

STRENGTHENING SECURITY IN EMBEDDED IoT SOLUTIONS



The Threat to Embedded Systems

In the Internet of Things (IoT), security has become an increasing concern and a non-negotiable requirement. Resource-limited embedded devices are on the front lines of broader IoT solutions that are more frequently enduring malicious attacks. Unlike their PC or Internet counterparts, resource limitations mean these devices must employ unique security methods to defend against these attempts to breach and disrupt operations.

As embedded devices increasingly rely on wireless networks for communication, the opportunities for compromise increase. The “attack surface” has grown as new wireless links remove the physical access barrier. Also, the tools and expertise to analyze and modify embedded devices are now available even to hobbyists. These two factors have combined to exponentially increase embedded-systems security attacks.

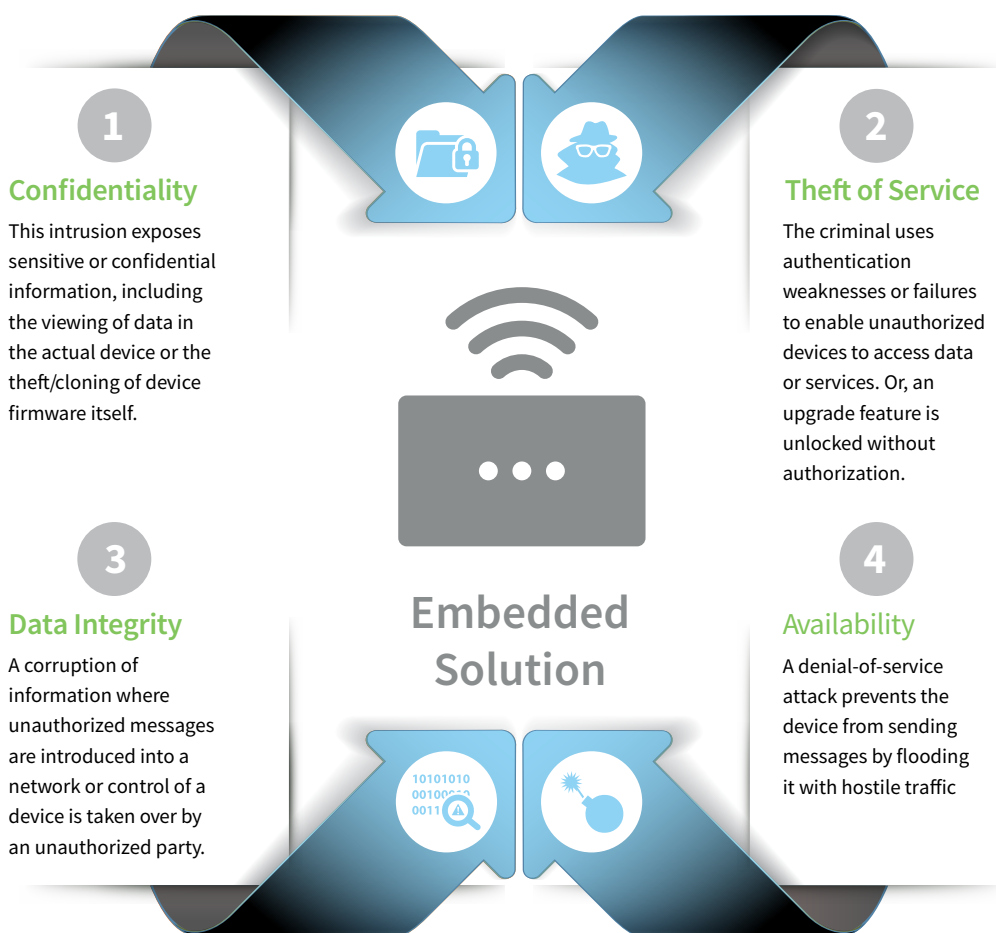
Many companies assess the data in their IoT networks – typically mundane data with little intrinsic value outside the organization – and mistakenly assume it is unlikely to attract the unwelcome attention of attackers. The fact is, secondary rewards are often the underlying motivation for successful breaches of otherwise mundane operational data typically found in a standard IoT solutions. Competitors may want to expose and exploit a vulnerability for sales purposes. Hackers may want to demonstrate their abilities in order to accrue social status and an elevated profile in their community. Disgruntled employees may seek to damage a company’s business and reputation.

As the frequency and impact of attacks has increased, regulators and customers are responding more aggressively to failed security implementations. Security breaches that violate the Health Insurance Portability and Accountability Act (HIPAA) can result in fines of \$50,000 per violation. Credit card processors that fail to comply with Payment Card Industry Data Security Standards (PCI DSS) may be fined up to \$100,000 per violation. Publicized security weaknesses can result in a crippling loss of customer confidence, leading them to choose other solutions. And in extreme instances, a security lapse in embedded security solutions can even open up companies to major liability exposures.

As embedded devices increasingly rely on wireless networks for communication, the opportunities for compromise increase.

Types of Security Threats to IoT Solutions

The nature of security threats will continue to evolve, leaving us numerous ways to classify and name them. However, for IoT terminal devices, we can sort threats into four categories:



These types of attacks may come from either a wireless or wired network or through local access. Efficiently controlling/minimizing the effects of these attacks is a challenge facing designers of these resource-limited embedded systems in the IoT network.



Identifying and Prioritizing Threats

To successfully mitigate the security risks that are increasingly prevalent in IoT networks, advance planning and analysis are essential. Unlike traditional PCs and servers, these environments have no add-on security solution like a firewall or anti-virus application. There is no “bolt-on” security in a production IoT network.

When we are forced to respond to a theoretically infinite range of attacks with an all-too-limited level of development resources, we must identify which threats are the highest priorities before undertaking monitoring and mitigation activities. Many companies use one of two primary methods:



FMEA - Failure, Modes and Effects Analysis (FMEA) is a popular methodology for security analysis. Using this approach, we estimate the potential risk, potential severity, likelihood and detectability for each identified potential risk. These are scored and combined to give each risk a relative priority, so that the highest risks can be given the most attention.







DREAD Analysis – Damage potential, Reproducibility, Exploitability, Affected users, and Discoverability (DREAD) assessment is another common model used for security analysis. For each identified potential risk, the severity for each of the five impact categories is estimated and summed. Like the FMEA approach, we use the resulting risk score to prioritize our mitigation efforts and responses.

Of course, few organizations are in a position to be experts in managing security risks for their embedded systems, leading them to turn to a number of contract development service organizations. Additionally, service organizations can leverage their independent status to offer credible attack testing and design-audit services. Your colleagues are a good source of contacts for effective contract development partners, or search for “wireless design services” or “embedded design services.”

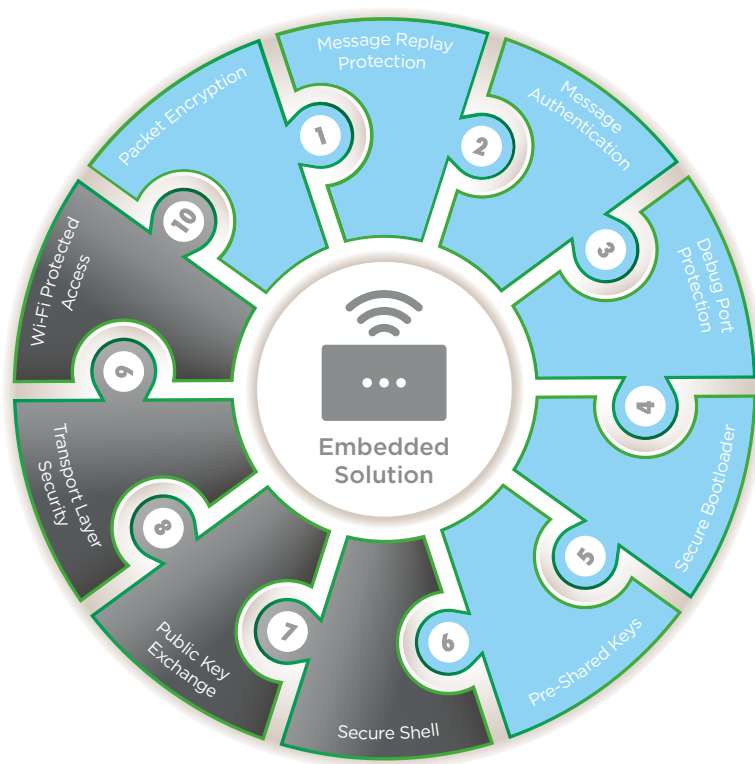


Security Methods for Embedded Systems

While embedded systems employ many of the same security methods used by their PC/Internet counterparts, the implementations are significantly different for embedded IoT solutions. With embedded systems, we must work with and around important constraints:

-  **INTERFACE DIFFERENCES** – Embedded systems typically use additional or specialized interfaces, especially for wireless connectivity.
-  **LIMITED RESOURCES** – Embedded devices have far less battery power, processing speed and memory.
-  **BASIC USER INTERFACES** – Embedded devices, have no GUIs, and error messages can be as basic as a coded series of beeps or flashing lights. This is particularly true for security status and control functions.
-  **HARDWIRED PORTS** – These provide unfortunate opportunities for compromise.

This means IoT solutions can't simply implement a strong password over a TLS connection – the most common approach for PC/Internet applications. IoT solutions need a different approach, and the effort required to identify and mitigate unique security risks in embedded systems is often underestimated, if not overlooked entirely. In the following sections, we look at some of the important and effective methods for improving the security of embedded systems.



Embedded Solution Security Methods

Method	Complexity, Resources Needed	Notes
Packet Encryption	Low	Foundation for most embedded system security
Replay Protection	Low	Prevents resubmission of recorded messages
Message Authentication Code	Low	Prevents messages from being changed
Port Protection	Low	Secures ports that may be physically accessed by an attacker
Secure Bootloader	Moderate	Ensures only authorized firmware is allowed to run
Pre-Shared Keys	Low	Preferred for smaller systems
SSH	High	Generally on OS-based systems; can prevent malicious connections
Public Key Exchange	High	Generally on OS-based systems; can prevent malicious connections
TLS	High	Generally on OS-based systems; can prevent malicious connections
WPA2	High	Generally on OS-based systems; can prevent malicious connections

Packet Encryption

This is the “go-to” method for protecting data exchanges in IoT solutions with smaller embedded terminal devices. Most systems have resources to implement basic encryption, such as FIPS-197/AES, protecting messages from unauthorized viewing or malicious changes. This method is easy to implement and use, especially when used in conjunction with private keys.

Message Replay Protection

In this approach, encrypted packets are enhanced with data fields that vary in a way known to the recipient (which could be as simple as a date stamp). Then, the recipient can enforce a rule that messages are accepted only once or in a sequence. This prevents recorded, but not necessarily decrypted, messages from being resubmitted at a later time to cause the original action, such as “open door.” This method is also simple to implement and is often used when individual messages can cause state changes.

Message Authentication Code

To create a message authentication code, we can run a cipher or a hash algorithm on the content of a data packet to create a short signature that accompanies the message packet. The recipient uses the same cipher or hash to confirm that the message has not changed. Message authentication codes provide explicit protection from tampering and can enable some systems to safely use clear-text messages. For example, we can use this method for systems that transmit data that is non-confidential (e.g., air temperatures) but that must not be tampered with. This is another low-complexity method that is useful for many types of embedded systems.

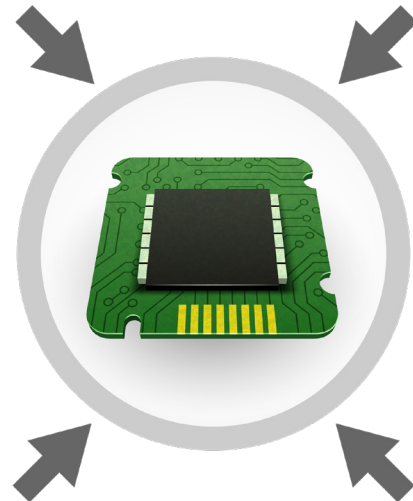
Debug Port Protection

The hardware ports we use for system configuration, control, and analysis (e.g., JTAG ports and serial logging ports used for firmware development and debugging) can also be vulnerable targets for security attacks. In fact, it's easy to underestimate the determination and expertise of someone who attacks M2M terminal devices. As a start, these ports may be protected with a factory password before further actions are allowed. Better yet, we can internally disable these ports in field-deployed units.

Secure Bootloader

Even for a development team with unrestricted access to the required technical information, it can be daunting to correctly build and load firmware into a resource-limited embedded device. That can make the chances for an attack based on a malicious firmware modification seem remote. But the rapidly-increasing sophistication of embedded-system attackers, combined with product requirements for easier field upgrades of device firmware, have created a risk that must not be overlooked.

We configure the device to check for a MAC signature in the firmware image during start up to ensure it is authorized to run on the product. The image may also be encrypted for further protection. Secure bootloader solutions demand careful attention to management of keys and support for debugging.



Once the attack risks have been identified and prioritized, we should focus on the steps we can take to control the higher-priority risks.

Pre-Shared Keys

To have secure IoT systems, the communicating devices need access to compatible keys. The use of Pre-Shared Keys (PSK), minimizes the demands on the resource-constrained embedded-system terminal device. Keys can be transferred through an independent

Secure Shell

We can use the Secure Shell (SSH) protocol to protect ports that are used for debug and configuration operations. SSH implements a standard protocol to encrypt console connections (e.g., Linux shell access) to prevent unauthorized viewing or operations. This substantially extends the protection offered by a simple debug port password. However, it may be too complex to implement on smaller embedded systems. But on larger OS-based systems, this may be straightforward to implement because the necessary resources for SSH are often already present.

Public Key Exchange

Some applications won't permit pre-shared keys, such as when the terminal device can't have the key configured at the factory, the necessary field-installation expertise is unavailable, or there is no key-distribution system available. Public key exchange (PKE) is an ideal solution in these instances, but it carries considerable complexity. With PKE, one of several methods is used to select and combine two large numbers, then send one number and the resulting combination to the recipient. The recipient can then derive a session key which is known to the sender and is used to encrypt/decrypt traffic.

While technically complex and potentially too resource-intensive for an embedded system, PKE can actually simplify system deployment and operation because the sender and receiver don't need prior knowledge of one another and manual configurations can be minimized. This approach is often used on Linux-based systems that communicate using IP, because the necessary resources for PKE are often already present.

Security threats to embedded terminal devices in IoT systems are increasingly common, as attacks have become easier to carry out.

Transport Layer Security

Transport Layer Security (TLS) is the current standard that covers the widely implemented Secure Sockets Layer (SSL) protocol. It provides a standard framework for PKE and encryption to secure traffic between devices. However, for resource-limited embedded systems, the memory and processing requirements it imposes on the TCP/IP stack may be impossible to support. For this reason, TLS is often used on larger embedded systems (e.g., those running Linux) where communication occurs in IP sessions such as TCP. Even smaller embedded system may have the resources to support TLS, but this must be carefully evaluated on a case-by-case basis.

Wi-Fi Protected Access (WPA2)

When an embedded terminal device uses Wi-Fi (802.11) for communication, the WPA2 suite of standards can be used to secure the communication channel. This widely deployed protocol allows interoperability of systems created by different design authorities. However, it is generally beyond the reach of smaller embedded systems unless specialized Wi-Fi-dedicated coprocessors are present. On larger OS-based (e.g., Linux) systems, WPA2 may be an attractive option in certain applications.



Summary

Security threats to embedded terminal devices in IoT solutions are increasingly common, as attacks have become easier to carry out. Threats can be classified into four types: confidentiality, service theft, data integrity and availability. FMEA and DREAD are effective analysis tools used to prioritize development resources when responding to security threats.

Terminal devices for IoT systems have unique security requirements and challenges, mostly due to resource limitations. Six core methods (packet encryption, message replay protection, message authentication code, debug port protection, secure bootloaders and pre-shared keys) are typically compatible with the unique needs of M2M terminal devices. Increasingly, four other methods (SSH, PKE, TLS and WPA2) can be used with smaller M2M terminal devices as available system resources expand.



About Digi International

Digi International (NASDAQ: DGII) is the M2M solutions expert, combining products and services as end-to-end solutions to drive business efficiencies. Digi provides the industry's broadest range of wireless products, a cloud computing platform tailored for devices and development services to help customers get to market fast with wireless devices and applications. Digi's entire solution set is tailored to allow any device to communicate with any application, anywhere in the world.

Key Takeaways:

- ✓ Security threats to embedded terminal devices in IoT solutions are increasingly common.
- ✓ Threats can be classified into four types: confidentiality, service theft, data integrity and availability.
- ✓ Six core protection methods (packet encryption, message replay protection, message authentication code, debug port protection, secure bootloaders and pre-shared keys) can be used on the smallest systems.
- ✓ Four other methods (SSH, PKE, TLS and WPA2) can be used on larger systems.

All registered trademarks or trademarks are property of their respective owners.

Contact a Digi expert and get started today

PH: 877-912-3444
www.digi.com

Digi International

9350 Excelsior Blvd.
Suite 700
Hopkins, MN 55343

Digi International - Germany

+49-89-540-428-0

Digi International - Japan

+81-3-5428-0261

Digi International - Singapore

+65-6213-5380

Digi International - China

+86-21-5049-2199



/digi.international



@DigiDotCom



/digi-international

© Copyright 2014 Digi International Inc. All rights reserved. B3/1118