BUILDING TRUST IN THE INTERNET OF THINGS

Developing an End-to-End Security Strategy for IoT Applications
EXECUTIVE SUMMARY

Recent security breaches in the Internet of Things (IoT) have brought to light the urgent imperative to protect IoT systems—and the people who depend on them—from external threats. Piecemeal approaches aren’t working. Effective security requires an end-to-end strategy starting at the design stage and extending throughout the application lifecycle. This paper explores the six key categories of threats, the variety of mitigation measures available, and how IoT developers can implement comprehensive, integrated security solutions without compromising system performance.

TABLE OF CONTENTS

Executive Summary ................................................................. 2
Driving Home the Need for IoT Security ..................................... 3
The Six Key Categories of Threats and How to Mitigate Them .......... 3
  Social Engineering ............................................................... 3
  Attacks on Hosted Components ............................................. 3
  Hacked Device Software ...................................................... 3
  Physical Attack ................................................................. 4
  Network Compromise ......................................................... 4
  Security Misconfiguration ..................................................... 4
Pre-configuration: Increase Efficiency, Reduce Complexity ............. 5
Wind River Helix Device Cloud: Secure Device Management and Monitoring ...... 5
Conclusion ........................................................................ 5
Building Trust in the Internet of Things

Driving Home the Need for IoT Security

In July of 2015, a pair of cybersecurity researchers made news when they successfully took over the controls of a Jeep Cherokee on a highway by hacking into the car’s infotainment system. The news generated widespread alarm. Despite almost daily reports of cyberthreats to critical infrastructure, banks, or government agencies, the takeover of a private automobile on the road had a far more unnerving effect on average citizens who drive every day. Although the hack was by all accounts ethical (the hackers had notified the carmaker and the driver participated in the test), it proved how easily someone with malicious intent could harm an unsuspecting populace.

If the incident brought the Internet of Things into the public consciousness, it was also a wake-up call concerning the urgent need for security in IoT applications—or more precisely, for a comprehensive, holistic approach to security over the entire IoT system lifecycle. No doubt there was security built into critical software components in this particular vehicle, but it was more than likely implemented in each component individually, in a patchwork approach rather than holistically, leaving gaps that made the car vulnerable.

With such a wide variety of known security threat types and new ones emerging all the time, developers of IoT applications and devices can no longer rely on patchwork approaches to mitigation. They need to be thinking end-to-end rather than one-by-one. If one area is overlooked or weakened, the entire system is at risk. A comprehensive approach to security must take into account not only the entire IoT system—from edge devices to the network and the cloud—but also the entire system lifecycle, from development to deployment through operation and even to end-of-life.

The Six Key Categories of Threats and How to Mitigate Them

Whether they are state-sponsored criminals, “hacktivists,” or simply thrill seekers trying to prove something, intruders seem to have a countless array of techniques to find and exploit system vulnerabilities. Most security threats, however, fall into six categories. For each of these categories, IoT application developers have a number of mitigation methods at their disposal.

1. Social Engineering

This term refers to the ability of hackers to obtain security credentials from legitimate users through such techniques as phishing, “dumpster diving,” or even personal blackmail.

There is little technical recourse for social engineering techniques, as they tend to play on human behavior and psychology more than technical vulnerabilities. But that fact makes it even more important to adhere to certain fundamental safeguards, including:

- Strong passwords
- Frequent password updates
- Role-based permissions that restrict access to different areas of the system to specific users at designated times
- Following software update instructions when alerted to common vulnerabilities and exposures (CVEs) by the software provider

2. Attacks on Hosted Components

These attacks include SQL injection, cross-site scripting (XSS) and other techniques used to compromise access and authentication controls in cloud-based control systems that handle large volumes of sensitive data.

Countermeasures include:

- API authentication tokens
- Role-based access
- Application whitelisting to protect against “impersonation” at the cloud level

3. Hacked Device Software

Hackers who gain access to software at the device level can execute a variety of techniques to disrupt or take control of the system, including denial of service, malware installation, false identities, elevation of privilege, jailbreaking, and others.

Developers can employ a number of measures to mitigate this threat, such as:

- Secure boot: When power is first introduced to the device, the authenticity and integrity of the software on the device is verified using cryptographically generated digital signatures. This practice ensures that the device is not loading software that has been tampered with or maliciously inserted.
• **Software updates:** Keeping software up-to-date is a fundamental requirement for the secure operation of IoT applications. The responsibility usually lies with software providers to have a mechanism for providing software updates at scheduled intervals or when a CVE is identified.

• **Secure package management:** Software providers should be monitoring for CVEs, alerting customers before they become widely known, and providing updates to mitigate them. Secure package management (SPM) helps ensure the update is authentic using signature keys, thereby mitigating the threat of running software with a known vulnerability.

• **Integrity Measurement Architecture:** IMA is used to verify the authenticity of code running on the system and protect against offline disk tampering.

• **Software isolation:** Separating software components into “containers” ensures that a breach of one part of the system will not compromise any others. This practice effectively mitigates the threat of elevation of privilege, in which an attacker enters the system via a single component and gains higher-level authorization for access to others.

4. **Physical Attack**

This may involve the theft of a device in order to reverse engineer or replicate and steal credentials or other sensitive data. IoT applications often entail numerous connected devices that are widely dispersed and difficult to monitor individually, making them vulnerable to various forms of tampering.

While developers may not be able to ensure the physical security of a device, they can prevent hackers from using a stolen or compromised device to do further damage. Techniques include:

• **File system encryption:** In the event of a physical attack, file system encryption ensures that stolen data cannot be read or copied.

• **Trusted Platform Modules:** TPMs are a hardware-based security solution that stores encryption keys used to authenticate the hardware on which a host system is running.

• **Remote attestation:** Remote attestation technology is quickly gaining favor as a means of verifying the identities and integrity of different parts of an IoT system throughout its lifecycle. It leverages credentials that have been planted in devices pre-deployment. The devices can then be queried at any time through a management console to confirm their identities and attest that no one has tampered with them.

5. **Network Compromise**

Commonly called “man in the middle” attacks, hackers use techniques such as session hijacking to enter a network to disrupt, block, or alter communications between devices and their cloud-based controllers. Network vulnerability provided the Jeep hackers with their original point of entry, which ultimately led them into the car’s onboard system.

Securing the network is a critical concern in IoT application development and can be achieved with network encryption methods that protect data in transit by rendering it unreadable in the event of a network breach.

6. **Security Misconfiguration**

Perhaps the most common threat to IoT applications, the improper configuration of security elements will leave the system vulnerable to system-wide attacks intended to find and exploit weaknesses. The developer may have incorporated a number of the standard security features outlined above, but if they are not put together correctly, they render at best a false sense of security.

Avoiding the risk of misconfiguration requires an end-to-end, integrated security strategy at the design stage. It is not sufficient to simply check off a list of security technologies. Once developers have mapped out and analyzed the threat model across the entire system and system lifecycle, they can begin thinking about an overall mitigation strategy and methods to protect each area of potential vulnerability.

As for the “magic bullet” software that can scour a system and pinpoint all of its vulnerabilities? It does not exist. That’s why the time to start thinking about security in an application is before it is developed and deployed, as an integral part of the design. A true end-to-end security solution must encompass the entire application lifecycle from development through the eventual decommissioning of devices. Indeed, in order to ensure that a decommissioned device is truly “dead” and cannot provide intruders with a path into the system, end-of-life security functionality must be implanted in the pre-deployment stage.
PRE-CONFIGURATION: INCREASE EFFICIENCY, REDUCE COMPLEXITY

As the Jeep example illustrates, security is an imperative for virtually any IoT application in which human safety could be threatened by a system compromise. It is arguably a prerequisite for any application from which people expect safe and reliable everyday performance without a second thought. Why, then, is it so often treated as an afterthought, secondary to system performance rather than essential to it?

Developers face a big enough challenge simply building an IoT application that does what it is supposed to do. Security adds a layer of complexity, particularly as it must address a growing number of increasingly sophisticated threats. Without proper planning, building in security functionality can slow down development, drive up costs and, in some cases, impair the performance of a deployed application.

But there is a solution. Developers can build IoT applications on a platform using pre-configured, integrated software components in which many security issues have already been addressed. This takes the onus off developers to identify, source, and patch together different security technologies as development progresses, resulting in a much more efficient development process, much less system complexity, and a reduced risk of security gaps due to misconfiguration.

WIND RIVER HELIX DEVICE CLOUD: SECURE DEVICE MANAGEMENT AND MONITORING

Wind River® Helix™ is a portfolio of software and services designed to help developers, device manufacturers, system integrators, and operators build and manage IoT systems from the edge to the enterprise. Included in the portfolio is Wind River Helix Device Cloud, a ready-made platform for collecting data from edge devices, monitoring device performance, and managing all aspects of the device lifecycle. It incorporates a wide range of pre-configured features that enable developers to implement security measures across the device lifecycle at the design stage, including:

- Secure boot
- Device software update mechanism
- SPM
- Application whitelisting
- Network, data, and device encryption
- Embedded credentials and certificates
- Trusted Platform Modules
- Access permissioning
- Software isolation
- Integrity measurement

By providing pre-integrated security components, Device Cloud helps developers mitigate the risk of misconfiguration and implement security without delaying development or compromising system performance. Moreover, Wind River Professional Services consultants are available to assist with the configuration of security components based on the threat model assessment and the unique requirements of the application.

CONCLUSION

Security is imperative for IoT applications, for the protection of the machines they control, and for the people who depend on their reliable performance. In fact, security is so fundamental to IoT system performance that it needs to be integrated into system design. It cannot be implemented piecemeal, but requires a well-thought-out, end-to-end strategy encompassing all aspects of the application, from the edge to the enterprise, and all stages of the application lifecycle, from development to decommissioning.

Without question this adds to the challenge of IoT development. That challenge can be easily met, however, through the use of pre-configured software and security components that accelerate the implementation of security functionality and free developers to focus their attention on system performance.