Initial Threat Overview

Dhruv Sunil Bhatia
24CSB0A19
CSE - A
Q119 - Secure OTA Update Compiler

Threats Addressed

Threat 1: Malicious Firmware Installation

An attacker attempts to install tampered or unauthorized firmware if update authenticity is not verified.

Impact:

- Device compromise
- Malware installation
- Loss of control over device behavior

Compiler Enforcement: Rejects update logic where firmware installation is reachable without signature verification.

Threat 2: Rollback Attack (Firmware Downgrade)

An attacker forces installation of an older but authentic firmware version containing known vulnerabilities.

Impact:

- Reintroduction of patched vulnerabilities
- Exploitation of known CVEs

Compiler Enforcement: Rejects update logic that does not enforce version monotonicity before installation.

Threat 3: Unauthorized Update Source

Firmware updates originate from spoofed or attacker-controlled sources if origin trust is not validated.

Impact:

- Malicious firmware delivery

- Integrity and trust failure

Compiler Enforcement: Rejects update logic where trusted source validation is missing or bypassable.

Threat 4: Information Leakage Through Debug Logs

Sensitive update information may be exposed through logs.

Impact:

- Exposure of firmware metadata
- Easier targeting of vulnerable firmware

Compiler Enforcement: Rejects update logic containing unsafe logging within the update routine.

Threat 5: Weak Cryptographic Validation

Use of weak cryptographic primitives in firmware validation.

Impact:

- Integrity verification bypass
- Spoofed firmware updates

Compiler Enforcement: Rejects update routines invoking banned weak cryptographic APIs.

Problem Definition

Dhruv Sunil Bhatia
24CSB0A19
CSE - A
Q119 - Secure OTA Update Compiler

Firmware updates in IoT and embedded devices are commonly delivered through OTA mechanisms, allowing remote upgrades without physical access. While OTA updates are essential, insecure update logic introduces significant security risks.

Insecure firmware update code may still compile successfully even when missing critical security checks such as signature verification, rollback prevention, and trusted source validation. This enables attacks such as malicious firmware installation and downgrade attacks.

This project proposes a compiler-based enforcement approach where firmware update security requirements are treated as compile-time correctness rules. The compiler statically analyzes update logic and rejects programs violating mandatory security invariants.

Objective

To design and implement a compiler extension that enforces firmware update security invariants at compile time, ensuring firmware installation code is generated only when all required security checks are present.

Scope Boundaries

Dhruv Sunil Bhatia
24CSB0A19
CSE - A

Q119 - Secure OTA Update Compiler

The compiler enforces the following mandatory security invariants:

1. Signature verification enforcement
2. Rollback protection via version monotonicity
3. Trusted update source validation
4. No debug or logging of sensitive update data
5. Cryptographic policy enforcement against weak APIs

If any invariant is violated on any control-flow path leading to installation, the compiler generates a compile-time error and prevents firmware generation.