ODVA, Inc. All rights reserved. For permission to reproduce excerpts of this material, with appropriate attribution to the author(s), please contact ODVA on: TEL +1 734-975-8840 FAX +1 734-922-0027 EMAIL odva@odva.org WEB www.odva.org. CIP, Common Industrial Protocol, CIP Energy, CIP Motion, CIP Safety, CIP Sync, CIP Security, CompoNet, ControlNet, DeviceNet, and EtherNet/IP are trademarks of ODVA, Inc. All other trademarks are property of their respective owners.