PRAETORIAN

WHITE PAPER

The 9 Most Prevalent Internet of Things Security Issues and How to Avoid Them

Observations from over 20,000 hours of IoT security assessments

INSIDE THIS REPORT

INTRODUCTION

Common Vulnerabilities Across IoT Ecosystems

In the past two years Praetorian has performed more than 20,000 hours of IoT security assessments. While every assessment is unique, there are several types of vulnerabilities that we see occur over and over, across widely different products and use cases.

We first noticed these patterns in 2016, and in early 2017 we published a white paper documenting the most common issues that we observed at that time. In the year and a half since, our perspective on the most common issues has evolved as the IoT marketplace has evolved, but we still see many of the same issues.

20,000+

HOURS OF IOT SECURITY ASSESSMENTS
IN THE PAST TWO YEARS

What's striking about this is just how common these issues are. The 20,000 hours of assessments included every portion of an IoT ecosystem – apps, APIs, devices, portals, platforms – in different combinations. They also included a wide swathe of industries and use cases:

- Consumer appliances
- Industrial automation
- Connected vehicles
- Connected locks and access control
- Power and SCADA systems
- Medical devices
- IoT platforms
- ATMs and kiosks
- Building automation

SECURITY ISSUES

1 Authorization Issues

Over the past year, we have discovered an increasing number of authorization issues, or locations where a user's permission to perform a specific action is not checked by the application. This allows an attacker to perform actions for which they do not have permission.

Architectures made up of many microservices are particularly prone to this problem. We've seen multiple cases in which the architecture performs authentication checks on all users, but not authorization.

Solution

For each request, the system needs to verify not only whether the user is authenticated, but whether the user has permission to perform the requested action. In many cases, this is as simple as verifying that the user has requested an action affecting their own account or resources.

2 Platform / Vendor Weaknesses

There's a growing market of service providers that provide a backend and SDKs for IoT device developers. These offerings can decrease development costs and time to market. However, these vendors face market pressures to make their platforms as accessible as possible, which can place business decisions at odds with security.

Solution

Customers should be proactive in asking vendors for third-party assessments. Ask vendors about their secure development lifecycle as part of the evaluation process. Don't be afraid to ask hard questions and challenge your vendor on the adequacy of their responses.

SECURITY ISSUES

3 User Information Leakage

We've increasingly seen issues in which APIs leak customer information. Most commonly, the leaked information yielded is email addresses, but server responses often include related information. We've seen profile pictures, full PII (name, address, email, phone, social), and account details leaked this way.

Risk: There's immediate brand and reputational risk from these vulnerabilities. This leaked information can also be a stepping stone to further attacks, such as enabling social engineering attacks against your customers.

Solution

Identify all endpoints that provide customer information. For each endpoint, verify that it has appropriate access controls. Perform a scrub as part of QA to ensure endpoints only provide intended information.

4 Hardcoded Secrets (Keys, Credentials)

"Hardcoded secrets" refers to things like default credentials, keys, or passwords that are set in source code. Since they are set in the code itself, these secrets are shared between products. Once a product is released, attackers can often recover these secrets from devices in their possession.

Compromised credentials can create opportunities for trivial remote compromise. We've recently seen attacks automated and used by worms to spread across IoT devices.

Solution

Use unique secrets for each device. Store secrets used by web services in environmental variables – don't include them in source code.

SECURITY ISSUES

5 Outdated Third-Party Components

Preventing the use of outdated services or libraries tends to be an especially vexing problem for device manufacturers. Managing an inventory of which third-party components are in each version of each device is not trivial administrative overhead.

The result is that many products contain vulnerabilities due to the third-party services that were included. The impact can include severe issues like remote code execution, authentication bypasses, or information leakages.

Solution

Maintain an inventory of third-party components. Monitor for new vulnerabilities and patches. Build your update functionality so that it's automatic and requires little or no user interaction. Consider compiling custom binaries that include only the functionality you need.

6 Insufficient Account Protections

This issue appears across our assessments. The problem often results from several lower risk issues appearing together – weak password policies, lack of lockout, and user enumeration. When two or more of these three issues are present, attackers can launch automated brute-force attacks.

Reality: Due to unfortunate statistics about people's password practices, this almost always succeeds in production environments. Attackers commonly search breached password dumps for users and test for reused passwords.

Solution

Implement account AND IP-based lockouts or throttling. Inform users by email when their account is accessed from a new device. Consider using a blacklist to prevent users from setting vulnerable passwords.

SECURITY ISSUES

7 Resource Enumeration

Often a system needs some sort of schema to refer to ecosystem components other than users – things like devices, stored files, or organizations. When these values are sequential or otherwise predictable, it can facilitate attacks on the ecosystem.

This issue is not an explicit problem on its own, but knowledge of resource identifiers is often a prerequisite for other attacks. Denying an attacker the knowledge they would need to perform an attack can prevent it altogether or limit its blast radius.

Solution

Use random, non-deterministic values for resource identifiers. Many frameworks provide built-in functionality to support this.

8 Debug Services Enabled

We often find production devices that have debug services enabled through things like JTAG, UART, serial, or USB. The purpose of these services is to enable developers or technicians access either to control the device directly or receive debugging information.

For an attacker, this often presents a quick path to compromise of the device. This then provides an avenue to pull firmware, identify secrets, and generally identify other vulnerabilities in the device.

Solution

Inventory all mechanisms enabled during development – UART, JTAG, serial, USB, SD card readers, adb, telnet, ssh, etc. Plan steps near the end of development to disable all such services not intended for production. Removing physical interfaces is insufficient – ensure debugging services are disabled at the hardware or software level.

SECURITY ISSUES

9 TLS Weaknesses

Transportation Security Layer weaknesses are endemic across our assessments. We frequently see TLS servers configured to use old versions of protocols, weak algorithms, or weak keys. These poor configurations can allow a suitably positioned attacker to decrypt sensitive communications.

Good News: This problem typically is easy to change with a configuration tweak. The sophistication required of the attacker can range from nation-state to moderately skilled, depending on which misconfigurations are made.

Solution

Unless you have a power or processing limitation, default to encryption with TLS. Don't try to implement your own encryption – rely on mature open source libraries or core OS functionality. Mozilla offers a great service to provide secure TLS configurations based on your use case.

CONCLUSION

Development Teams Face Similar Challenges

These commonalities reaffirmed the observation we'd made in 2017: Development teams, regardless of their industry and product, struggle with similar problems. Despite all the differences across industries and use cases, the same nine issues appeared again and again.

Key Takeaway: The risk of these and other issues will always be context dependent. A low-powered device that only collects telemetry is different from a device that records video or audio in people's homes.