Electronic Design

By COLIN DUGGAN, Founder & CEO,
BG Networks, www.bgnetworks.com

# Preventing and Detecting Cyberattacks on Connected Devices

**This article delves into the regulations and best practices for IoT device cybersecurity by design.**

Protecting an IoT device from cyberattacks is a matter of managing potential risks. Some vulnerabilities stem from failing to build-in appropriate security measures, such as secure encrypted communication, secure boot, and secure software updates. Others result from the use of default or easily cracked passwords. Addressing these issues requires proper implementation of appropriate security features.

A more challenging problem is eliminating security vulnerabilities that occur when open-source solutions or commercial third-party solutions are utilized in developing a product. These components may include vulnerabilities that are unknown to the developer. IoT devices increasingly rely on a combination of in-house developed software, open-source software, and commercial software from vendors.

This creates a complex software supply chain with a broad attack surface for malicious actors to exploit. It's critical to monitor for reported vulnerabilities in these software components and best practice to add real-time monitoring and reporting for a defense-in-depth approach to supply-chain security.

New regulations and standards are in place that mandate cybersecurity requirements for IoT devices. Some of the regulations that IoT device designers should be aware of include:

- UNECE Regulations UN R155 & UN R156 for Vehicle Cybersecurity
- ISO 21434 Road Vehicle Cybersecurity Standard
- Cybersecurity in Medical Devices: Refuse to Accept Policy from March 29, 2023
- EU MDR Medical Device Regulation
- ISA/IEC 62443 and NIST Cyber Security Framework (CSF) guidelines for the security of industrial automation and control systems
- UK Product Security and Telecommunications Infrastructure Act 2022
- EU Cybersecurity Resilience Act

**Software Supply-Chain Attacks**

Software supply-chain security isn't new, but it's more critical now than ever. Its significance is highlighted by several recent software supply-chain attacks with widespread impact.

In 2019, attackers infiltrated SolarWinds' networks and inserted malware within a software component called Orion. This malware was then unknowingly distributed as a legitimate, digitally signed SolarWinds update starting in March 2020. The software supply-chain attack remained undetected until December 2020, giving the attackers months of undetected access to affected systems.

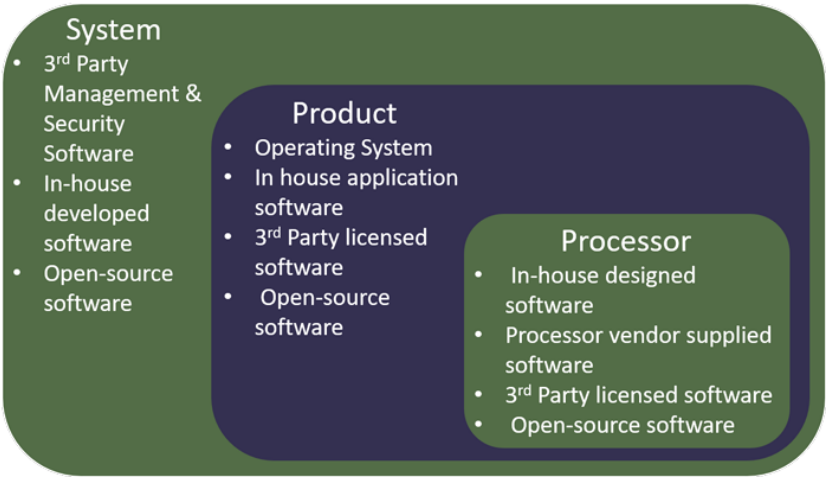
The SolarWinds attack is an example of a supply-chain breach with which attackers purposefully insert malicious code upstream in the supply chain. However, software supply-chain security also can result from using open-source software containing vulnerabilities.

Log4Shell refers to a software vulnerability in Apache's Log4J library, which allows attackers to execute arbitrary code on impacted devices and systems. While the vulnerability existed in the Log4j library since 2013, it was only publicly disclosed and mitigated in December 2021. That same month, attacks exploiting that vulnerability were detected.

Because Log4j is widely adopted as the standard logging method for Java applications, it was a huge challenge to find all instances of its use within a corporation. This was especially the case for those that did not have a list of software components used by their software applications.

**Software Supply-Chain Security**

In May 2021, in response to high-profile cyberattacks such as the SolarWinds attack, the President of the United States issued Executive Order 14028 outlining cybersecurity requirements. This executive order specifically includes directives to establish guidelines and standards for software supply-chain security. The National Institute of Standards and Technology (NIST) subsequently released an update to their foundational supply-chain risk-management guidance in NIST Special Publication SP 800-161r1, titled "Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations."




**1. The software supply chain for IoT products is complex, making it difficult to trace all components.**

A key element of NIST guidance is a requirement for software providers to maintain a software bill of materials (SBOM). An SBOM serves as a "nutrition label" for software, identifying all software "ingredients" and providing important information (e.g., version number, licensing terms) about each component *(Fig. 1)*. SBOMs can be used to determine the existence of known vulnerabilities in software components, assess the availability of updated versions of software, and verify the integrity of software downloads.

Dependency Track, a free open-source solution, is one of several SBOM analysis platforms that enables software developers to continuously analyze their software. These platforms automatically cross-reference vulnerability databases against the SBOMs and generate reports of known vulnerabilities present in the software build. Automation is critical to establishing processes that effectively leverage SBOM information on a consistent basis (e.g., every time there is a new build).

**Automating Supply-Chain Security**

For IoT software developers, efficiently creating an SBOM that identifies all software dependencies within a build is a challenge. Support for automatically building SBOMs using standard formats is inconsistent across build systems. SPDX and CycloneDX are the two leading SBOM formats, but many tools only support one format. This results in incompatibilities between build systems and tools for analyzing SBOMs.

SPDX was originally created for managing open-source software licenses, while CycloneDX was designed to enable SBOM creation for vulnerability management. However, their functionality has been converging and both now enable creation of SBOMs. There are two reasons why an IoT developer might choose SPDX over CycloneDX:

- SPDX is more common and currently is supported by more tools.
- SPDX has a strong licensing component, which is important to get right when using open-source software in commercial products.

Integrating build systems with SBOM analysis tools is essential. Not only do these tools detect known vulnerabilities at the time of the build, but they also continuously monitor vulnerability databases for newly disclosed vulnerabilities. If a new vulnerability is discovered in a component you're using, a notification will be generated, allowing you to proactively address the problem.

One such tool for IoT developers is BG Networks' Vulnerability Scanning solution for embedded Linux. This tool integrates with the Yocto build management system to automatically produce an SBOM for each new build. It then utilizes Dependency Track to scan for known vulnerabilities, ensuring that software projects comply with NIST's supply-chain security guidelines.

**Device Hardening and Mitigating Supply-Chain Vulnerabilities**

Supply-chain analysis can identify known vulnerabilities within the software components of an IoT device, and even provide information on the availability of new versions of those components with fixes to mitigate those vulnerabilities.

In many cases, newer versions of the software components are available with fixes for known vulnerabilities. Updating to the newer version will, in these instances, eliminate the vulnerability. In other cases, developers may need to address

| Security Vulnerability | Description |
|---|---|
| Lack of security controls | Interfaces without security controls (i.e. un-authenticated web interfaces, unauthenticated firmware updates) |
| Weak authentication | Default or weak passwords. Outdated, vulnerable encryption (i.e., TripleDES, MD-5 hashing, SHA-1, RSA-512). |
| Supply chain | Any vulnerability that results from the use of an open-source or 3rd party software solution that contains vulnerabilities. |
| Software bugs | Software bugs that result in buffer overflows or other errors that can be exploited by hackers. |

**2. Common security vulnerabilities in IoT devices range from a lack of fundamental security controls, best practices, and software supply-chain vulnerabilities.**

the problem by correcting the code themselves or by utilizing an alternative software solution.

In some instances, updating to a newer version of software will not eliminate all known vulnerabilities. Worse still, unknown vulnerabilities will almost certainly remain in a large and complex system. The Log4j vulnerability, for example, remained undetected for years before its discovery. Additional security measures are required to harden devices against such vulnerabilities.

As each IoT device is unique, a threat assessment/risk analysis (TARA) should be performed to guide the design of security features for every device *(Fig. 2)*. While a TARA provides specific guidance for each device, several general principles apply to all IoT devices. Every IoT device should include the following security features:

- Hardware root of trust
- Secure boot
- Secure software/firmware updates
- Cryptographically strong device identity
- Secure communication protocols
- Encryption of critical data at rest

Implementing all of these security features can be complex and time-consuming. Time-to-market is often crucial, leading to pressure to eliminate unneeded features. Unfortunately, this often results in products being released without critical security features. New solutions have been developed, such as BG Networks' Security Automation Tool, that automates the implementation of many of these security features. By using such tools, the development time for key security features can be reduced to just a few weeks.

**Real-Time Attack Detection for IoT Devices**

Utilizing supply-chain analysis and hardening IoT devices with critical security features represent significant strides in building secure IoT devices. These steps substantially reduce the number of vulnerabilities exploitable by potential hackers. However, regardless of how careful a development team is, no device will be perfectly secure.

IoT devices have vulnerabilities and will inevitably be attacked, so it's crucial for IoT device operators to have visibility into attacks against their devices. Real-time attack detection and response form the cornerstone of any enterprise security solution. Likewise, IoT devices must be capable of detecting an attack and providing security alerts when under attack.

A host-based software anomaly detection solution, such as BG Networks' AnCyR, combines statistical, probabilistic, and machine-learning algorithms to accurately detect attacks with low false positive rates. Deployed as a software agent, AnCyR builds a model of how an IoT device's software runs. Once the learning phase is complete, it monitors the execution of software on the device and detects any deviations from the expected model.

Cyberattacks alter the behavior of the software on the device; therefore, any change is potentially a cyber-attack. When an attack is detected, the solution sends an alert to a security information and event management (SIEM) system or other networking monitoring solution.

Designed for use in small-footprint IoT devices, the solution operates with minimal overhead. It can function with just tens of kilobytes of memory and less than 10% processing overhead.

In the enterprise world, such solutions are known as endpoint detection and response (EDR). An equivalent term for IoT is intrusion detection/prevention solutions (IDPS). When a device is targeted by a cyberattack or compromised, resulting in an attacker controlling the device or exporting data, its behavior changes. It may run malicious software as part of a DDoS network or crypto-mining bot network, export private data, or engage in other malicious activities.

By detecting and reporting this anomalous behavior, network operators can take action to mitigate and prevent the spread of cyberattacks on critical infrastructure. This is

an important additional layer of security. It doesn't replace features such as secure boot and secure communication or eliminate the need for supply-chain security. Rather, it provides real-time protection against any remaining vulnerabilities, even those that aren't yet known to developers.

**Summary**

Recent legislation and industry standards now require manufacturers to be proactive in adding security to their devices. Legislative requirements in the automotive industry mandate adherence to cybersecurity guidelines for anyone wishing to sell a vehicle into Europe. In the U.S., the FDA requires medical-device companies to implement cybersecurity measures. Consumer, energy, and industrial markets each have their own standards that must be followed.

These new regulations have led to significant advances in cybersecurity for IoT devices. Many new devices now utilize secure boot, secure software updates, strong authentication, and secure communication protocols to help protect these devices against cyberattacks. Supply-chain security management and real-time attack detection are also required to ensure the security of these devices.

Making sure that a device is both secure and compliant with cybersecurity mandates requires performing a TARA for the device, managing supply-chain security, and implementing key cybersecurity features. Cybersecurity features can be time-consuming to implement, but automated tools helps significantly reduce this burden for developers.

*Before founding BG Networks in 2020, Colin Duggan worked at Analog Devices (ADI) for 29 years in various engineering, management, and marketing leadership roles, managing teams located in U.S., China, Europe, and India. Colin's experience includes work in automotive, consumer, industrial, and aerospace & defense markets.*