Secure Plugin for Automated Software Updates using Public Key Infrastructure for Embedded Systems

Malombe Victor
Digital Learning Assistant, @iLabAfrica – Strathmore University
Quick Overview

- Introduction
- Research Objectives
- Problem Statement
- Literature Review
- Solution
- Conclusions and Recommendations
- References
Introduction

• **Embedded System** - Computing system built into a larger system, designed for dedicated functions (Papp, Ma, and Buttyan, 2015).

• **Single board computers** - Used to build embedded systems.
  - Shipped with vulnerabilities
Problem Statement

- Most embedded device software is not updated after deployment.
- Insecure update techniques
- Design limitation – Minimal user interaction
Research Objectives

To investigate how software updates are delivered and identify gaps and challenges in updating software in embedded systems,

To design and implement an automated PKI based updates plugin for embedded systems, and

To investigate the suitability of PKI in delivering software updates securely,

To test and validate the PKI based updates plugin for authenticity and integrity.
Literature Review (Software Update Techniques)

SSH – NO SERVER AUTHENTICATION (TURNER, 2014).

PORT FORWARDING EXPOSES A NETWORK PORT ON A PRIVATE LAN TO THE PUBLIC INTERNET (RASPBERRY PI FOUNDATION, 2018).

FTP CLIENTS – TELNET ISSUES (BOE & ALTMAN, 2002).
Literature Review

• Requirements for software updates (Simmonds, 2016), and (Farrell & Tschofenig, 2017)
  • Secure
  • Robust
  • Atomic
  • Failsafe feature
  • Scheduling and distribution
  • Scalability
Solution

Auto Update Plugin System Architecture

1. TLS Handshake (Server authentication)
2. Check & download updates
3. Close TLS session
4. Apply updates on local repo

Client

Certificate Authority

TLS VPN Tunnel

Git Server

Server Cert Verification

AES Encryption
Solution

Use Case Diagram
Conclusions

- IoT developers use unsecure methods
- Lack of automation leads to more administration time and effort
- Existing tools are platform depended and not free or open source
- The aim of the developed system is to automate the update process of any custom software of single board computers in a secure way.
Recommendations

- IoT developers should be concerned about security risks during software updates.

- **Public awareness** to the existence of this open source tool.

- The open source community is encouraged to **evaluate, critique and give recommendations** to improve this tool.

- Developers using private CAs and self-signed certificates should **install git server locally** and point to auto-update tool to it.

- Integration with the single board computer **default software base**.
Suggestions for Future Research

- SECURING DOWNLOADED UPDATES.
- ADD SECURITY INTRUSION DETECTION AND ALERT CAPABILITIES.


• Simmonds, C. (2016). Software update for IoT. 2net Ltd.
Questions?
Extra Slides (Post Presentation)
This research focus only on network security, specifically on transport layer of the networks communications stack. This is achieved through Transport Layer Security cryptographic protocols. This only applies to data in transit, but not protection of systems or stored data.
Recent IoT Attacks

Linux.Darlloz Worm (Nov 2013)

- PHP vulnerability
- Script preloaded with default usernames & passwords
- Starts a HTTP Web server on port 58455 in order to spread
- Primary objective was to mine cryptocurrency
- At least 31,716 identified IP addresses in 139 regions were infected

Mirai Malware (September 2016)

- Uses a list of 62 common default usernames and passwords to gain access
- The botnet was responsible for a 600-Gbps attack targeting Brian Krebs’s security blog (krebsonsecurity.com)
Top 10 IoT vulnerabilities (2016)

1. Insecure Web Interface
2. Insufficient Authentication/Authorization
3. Insecure Network Services
4. Lack of Transport Encryption/Integrity Verification
5. Privacy Concerns
6. Insecure Cloud Interface
7. Insecure Mobile Interface
8. Insufficient Security Configurability
9. Insecure Software/Firmware
10. Poor Physical Security
Update Checking and Delivery

1. Check & download updates (git fetch)

2. Apply updates to app (git merge)
Public Key Infrastructure Operations

Certificate Enrollment

OOB Authentication of submitter Public Key

CA admin

Certificate request

Signed certificate

Web server admin

POTS

PKI for Automating Updates in Embedded Systems
Public Key Infrastructure Operations
Authentication Using Certificates

Private Key (Bob)
Certificate (Bob) with Public Key (Bob)
Certificate (CA) with Public Key (CA)

Certificate (Bob) with Public Key (Bob)

Bob

Certificate (Alice) with Public Key (Alice)

Certificate (Alice) with Public Key (Alice)

Certificate (CA) with Public Key (CA)

Private Key (Alice)
Public Key Infrastructure Operations

Certificate Revocation

1. Client sends certificate to the SSL proxy
2. SSL proxy sends request for certificate revocation status
3. OCSP responder reports certificate revocation status
4. If certificate not revoked, BIG-IP system forwards packet to server
Public Key Infrastructure Operations

SSL / TLS

1. ClientHello (TLS Protocol, client’s random value, supported cipher suites, suggested compression methods)

2. ServerHello (Chosen TLS protocol version, server random value, CipherSuite, compression method)

3. Server Certificate

4. Pre-master key (Encrypted with server’s public key)

5. Decrypt Key info using Private Key

6. Calculate symmetric key

7. Finished Message (Encrypted with RSA sym key)

8. Finished Message (Encrypted with sym key)
**Public Key Infrastructure Operations**

**Cipher Suites**

```plaintext
Cipher Suites (26 suites)

- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
- Cipher Suite: TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x009f)
- Cipher Suite: TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x009e)
- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xc024)
- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 (0xc023)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xc027)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc00a)
- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
```