Formal Invariant Definitions

Dhruv Sunil Bhatia
24CSB0A19
CSE - A

Q119 - Secure OTA Update Compiler

1 Overview

This document formally defines the security invariants enforced by the Secure OTA Update Compiler. These invariants must hold on all control-flow paths leading to firmware installation within the `updateFirmware()` routine.

2 Invariant 1: Verification Dominance

Within `updateFirmware()`, any call to `install(pkg)` must be dominated by a call to `verify_signature(pkg)`.

Formal Definition:
For every control-flow path leading to `install(pkg)`, there exists a `verify_signature(pkg)` executed before `install(pkg)`.

3 Invariant 2: Rollback Protection

Before `install(pkg)`, there must be a condition ensuring:

$$\texttt{pkg->version} > \texttt{current\_version()}$$

Formal Definition:
Installation is reachable only under the predicate:

$$\texttt{pkg->version} > \texttt{current\_version()}$$

4 Invariant 3: Trusted Source Validation

Before `install(pkg)`, the update source must be validated using `source_is_trusted(pkg)`.

Formal Definition:
For all execution paths reaching `install(pkg)`, the predicate

$$\texttt{source\_is\_trusted(pkg)} = \texttt{true}$$

must hold.

5 Invariant 4: Logging Safety

No sensitive logging operations related to update package metadata are allowed inside `updateFirmware()`.

Formal Definition:
No calls to known logging APIs (e.g., `printf`, `fprintf`, `log`, `syslog`) may occur within the scope of `updateFirmware()`.

6 Invariant 5: Weak Crypto Ban

Weak cryptographic APIs such as `MD5`, `SHA1`, or `rand()` are prohibited inside `updateFirmware()`.

Formal Definition:
No calls to banned cryptographic functions may appear within the scope of `updateFirmware()`.