Device Security Requirements

Version 1.0

REVISION HISTORY

Version	Date	Authors	Description
0.1		Emilia Quijano (Verizon)	Initial Requirements from Verizon
0.2		Brendan Black (Vodafone)	Merge Requirements from Vodafone
0.3		Emilia Quijano (Verizon)	Merge Comments & Requirements
0.4		Tomer Semo (SAM Seamless Network)	Restructure
0.5		Tomer Semo (SAM Seamless Network)	Merge Comments from members such as Allot, Avast, Broadcom, BT, Deutsche Telekom, Intel, MaxLinear, McAfee, SAM Seamless Network, Verizon, and Vodafone.
1.0	16/11/2020	Brendan Black (Vodafone), Dave Barr (MaxLinear)	Reformat for Initial Release (BB), Edited for clarity (DB)

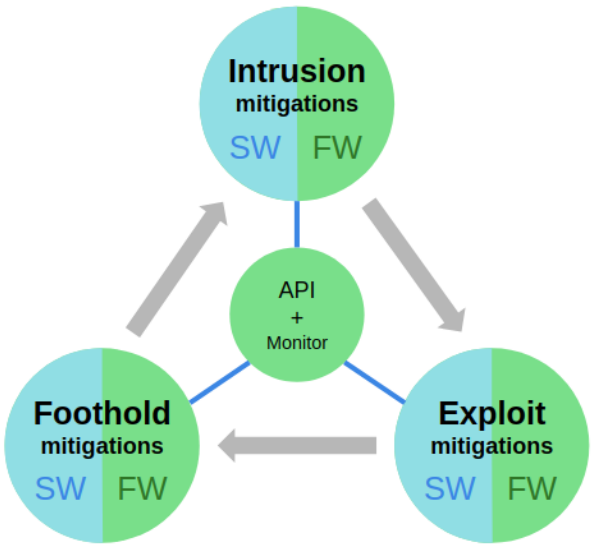
CONTENT

ABOUT THIS DOCUMENT

1. INTRODUCTION

This document describes requirements to assure security of CPE Devices. The requirements span both hardware and software components, and their suppliers, such as SoC silicon vendors, BSPs, OEM system integrators, middleware providers, router operating systems, and high-level application developers.

- The categorization model that was used to raise and categorize the security requirements for a secured, future-proof firmware standard was the following:



  - From an attacker point of view, there are three main parts of a cyber-attack lifecycle: **Intrusion**, **Exploitation**, and **Foothold Maintenance**.

- Our goal is to make those three aspects too-hard-to-achieve by an attacker (meaning - to take the risk level of threats in each aspect to a manageable level), through dedicated mitigations via Firmware and Software.
- Beyond mitigations, a crucial part of future-proofing the solution regarding all three aspects would be:

    - Exposing an internal API for future security apps to be composed on top of the stack.

    - Provide a strong monitoring infrastructure, integrated at firmware level, which monitors attacker-created footprints and anomalies **regarding the three aspects mentioned above**. The data could be subscribed to by internal apps and solutions running on the firmware via API or sent to an external collector for cross-device aggregated analysis.

    - Following this model through this paper will allow us to validate the effectiveness of the entire set of requirements for software stacks such as prplWRT.

2. INTRUSION MITIGATIONS

* This chapter covers all the mitigations concerning the effort to reduce the attack surface on the device. Obviously, the focus here will be the admin interfaces the device has to offer.

2.1. PATCH MANAGEMENT AND UPDATE POLICY

- The device shall be launched with the most recent versions of all its components regarding security patches and updates - OS, services and daemons, libraries software components.
- The device should be maintained with frequent updates to continue and run the most recent version of said components.
- Any known vulnerabilities of the components shall be fixed before device launching. All software / firmware used by the device must be checked, patched [and verified] against known vulnerabilities.
- The device should be delivered with a secure and failsafe update mechanism.
- The device should communicate only via secure protocols (e.g., HTTPS over HTTP, SFTP over FTP, SSH over Telnet), using the standard cryptographic abilities listed below.
- The transports protocols should be configured to not support protocol-downgrades to any unsecure version - both at OS and application levels.

- Communications between devices and external management services shall be properly secured using TLS 1.2 or higher. The connection shall be validated mutually - each device should have its own unique certificate signed by a trusted CA.

- If the device sends/receives any data to/from other devices in the network, the data shall be encrypted as specified in the secure transfer protocol being used (e.g., ZigBee, Z-Wave, Bluetooth, Wi-Fi, NFC, LTE, 5G), and data integrity shall also be checked.
    - More on protocol specifications in Appendix A - Wireless Communications Specifications.

2.2. CRYPTOGRAPHIC CAPABILITIES

2.2.1. TRANSPORT MECHANISMS

- The device should communicate only via secure protocols (e.g., HTTPS over HTTP, SFTP over FTP, SSH over Telnet), using the standard cryptographic abilities listed below.

- The transports protocols should be configured to NOT support protocol-downgrades to any unsecure version - both at OS and application levels.

- Communications between devices and external management services shall be properly secured using TLS 1.2[2.] or higher. The connection shall be validated mutually - each device should have its own unique certificate signed by a trusted CA.

- If the device sends/receives any data to/from other devices in the network, the data shall be encrypted as specified in the secure transfer protocol being used (e.g., ZigBee, Z-Wave, Bluetooth, Wi-Fi, NFC, LTE, 5G), and data integrity shall also be checked.
    - More on protocols specifications in Appendix A - Wireless Communications Specifications.

2.2.2. CERTIFICATE MANAGEMENT

- The device shall support Certificate Enrollment using Simple Certificate Enrollment (SCEP) Protocol or equivalent like Certificate Management Protocol (CMP), or other common industry standard. The use of SCEP or equivalent allows any standard network device to simply request a digital certificate electronically.

- The device shall have the mechanism to renew and revoke certificates (e.g., using CRL or OCSP certificate checking methods).

2.2.3. DATA ENCRYPTION

- The device shall support required cryptographic features like secure transports and certificate validation with a hardware-backed security framework including, but not limited to, key generation and management, random number generation, encryption/decryption, hashing and secure storage.

- Proprietary encryption algorithms must not be used. Whenever possible, publicly available libraries should be used instead.

- The device shall encrypt any provisional, operational or user data stored on the device using a unique random encryption key locally generated on the device. The encryption key shall be security stored on the highest trust domain available, preferably a hardware-backed secure storage.

2.3. SECURE FIRMWARE/SOFTWARE UPGRADE

- During the firmware/software upgrade process, the device shall verify that all firmware/software upgrade images have been signed by an authorized party providing a root of trust stored securely on the device. This requirement is applicable to all firmware/software upgrade methods including but not limited to: OTA, PC tools, file loading from USB or SD, file transfer from other devices or network/cloud.

- If the verification of a firmware/software image fails, the device shall abort the upgrade process and indicate it by any available method, e.g., displaying an error message on screen, turning a red light on, blinking lights, etc., and shall report it to other devices or network/cloud which manage the device.

- If firmware/software upgrade is supported OTA or over the internet, the device shall use a secure communication channel (e.g., HTTPS) to an upgrade server, and certificate pinning shall be supported, and the certificate shall be hard coded on the device.

- Before updating any firmware/software, the Device shall validate the firmware signature and firmware version to ensure it's a newer version than the current build, and shall NOT allow any downgrade to an older version.

- Firmware images shall not contain any secrets, sensitive files, or shared (symmetric) cryptographic material.

2.4. SECURE DEVICE MANAGEMENT

- A managed device shall be associated with a managing device (e.g., server) or network/cloud through a registration and authentication process, either on the managing device/network/cloud or on both sides. The device shall NOT support automatic association (pairing) with managing device/network/cloud.

- Consider adding additional protections to Device Management Servers that would require devices to authenticate with the server.

2.5. ADMINISTRATIVE INTERFACES SECURITY

2.5.1. VISIBILITY AND CONTROL

- All administrative interfaces the device exposes on the network (LAN/WAN) or consumes from the WAN, including but not limited to WEB, SNMP, SSH, TR69, etc. shall be managed (ability to turn on and off in LAN and in WAN separately) via a dedicated web interface and API.

- Unused network services and ports shall be disabled. The device shall load only necessary components and shall run minimum-necessary services based upon its configuration.
- All services providing additional functionality to the Device (e.g., Samba, printer support, WebDAV for USB access, DNLA, etc.) should be disabled by default. When enabled, these services should use the encrypted versions of the protocols and strong passwords for authentication whenever possible. However, it should be possible for users to manually activate individual services through the Management Web-GUI or via remote management methods.

2.5.2. DEBUG BLOCKING

- Devices shall disable all hardware debugging ports, e.g., USB, JTAG, UART at production.
- Devices shall disable all debugging commands or hidden menus in admin or user interfaces.

2.5.3. APPLICATION SECURITY

- All Administrative interfaces, including but not limited to a Web-GUI, a dedicated configuration Shell interface, REST or RPC service, shall be frontend unprivileged services - responsible only for authentication and exposing the device APIs to the customer.

- Administrative interfaces shall perform privileged functions in the device only via external services that expose dedicated, privileged APIs.
    - The APIs shall be used as B2B services between applications to the OS.
    - The APIs shall enforce an authorization model sufficient to limit access to resources.
    - Only minimum-necessary API functionality shall be exposed to applications and to the user.

- Administrative interfaces and APIs exposed by the Device shall be developed according to OWASP's guidelines to prevent logical exploits on said interfaces.

    - All of the coding practices in each version of the OWASP Top-10[1.] shall be implemented. Note: Even items dropped from the Top-10 still represent vulnerabilities.
    - Services and apps shall be developed with mitigations to all application-based attacks explained at OWASP[3.], including but not limited to:
        - Command Injections
        - Path-Traversals
        - Cross-Site Request Forgery[10.]
        - Session Fixation
        - Informative Errors
    - Low level APIs shall include mitigations against all types of Memory Overflows.
    - Input sanitization with a white-list of characters shall be implemented system-wide and performed on any user / external input to a service or API, rejecting any input that does not match the white-list.

- Access to administration and management API functionality shall be limited, assigned at the functional level, and shall not cascade down. Principles of minimum-necessary privilege must always be enforced for all administration and management functions.

- Network facing interfaces shall not expose detail or sensitive or technical information through headers or banners.

2.5.4. AUTHENTICATION

- Any admin access to the device (No matter the interface - Web, SSH, REST APIs, etc.) shall be secured with a strong authentication scheme.

- Authentication processes in applications shall use session-based authentication and authorization. Avoid the use of static identification identifiers.
    - Tokens created while authenticating customers in applications shall be passed through administrative APIs mentioned above for authorization.

- Basic Authentication (RFC 7617) /Digest Authentication must not be used.

- Passwords shall be stored only as hashes. This applies to both operating system users and applications that implement proprietary authentication.
    - The hashes shall be stored with a standard salting mechanism.
    - Only strong hashing algorithms shall be used.

- All admin Interfaces shall avoid storing hardcoded credentials (e.g. within a binary) used to authenticate special users.

- All authentication-requiring services should support 2FA authentication.

- Admin users are able to change the admin password, which shall revert back to default following factory reset.

- Password-change functionality has the following requirements:
    - Users must input their old password before updating the account with a new password.

    - Application that do not use SSO, must send an out-of-band message (e.g., email/postal service, text, phone, etc.) to the account owner on record informing them that their password has been changed.
    - Password reset, or forgotten password, functionality has the following requirements:
        - Passwords must be obfuscated. If a challenge response is used, the challenge answer must be requested via a secure protocol (e.g., HTTPS), encrypted while stored, and not be easily guessed or deciphered.
        - If an offline password recovery mechanism is used, use a Time-based One-Time Password [TOTP] or some other soft token.
        - For such purposes, e-mail is deemed an insecure protocol; hence, no sensitive auth data, such as a password may be exchanged. For e-mail driven resets (considered a weak, last resort) the following procedures are required:
            - The application will send a one-time validation token to the e-mail address on record for that account that does not compromise the account. The token must not be the only method of authenticating the user for a reset
            - The application will then redirect the user to a secure password reset link, which they will then select and enter the password reset flow
            - The application will send a message to the registered user explaining that their password has been reset

- When using certificate-based authentication:
    - A unique certificate shall be provisioned per device and signed with a trusted CA.
    - Whitelisted, valid certificates shall be enforced.

- Secrets, API keys, and passwords must not be included in the source code, or online source code repositories.

2.5.5. SECURITY TESTING

- Vulnerability and Penetration testing shall be completed on ALL (WAN, LAN, and Management) network-facing interfaces.

2.6. VULNERABILITY MANAGEMENT

- A Vulnerability Management Program aimed to protect, detect, and promptly respond to and recover from vulnerabilities and threats of devices shall be established and maintained, including continuous enhancement of the program through effective measurements and reporting of metrics

- An annual review of existing Vulnerability management practices shall be reviewed to ensure it aligns with common industry-wide Vulnerability Management standards and updated when major changes have occurred.

- A Vulnerability Management team with defined roles and responsibilities shall be identified and established.

- Vulnerability metrics and reporting shall be developed, documented and reviewed periodically to identify patterns and help mature the program.

2.7. MONITORING REQUIREMENTS

- If the device supports admin access via web or other connection, a countermeasure against brute-force attack shall be provided. For example,

locking the device after x number of unsuccessful login attempts (which can be unlocked by a managing device or network/cloud), introducing time delays between unsuccessful login attempts.

- APIs shall log security and other important events as useful information in DOS detection and better profiling/characterization of the system (e.g., number of API invocations/sec, response time, etc.).
  The following events shall be logged:
  - All Authentication events.
  - Special Logins (admins).
  - Failed Logins.
  - All APIs: their actions and the user requesting the action.
    - Remote commands and their source (user + IP).
  - Firmware updates.
  - Failed validation of an update.
  - Communication channel setup with server, and communication errors.

3. EXPLOIT MITIGATIONS

This chapter covers mitigations against attackers with an execution vector to execute code remotely. The aim is to handle both logical and memory corruption exploits alike.

3.1. MEMORY CORRUPTION MITIGATIONS

- The device shall include the following mechanisms, running OOTB to mitigate memory corruption attacks during exploitation attempts:
  - DEP
  - ASLR + KASLR
  - CFI + KCFI
  - Stack Canaries
  - LKRG

3.2. RUNTIME INTEGRITY

- The device shall support runtime system integrity check which detects unauthorized firmware/software modification, device rooting or unknown applications/processes.
  - If the runtime system integrity check failed, the device shall indicate it by any available method, e.g., displaying an error message on screen, turning a red light on, blinking lights, etc., and shall report it to other devices or network/cloud which manage the device.
  - The indication and recovery methods shall be described in a user guide of the device.

- The device should embed in its operating system's loader a certificate-validation functionality and only allow to load an ELF or similar file from an authorized vendor.

- The device should include an application whitelisting functionality and prevent loading files (based on hashes) not appearing in a pre-defined configuration file.

- The device should not expose any ptrace() system calls.

3.3. SOFTWARE COMPONENTS

- All OEM provided applications and services shall run with minimum-necessary privileges (i.e., least-privileges model). Using the privileged API model mentioned before, means that all outward-facing applications shall run as weak users.

- Services exercising strong APIs shall perform delegation of requesting users.

- If 3rd -party applications are allowed to run on the device, the device shall support application security features including, but not limited to, application sandbox, permission and privilege control, app signing signature checking and malware detection.

3.4. MICROSERVICES/CONTAINERS

- Applications on the device including, but not limited to, administrative interfaces, should run as containers / microservices for isolation.

- Micro-services shall not have "ulimits" values higher than their parent.

- Every microservice shall have a unique UID/GID mapping range and shall not share it with other microservices.
- Resource exchange between micro-services shall be provided through shared resources.
- The cgroups limits for each micro-service shall be provided as lxc.cgroup configuration entries (CPU, RAM, PIDs ranges).
- Configuration of each microservice shall be available through the means of service providers' device-management communications protocols.
- Micro-services vendors shall provide security and functionality test plans for testing each of their microservice solutions.
- Every micro-service solution shall be tested and certified through a defined certification process.
- Micro-services shall be deployed securely by checking their corresponding signatures both in a microservices store and on the device where they are being deployed.
- Micro-services supporting a network bridge-connection shall limit user access to bridges allocated in /etc/lxc/lxc-usernet.
- Updates of micro-services shall be transaction-based, in which updates successfully complete or not at all.
- Updates of micro-services shall not impact the host operating system.
- Delivery of updates shall be secured using TLS[2.].

3.5. MONITORING REQUIREMENTS

- The following events shall be logged:
    - All files created on the device since boot.
    - All loaded ELF and .so files (path and MD5).
    - Failed loads at validation, including MD5 and paths.
    - PID and PPID for all running processes and command lines.
    - Crashed processes.
    - Boot Events.
    - Container events.

- Core Dumps shall be saved and reported.

4. FOOTHOLD MAINTENANCE MITIGATIONS

This chapter covers mitigations preventing an attacker who has executed malicious code on the device from maintaining a foothold on it - from persistency to command and control. This chapter's focal point would be the monitoring part, for this is the best way to tip the attacker off balance.

4.1. SECURE BOOT

- During the boot process, the device shall verify that all firmware/software images including bootloader(s) and system images on the device have been signed by the OEM, based on a root of trust securely stored on the device.
- If the verification of a firmware/software image fails, the device shall abort the boot process and indicate it by any available method, e.g., displaying an error message on screen, turning a red light on, blinking lights, etc., and shall report it to other devices or network/cloud which manage the device.
    - The indication and recovery methods shall be described in a user guide of the device.

4.2. READ ONLY FILE SYSTEM

- The File system shall be defined as READ ONLY except the RAM (RWX) and the config zones (RW).

- The fact that Config files are defined RW stresses the need to evaluate with extensive penetration tests the way administrative services handle the information written in those files and treat it as untrusted.

4.3. BINARY DECEPTION

- The device should replace tampered versions in its default /bin directory with binaries which do not do anything but log their use. Of course, the log of such events should be classified as critical.

4.4. MONITORING REQUIREMENTS

- The following events shall be logged (respecting any anonymization restrictions):
  - All running processes
  - All open sockets from the router
  - All loaded SOs
  - All loaded kernel modules
  - All files created, deleted and modified since boot and corresponding process

5. CONTACTS

The following table describes the primary contacts to ask about the prpl Security Device Specification

Contact at prpl	Email address
Info	info@prplfoundation.org

6. APPENDIX A: WIRELESS COMMUNICATIONS SPECIFICATIONS

The sections in this appendix specify requirements that apply only when the corresponding wireless interface is supported by the Device.

6.1. WI-FI

- The device shall be certified by the Wi-Fi Alliance.
- The device shall set WPA2 as the default encryption method.
- The device shall support 802.11i standards.
- The Wi-Fi password shall be changed from default and shall conform with an approved password standard/policy.
- If the device supports Wi-Fi Protected Setup (WPS) then it must support the use of the Push Button Configuration (PBC) as a way of adding a new device to the network.
- Support of the WPS PIN method is optional; however, if it is used then:
    - Devices must be configured to support the Internal Registrar (IR) after activation in the WebUI by entering of the client's device PIN into a form field of the device's protected WebUI for just one time.
    - Devices must be configured to support External Registrar (ER) with a block after 10 successive failed login attempts (until factory re-set applied); the counter for the failed authentication attempts must be independent from the elapsed timeframe and the used devices.
    - Default WPS PIN must be generated sufficiently random (e.g., by using a pseudo-random number generation) and must be unique for each device.

- Warnings must be displayed to the user on the WebUI as long as a default PIN is in use.
- A strong WPS PIN policy must be used for user-customized PINs. If more than one device is detected during the pairing period, connection attempts must be aborted, and a "session overlap" error shall be reported and logged.

6.2. Bluetooth Low Energy (BLE) Security

- Device shall support BLE version 4.2 or higher.
- Device shall support LE Secure Connections Pairing which utilizes P-256 elliptic curve.
- Each device shall generate its own Elliptic Curve Diffie-Hellman (ECDH) public-private key pair, i.e., private (secret) key and public key.
- The public keys shall be exchanged between the connecting devices and shall be initiated by the initiating device by sending its public key to the receiving device and the receiving device replies with its own public key.
- During Pairing, the following Pairing features shall be exchanged:
  - Authentication requirements: MITM protection, Bonding
  - I/O capabilities – Display Only
  - Out-of-band flag – OOB Authentication data not present
  - Pairing Method – Passkey Entry
- For Passkey Entry, the user shall input an identical Passkey (6-digit number) into both devices. Alternately, the passkey may be generated and displayed on one device and the user then inputs into the other. This short-shared key will be the basis of the mutual authentication of the devices.

- Bonding and Man-In-The-Middle (MITM) protections shall be enabled.

6.3. Z-Wave

- The 4th generation or later specification of Z-Wave SHALL be used.
- Security 2 (S2) SHALL be implemented.
- If any personally identifiable information (PII) is being passed between devices, then encryption SHALL be used.

6.4. ZigBee

- Both the Standard Security Mode and High Security Mode SHALL be implemented.
- Key provisioning shall use the Key Transport method. If ZigBee Pro is being used, then SKKE should be implemented in addition.
- If any personally identifiable information (PII) is being passed between devices, then encryption shall be used.

6.5. Bluetooth Classic

- Bluetooth 2.1 or higher shall be implemented.
- The device shall not remain in discoverable mode once pairing has been completed.
- Pairing of the device shall use the Secure Simple Pairing (SSP) protocol.
- Only those services that are essential to the Bluetooth communications shall be listed via the Service Discovery Protocol (SDP).
- If any personally identifiable information (PII) is being passed between devices, then encryption of the communications shall be used.
- Bluetooth anonymity mode should be implemented.

7. APPENDIX B: LOGGING AND REPORTING SPECIFICATIONS

- The device shall be capable to upload/report event logs to other devices and/or a network/cloud which manage the device using a secure communication channel.
- Device shall have enough processing capacity for nominal logging levels, and headroom capacity for excess logging spikes e.g., system errors, unusual events, etc.
- Device shall have enough storage to save said logs for a substantially significant amount of time.
- Device logs shall be restricted to authorized users only via ACL.

- Log storage space needs to be monitored against overflow.

8. APPENDIX C: USER SERVICES

- Any export functionality for configuration files must ensure that if sensitive data is included in the export file it is protected by strong industry standard encryption providing both confidentiality and integrity, and with a sufficiently strong user selected password.

- Devices must delete any stored personally identifiable/private information (e.g., usernames, passwords, computer names, hardware addresses, IMEI, IMSI, Wi-Fi settings, custom settings, log files etc...) when performing a factory reset, or provide alternate functionality to delete such information.

9. APPENDIX D: OMITTED BULLETS (AND REASONING) (INFORMATIVE)

- Source code and development (e.g., testing, API) documents should not be publicly disclosed.
  The prpl Foundation fosters a robust community to innovate CPE devices by harmonizing interfaces in open APIs and delivering open-source reference implementations of common infrastructure.

- Consider using encrypted firmware builds to slow attackers from reverse engineering the device's firmware.
  Same reasoning as above.

- Generation of a random filenames when uploading files should not be permitted.
  Random file names at upload are actually good, but should not be user controlled.

- Code shall be audited to ensure that no file paths can be formed by concatenating user input.
  Other requirements were made more general to comply with OWASP.

- Access to administrative and management functions via the WebGUI shall be limited to fulfill administrative functions (i.e., use of system commands, change device setting and configuration).
  Didn't get that? Suggest that Admin login only needed if administration functions are to be used.

- A new random Session ID shall be generated after each login.
  Other requirements were made more general to comply with OWASP.

- Session IDs shall not appear in the WebGUI URL, but securely stored and invalidated after logout, idle and time-outs.
  Other requirements were made more general to comply with OWASP.

- WebGUI shall return error messages containing no debug information.
  Other requirements were made more general to comply with OWASP.

- REST services must be protected from Cross-Site Request Forgery via the use of at least one or more of the following: ORIGIN checks, CSRF nonces, and/or referrer checks.
  Other requirements were made more general to comply with OWASP.

- REST services must explicitly check the incoming Content-Type header to be the expected one, such as application/xml.
  Other requirements were made more general to comply with OWASP.

- API functionality that could add a backdoor and capable of shell execution shall be removed.
  Other requirements were made more general to comply with OWASP.

- Forms requiring credentials must require re-entering credentials for every authentication.
  Goes against session based auth? Check credentials in OWASP
  https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html#require-re-authentication-for-sensitive-features and
  https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html#session-expiration

- Filtering and blocking of MAC addresses must be supported on LAN and Wi-Fi interfaces.
  This mitigation will be addressed in other prpl documents, such as the forthcoming Security API.

- Data encryption: Items like data discard and data in various locations on the device are not handled. Such as keys and user data in RAM, keys and data in persistent storage, key immutability on public keys, Copy and use of the keys, etc. What is prpl trying say here? Keys should not be kept in plaintext or unencrypted in ram or on temporary storage for devices without TPM/Trusted execution environment or equivalent, hardware cryptographic acceleration or protected storage. Forcing this requirement could put a lock on hardware designs that have hardware crypto and or TPM/equivalent, so care is needed.

10. REFERENCES AND FURTHER READING

[1.] OWASP Top-10 : https://owasp.org/www-project-top-ten/

[2.] TLS 1.2 RFC 5246: https://tools.ietf.org/html/rfc5246

[3.] OWASP Attacks: https://owasp.org/www-community/attacks/

[4.] BBF Technical Reports: https://www.broadband-forum.org/technical-reports

[5.] SCEP - Simple Certificate Enrolment Protocol: https://tools.ietf.org/html/rfc8894

[6.] OCSP – Online Certificate Status Protocol: https://tools.ietf.org/html/rfc6960

[7.] CMP – Certificate Management Protocol: https://tools.ietf.org/html/rfc4210

[8.] OWASP Authentication Cheat Sheet:
https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

[9.] OWASP Session Management Cheat Sheet:
https://cheatsheetseries.owasp.org/cheatsheets/Session_Management_Cheat_Sheet.html

[10.] OWASP Cross Site Request Forgery https://owasp.org/www-community/attacks/csrf

[11.] OWASP Cryptographic Storage Cheat Sheet:
https://cheatsheetseries.owasp.org/cheatsheets/Cryptographic_Storage_Cheat_Sheet.html