White Paper – Security for the Industrial Internet of Things

Device Security for the IIoT

Authors: Carl Stenquist and Wil Florentino, Renesas; Alan Grau, Icon Labs

July, 2016

Abstract

The Industrial Internet of Things (IIoT) is driving investment in new technology and manufacturing methodologies as companies rush to capitalize on a predicted 14 trillion dollar economic gain. A wave of new connected devices is the result, many of which are small endpoint devices running real-time operating systems. These systems often perform critical functions in our factories, electric grid, transportation infrastructure and other essential elements of modern society. The viability of the IIoT therefore depends on the security of the endpoints, the network and all of its subsystems. Unfortunately though, traditional security solutions do not scale down to support the RTOS-based devices that make up the bulk of the IIoT. New solutions and approaches are required.

This paper describes security requirements for IIoT devices and details solutions available today for securing these devices.

I. Introduction

Multiple benefits

IIoT, the Industrial Internet of Things, is being hailed by some as the next great industrial revolution. By some estimates, there are more than 60 million machines in factories worldwide and 90 percent are not connected. It is no surprise that companies are looking at this opportunity to create new connected IIoT, not just to reduce costs of operation or eliminate downtime, but to add new solutions and services for new revenue streams.

An important area in which the IIoT creates value is the creation of a network of device endpoints: smart, connected sensors and controllers that not only talk to each other, but also monitor and manage a wide range of machines and industrial systems. By combining this connectivity and functionality with analytics, information technologies and operational technologies, owners of industrial plants will obtain major benefits. For example, factories can be designed to adapt in real time to changes during production, or to anticipate and avoid events that might degrade operations. Additionally, predictive maintenance programs can be implemented to eliminate the downtime or catastrophic consequences caused by unanticipated failures of critical system components. By achieving even small percentage gains in plant operations or reductions in unplanned downtimes, these types of upgrades will dramatically improve the profitability of manufacturing operations.

To take full advantage of the improvement opportunities offered by the IIoT, an entire system—from sensors, actuators and motors, up through the controllers—should be connected to information and operational technology systems, and beyond into the cloud. Expanded connectivity will boost efficiencies in operations and integrate the supply chain more tightly and in innovative ways. It will also enable entirely new business models and revenue streams.

Substantial challenges

Although connectivity is the key to unlocking the full potential value of the IIoT, it brings a risk of cyberattacks. When systems are connected to the Internet and larger corporate networks, cyberattacks become possible, even likely, from external and internal sources—whether accidental or malicious. The benefits of the IIoT therefore cannot be achieved without multiple layers of security that successfully protect all the networked systems and devices. Secure communication, secure network monitoring and securing code execution at the device level are essential, not optional.

It's critical for system engineers to address security issues at every layer. Although traditional IT-endpoint security and network-monitoring solutions protect IT business applications, such solutions won't work for the embedded devices closest to the physical systems. These operational assets must be protected against cyberattacks by integrating security directly into the endpoint devices themselves.

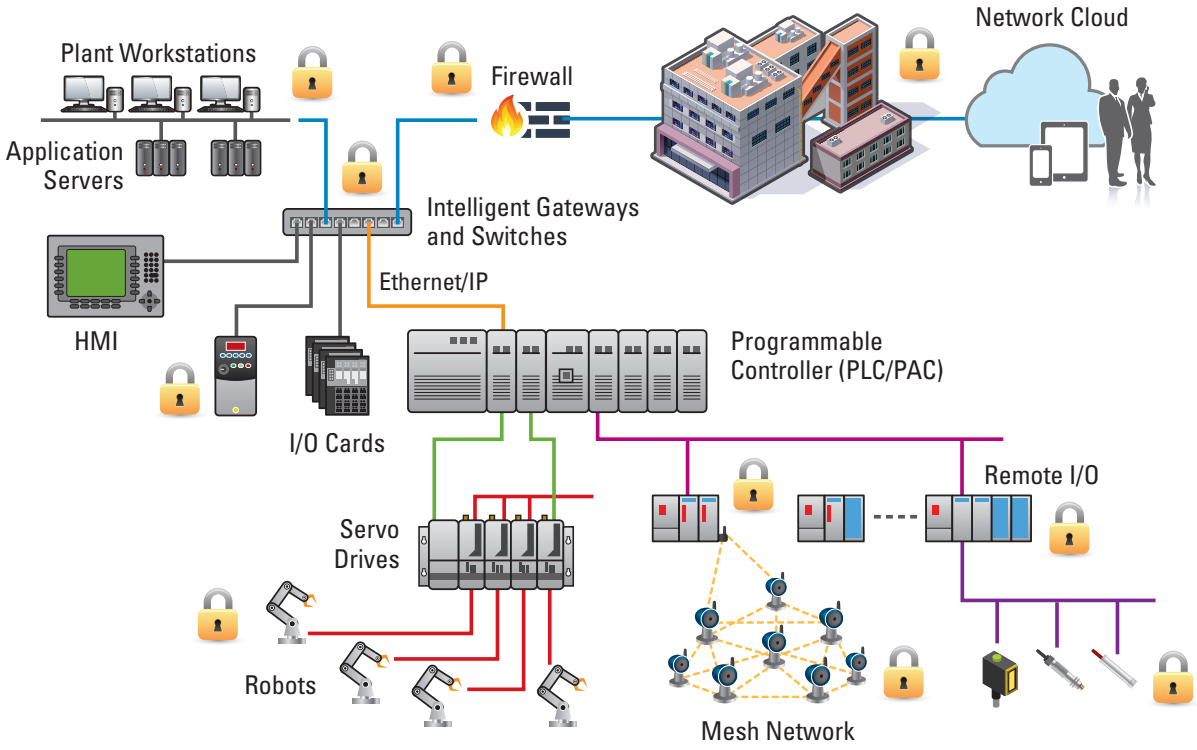

Figure 1. The factory network is comprised of countless endpoints from sensors, actuators, and controllers – each a critical component that needs security.

Minimizing vulnerabilities requires both specialized security hardware as well as software. To support enterprise security standards, embedded devices must incorporate the following key features: secure boot code, secure application updates, tightly controlled authentications, and secure communication protocols.

Figure 2. Objectives of a Security Solution

II. What Do We Mean by Security?

Security in its simplest form entails ensuring that authorized operations and actions are allowed, while unauthorized actions are blocked. Most cyberattacks against embedded devices exploit one of the following categories of vulnerabilities:

Insecure by design: Devices that use hard-coded passwords, transmit login credentials in the clear, allow remote accesses without authentication, or have other obvious unprotected interfaces.

Security with significant loopholes: Devices with built-in backdoors, which use weak or default passwords, permit plaintext storage or transmission of encryption keys, or have similar vulnerabilities.

Good, but partial security: Devices that provide strong security against certain types of attacks, but leave other interfaces unprotected. Prime examples are systems that implement TLS only for some but not all communication; and security protocols in which the setup-phase encryption key is exchanged without being encrypted, making them vulnerable to eavesdropping attacks. Other examples include systems that implement secure communication, but don't incorporate secure boot capability; and systems that have a secure operating system, but fail to secure the application layer.

Features that are vulnerable to exploitation: Devices that have weak encryption, exploitable buffer overflows or zero-day vulnerabilities, or cannot withstand brute-forcing of their authentication mechanisms. The hackers depicted in movies and on TV typically take advantage of these types of deficiencies.

Unfortunately, although it's vitally important to secure networked devices against such vulnerabilities, the reality is that most currently deployed embedded devices cannot withstand even very basic forms of cyberattacks.

Ensuring the security of IIoT devices requires addressing all of the issues described above. Ideally, robust design solutions will include adaptable security policy management and the ability to securely update firmware to protect against new types of attacks as they emerge.

Challenges for Creating a Secure IIoT

The Industrial Internet of Things encompasses a wildly diverse range of connected devices and systems: from small to large, simple to complex. They span from commercial gadgets to sophisticated systems found in military, utility and processing/manufacturing systems.

Embedded devices are very different from standard PCs or other IT products, but they constitute important and growing elements of the expanding web-connected network. Many of them use specialized real-time operating systems such as ThreadX, µC/OS-III or Nucleus, or a stripped-down version of Linux.

Installing new software on most embedded devices deployed in the field either requires a specialized upgrade process or simply can't be done. Further, in most cases, these ubiquitous devices are optimized to minimize processing cycles and memory usage. Therefore, they don't have the extra processing resources required to support traditional security mechanisms. As a result, standard PC security solutions can't solve the challenges of making embedded devices safe from cyberattacks. In fact, given the specialized nature of embedded systems, PC security solutions won't even run on most embedded devices.

The driving principle for enterprise security is to provide multiple layers of protection. Firewalls, authentication/encryption schemes, security protocols, and intrusion-detection/intrusion-prevention mechanisms are well established, widely adopted enterprise security solutions. Nevertheless, firewalls and intrusion detection features are virtually absent in embedded systems, which typically rely on simple password authentication and security protocols.

Typically, makers of embedded devices have assumed that their products aren't attractive targets to hackers. Other common perceptions have been that networked embedded devices aren't vulnerable to attacks and that authentication and encryption can adequately protect against cyberattacks. These assumptions are no longer valid. Today the number and sophistication of attacks against embedded devices is rising to worrisome levels.

This trend has impacted many new product designs. Whereas cybersecurity has long been a critical focus for large enterprises, it's now a strong focus for most system engineers building sensing and control devices. Fortunately, rather than reinventing the wheel, product developers can apply the security principles used to implement enterprise security.

To ensure security for embedded devices, given their specialized nature, the following concerns must be addressed:

- **Preservation of functionality**

  Embedded control devices are at the heart of the transportation infrastructures, utility grids, communication systems and other elements essential to modern society. Successful cyberattacks on them can have catastrophic consequences. Thus, security solutions must protect both the data stored on networked embedded devices and safeguard the operations they perform.

- **Attack replication**

  After embedded devices are developed, they are mass-produced. If a hacker can find a way to successfully attack one of these devices, that attack can be replicated across all devices of the same type. Thus, a single-point breach can become a mass-failure mechanism.

- **Assumed security**

  Many system engineers have long assumed that embedded devices are not targets for hackers; i.e., they have relied on security by obscurity. That assumption is totally false today. Security should now be considered a top priority for most embedded designs.

- **Upgrade difficulties**

  Most embedded devices are not easily patched. After they are deployed, they run factory-installed software for as long as they remain operational, even if that code has security vulnerabilities.

- **Long life cycles**

  Life cycles of embedded devices are typically much longer than for PCs or consumer devices. Devices may be in the field for 15 to 20 years or more. Implementing designs that will withstand the ever-evolving security threats expected over the next two decades is a tremendous system engineering challenge.

- **Deployment outside enterprise security perimeter**

  Many embedded devices are either mobile or deployed in application environments. As a result, they may be directly connected to the Internet, totally outside the security protections installed in corporate environments.

III. Secure Foundations

Cryptographic methods provide the foundation for implementation of many of the features used to secure embedded devices including secure data storage, secure communication protocols, and secure boot. These techniques require cryptographic keys, which must be kept secret.

System engineers often implement secure key storage in hardware using a hardware security module (HSM). Most HSMs also offer both crypto-acceleration to offload computation-intensive operations from the main CPU and True Random Number Generation (TRNG). Additionally, some HSMs provide protected code execution that allows security-critical operations to run in a separate memory space that user-space code cannot access. This prevents programs in the user space from tampering with the operation of security-critical features or stealing keys.

The hardware security features of the Renesas Synergy™ Platform (Figure 3), and versions of the Trusted Platform Module (TPM) are examples of secure crypto-processing.

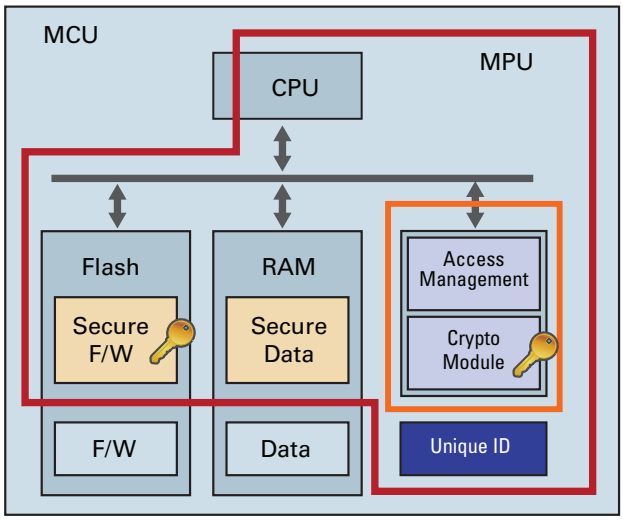

Figure 3. Block diagram showing details of Renesas' Synergy HW security modules.

Asymmetric vs. symmetric encryption

Regardless of whether encryption is implemented in hardware or software, most security protocols employ both symmetric and asymmetric types. IPsec does this unless pre-shared keys are used, and TLS uses asymmetric encryption to securely exchange a secret key at the start of a session. Subsequently, IPsec and TLS both transition to symmetric encryption using the secret key established during the key-exchange process. The reason for this approach warrants some explanation.

Symmetric encryption

Symmetric encryption is simple. It uses a single secret key that is shared by both of the communicating entities or nodes (see Figure 4).

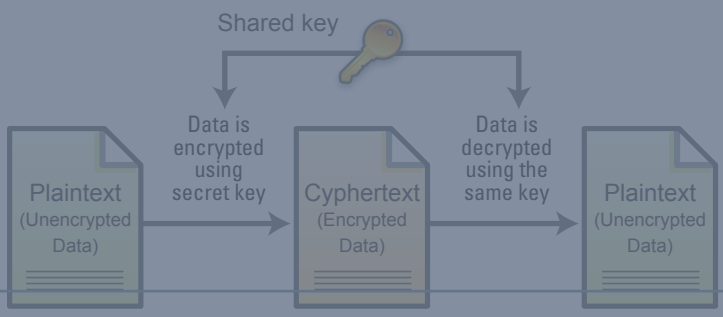

Figure 4. Symmetric encryption uses a secret key with the potential flaw of the network link; it provides limited security.

The problem, of course, is how to keep the shared key secret – something intercepting them?

Asymmetric encryption

Asymmetric encryption uses a key-pair consisting of a public key and a private key (see Figure 5). Each node has its own key pair. The private key must be protected and kept secret, but the public key can be shared with other nodes.

Key-pairs are created in such a way that data encrypted with a public key can be decrypted only with the correct private key, and vice versa. Given a key-pair (PrivK, PubK), data is encrypted and decrypted as follows:

EncData = Encrypt(PubKey, Data)

DecData = Decrypt(PrivKey, EncData)

The resulting decrypted data will be the original data.

Data = DecData.

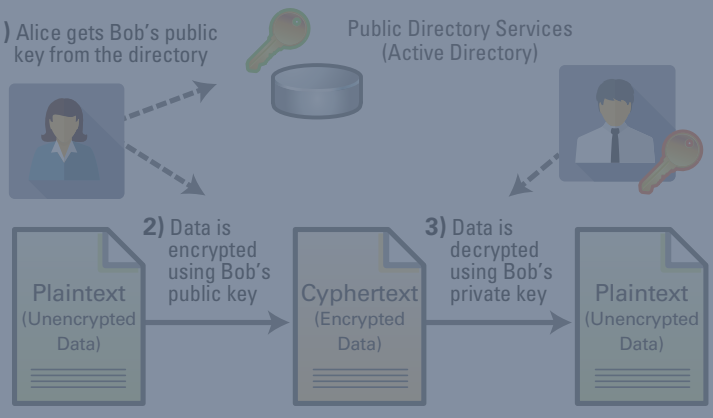

Figure 5. Asymmetric encryption uses public and private keys to provide a high level of security. Alice, to the left, uses Bob's public key to encrypt data. Bob then uses his corresponding private key – the only key available – to decrypt the message.

All key-pairs are mathematically related to enable encryption/decryption in this manner (Figure 5).