IoT DEVICE SECURITY BUILT-IN, NOT BOLT-ON

The 10 Security Factors Every Device Designer Should Consider

The Rising Tide of Security Threats

Limited only by designers' imaginations, the Internet of Things (IoT) is changing how people live. From medical devices and fitness trackers to tank sensors, smart thermostats, intelligent streetlights, water monitors, and more, the IoT is in more places than ever.

However, by relying on wireless networks, those hundreds of millions of IoT devices present a greater "attack surface," making them tempting frontline targets for competitors, hackers, disgruntled employees, and other bad actors. Unfortunately, the tools and techniques we've applied to PC/smartphone platforms often don't work well in the IoT, for several reasons:

- **RESOURCE LIMITATIONS** – Small-footprint IoT devices typically have far less battery power, processing speed and memory. They lack the power and sophistication required to support traditional security measures.
- **DATA COMPLACENCY** – Many companies view the data in their IoT networks as mundane and having little intrinsic value outside the organization. But many breaches are motivated by other factors, such as competitive advantage, social status, or revenge. The data isn't the goal – the hack is.
- **AVAILABILITY OF TOOLS** – The tools and expertise to analyze and modify embedded/IoT devices are widely available – even to hobbyists.

Thanks to wireless connections, hackers don't need physical access to devices such as USB outlets or network ports.

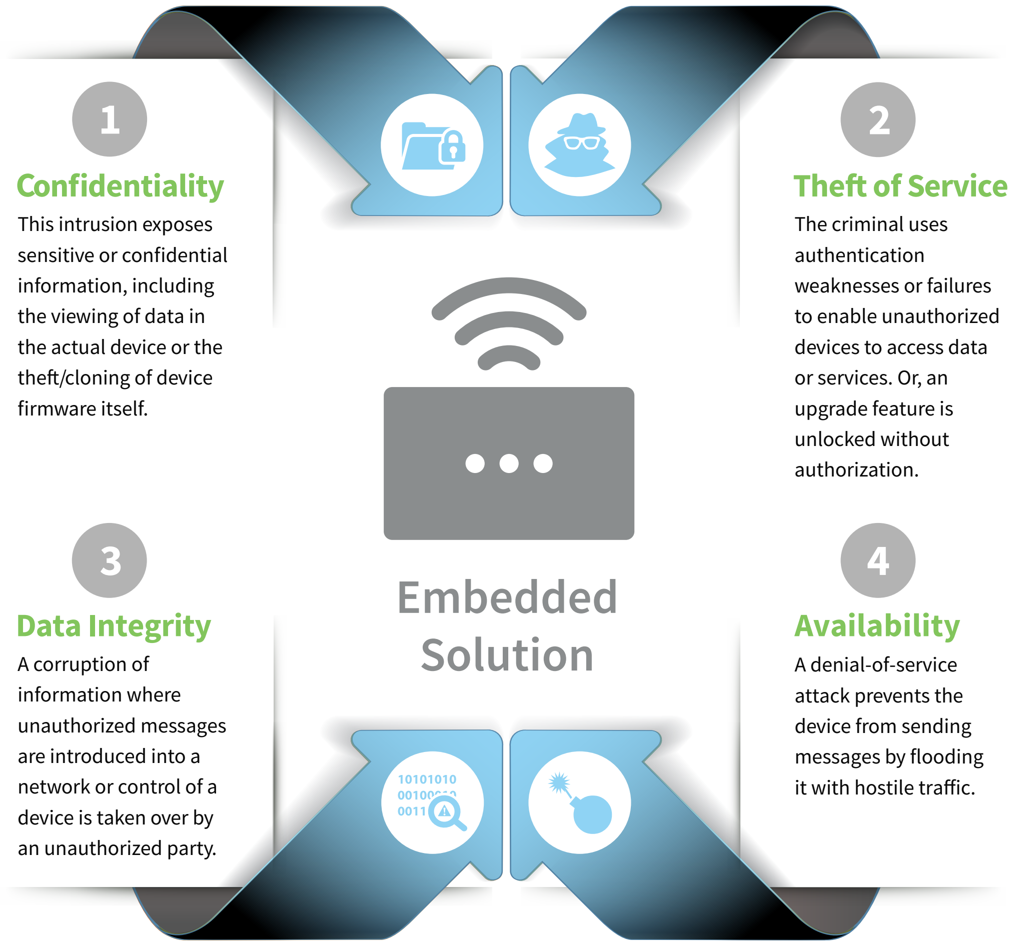
- **NO PHYSICAL ACCESS REQUIRED** – One of the advantages of the IoT is that devices can be remotely configured/upgraded without the need for dispatching a truck. However, thanks to wireless connections, hackers don't need physical access to devices such as USB or other I/O ports.
- **INTERFACE DIFFERENCES** – 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 allowing for security alarms to be overlooked.
- **HARDWIRED PORTS** – These provide unfortunate opportunities for compromise.

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.

But the risks of this rising tide of security threats are significant. Beyond reputational damage, competitive threats, eroding customer confidence, and safety challenges, regulators are paying increasing attention as well. For instance, security breaches that violate HIPAA regulations can lead to fines of $50,000 per violation. Credit card processors that fail to comply with the PCI DSS standard may be fined up to $100,000 per violation.

Distributed Denial of Service (DDoS) attacks are becoming more and more prevalent. These attacks may not necessarily be targeted at the average IoT edge device but a hijacking of a connected IoT edge device may be used to create a ‘BotNet’, a group of hijacked devices working together to work in unison to attack a central point on the IoT network or an external server/computer outside of the local network. Even if these attacks are not targeted at the local IoT network, they still pose multiple problems by preventing regular IoT work to take place or even simply draining the battery on a mobile IoT edge device creating maintenance cost for the administrators leaving them wondering why the battery didn’t last longer.

Four Types of Security Threats that Disrupt IoT Devices



1

Confidentiality

This intrusion exposes sensitive or confidential information, including the viewing of data in the actual device or the theft/cloning of device firmware itself.

2

Theft of Service

The criminal uses authentication weaknesses or failures to enable unauthorized devices to access data or services. Or, an upgrade feature is unlocked without authorization.

3

Data Integrity

A corruption of information where unauthorized messages are introduced into a network or control of a device is taken over by an unauthorized party.

4

Availability

A denial-of-service attack prevents the device from sending messages by flooding it with hostile traffic.

Security is a Balance Between Economic Cost and Benefit

Given enough time, money and expertise any system can be hacked, so it is important to design a system to deter an attacker by making it uneconomic (i.e. the cost or effort of an attack far outweighs any benefit to an attacker).

Types of attacks can be classified in terms of investment, the type of attacker and equipment used.

These range from:

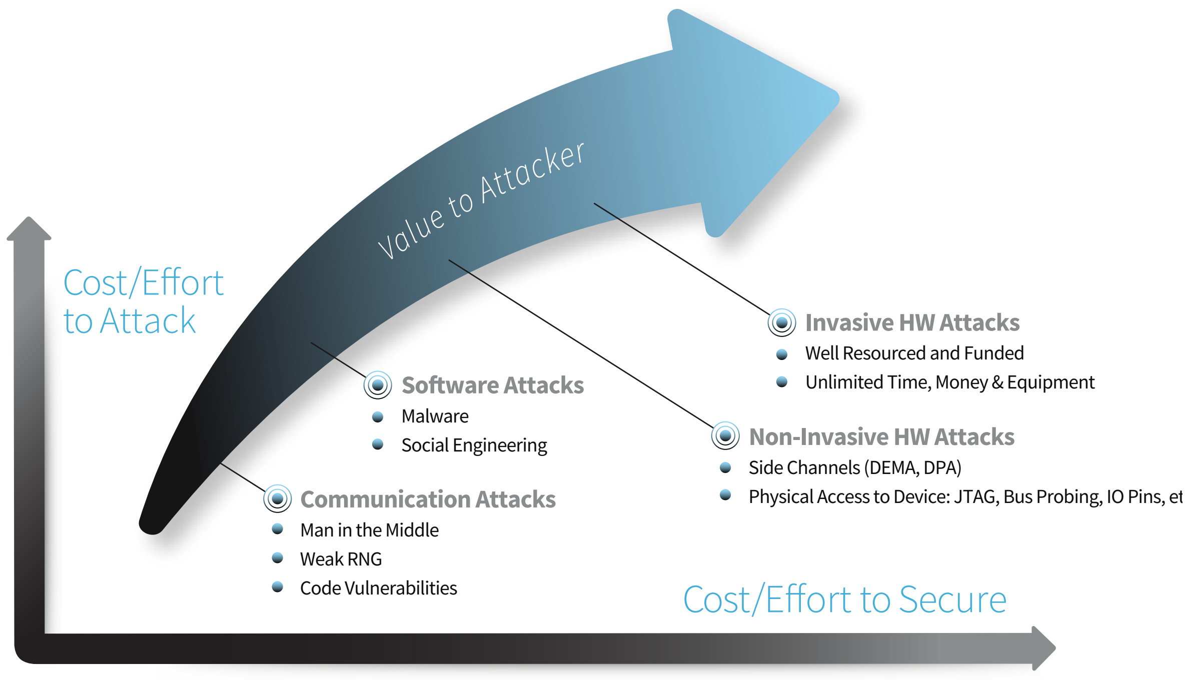
- **EXPENSIVE INVASIVE ATTACKS** (such as reverse engineering, or sophisticated micro probing a chip)

To lower cost:

- **PASSIVE SOFTWARE ATTACKS** (exploiting unintentional security vulnerabilities in the code)
- **COMMUNICATION ATTACKS** (e.g. exploiting weaknesses in the internet protocols, crypto or key handling)

Security is always a balance between economic cost and benefit, dependent upon the value of assets on the one hand and the cost of security features on the other.

The success of the Internet of Things will depend on data and services being protected, and when the security balance is right, it can open up new opportunities and markets.



Hacks can be categorized by the cost and effort involved

The 10 Security Techniques Every IoT Designer Should Consider

For design engineers who are striving to enhance the security of their IoT devices, there are numerous options at hand. In the following pages, we describe 10 strategies that can have a direct impact on improving device security.

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

1 Packet Encryption

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

2 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). The recipient enforces a rule that messages are only accepted 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. This can also be part of an encryption mode that will use this information within a block cipher. Examples of this is the AES counter mode block cipher.

3 Message Authentication Code

In this method, we run a cipher or 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 provide explicit protection from tampering and enables some systems to safely use clear-text messages. For example, we can use this method for systems that transmit non-confidential data (e.g., air temperatures) that nonetheless must not be tampered with. This is another low-complexity method that is useful for many types of embedded systems.

4 Debug Port Protection

Hardware ports used for configuration, control, and analysis (e.g., JTAG ports and serial logging ports for firmware development and debugging) are also vulnerable and tempting targets for security attacks. To start, these ports can be protected with a different factory password per unit before further actions are allowed. Of course, the better move is to internally disable these ports in field-deployed units.

5 Secure Bootloader

Even for a development team with unrestricted access to required technical information, it can be daunting to correctly build and load firmware into a resource-limited embedded device, which makes it unlikely you'll experience a successful attack based on a malicious firmware modification. 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. One best practice is to configure the device to check for a HMAC signature in the firmware image during startup to ensure it is authorized to run on the product. The image may also be encrypted for further protection. Secure bootloader solutions demand careful management of keys and support for debugging.

One best practice is to configure the device to check for a HMAC signature in the firmware image during startup to ensure it is authorized to run on the product.

6 Pre-Shared Keys

Secure IoT communications requires access to compatible keys. The use of pre-shared keys (PSKs) minimizes the demands on the resource-constrained device. Keys can be transferred through an independent, secure channel and then manually entered into the terminal device. While the overall system to share the keys may have some complexity, the demands on the actual terminal device are minimal. When allowed by the application.

7 Secure Shell

The Secure Shell (SSH) protocol protects ports 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 protection beyond a simple debug port password. This can often be too complex to implement on smaller embedded systems. But it's quite straightforward and feasible on larger OS-based systems because the necessary resources are typically present.

8 Public Key Exchange

Sometimes, pre-shared keys aren't a viable option, 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. In these instances, public-key exchange (PKE) is an ideal solution – thought it adds considerable complexity. With PKE, one of several methods is used to select and combine two large numbers, and then send one number and the resulting combination to the recipient. The recipient derives a session key that is known to the sender and this establishes a channel 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 over IP, because the necessary resources for PKE are often already present.

9 Transport Layer Security

Transport Layer Security (TLS) is the current standard for 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 for the TCP/IP stack may be impossible to support. That's why 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 systems may have the resources to support TLS, but this requires careful evaluation.

10 Wi-Fi Protected Access (WPA2)

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

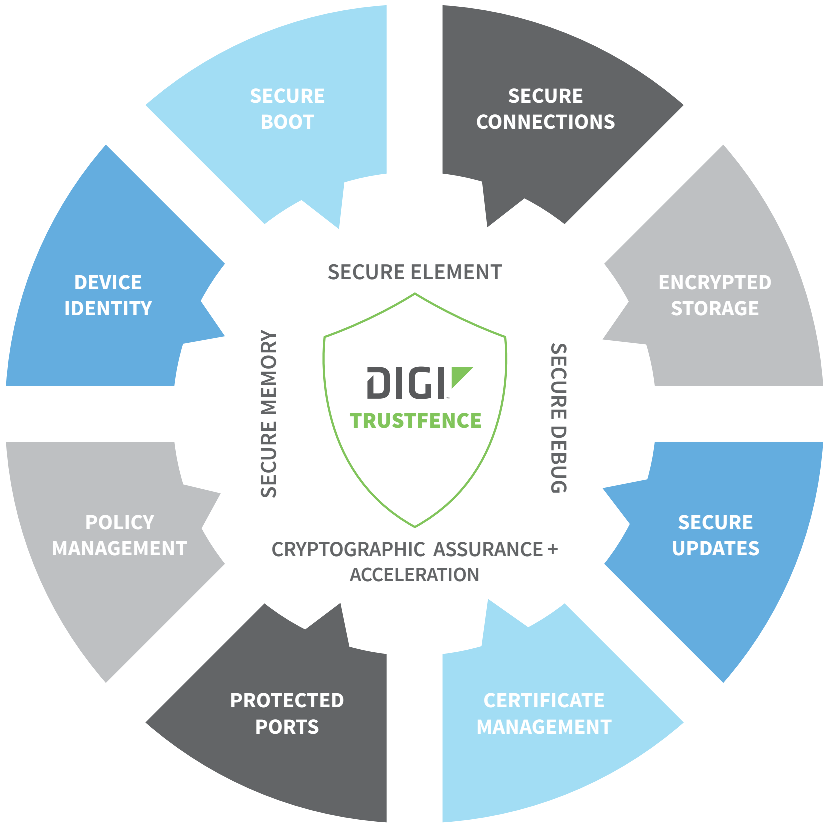
Digi TrustFence® Featuring the Digi ConnectCore® 6 and 6UL and NXP i.MX 6 and i.MX6 UL® Applications Processor

To help designers and builders effectively respond to the IoT security mandate, Digi offers Digi TrustFence, a fully integrated, tested, and complete Linux device security framework for the Digi ConnectCore 6 and 6UL system-on-module solutions featuring the NXP i.MX6 and i.MX6 UL applications processors.

By leveraging multiple H/W security components of the i.MX series, Digi TrustFence simplifies efforts to build secure, trusted, and reliable connected products; speeds your time to market; and lets you focus on your core competency. You gain immediate access to critical features such as secure connections, authenticated boot, encrypted data storage, access-controlled ports, secure software updates, and seamless integration of the dedicated on-module Secure Element (SE).

Build connected, embedded products on Digi ConnectCore 6 for the NXP i.MX6UL to capitalize on out-of-the-box, integrated security with Digi TrustFence. The result: you can protect your brand's reputation. You focus on delivering products that take advantage of the benefits of connectivity. Digi TrustFence handles the security for you. Digi TrustFence offers:

- **SECURE BOOT** – TrustFence ensures only signed software images run on your device.
- **ENCRYPTED STORAGE** – Local file system encryption keeps your internal data safe.
- **PROTECTED PORTS** – Protected, access-controlled internal and external ports prevent unwanted "back doors."
- **DEVICE IDENTITY** – Root of trust, certificate management, and secure key storage protect the identity of your device.
- **DEVICE INTEGRITY** – Tamper-proofing and device-integrity monitoring with low-power support protect against physical intrusion.
- **SECURE CONNECTIONS** – Enterprise-level data encryption provides privacy for wired and wireless network connections.
- **FUTURE-PROOFING** – Digi platforms are built for longevity and long-life product lifecycles with availability for years to come.



Summary

Security threats to embedded devices in IoT solutions are increasingly common, as attacks have become easier to carry out. These can include confidentiality breaches, service theft, data integrity, and service availability. 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 NXP

NXP Semiconductors N.V. enables secure connections and infrastructure for a smarter world, advancing solutions that make lives easier, better, and safer. As the world leader in secure connectivity solutions for embedded applications, NXP is driving innovation in the secure connected vehicle, end-to-end security and privacy, and smart connected-solutions markets. Built on more than 60 years of combined experience and expertise, the company has 44,000 employees in more than 35 countries.

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:

- ✓ The IoT is connecting innumerable end devices in what were previously closed systems that are now being opened for the first time to remote access and control.
- ✓ There are six core methods to securing IoT devices: packet encryption, message replay protection, message authentication code, debug port protection, secure bootloaders and pre-shared keys.
- ✓ Increasingly, four other additional methods (SSH, PKE, TLS and WPA2) can be used to secure end devices.
- ✓ As urgency continues to grow for IoT device security, make sure your security is built-in.

Contact a Digi expert and get started today

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

Digi International
Worldwide HQ
11001 Bren Road East
Minnetonka, 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®

/digi.international @DigiDotCom /digi-international