Security Guidelines for Works with SmartThings (WWST) Devices

March 2025

Introduction

Devices connecting to IoT services are diverse, ranging from small sensors to security cameras to smart appliances. As a consequence, security is a major concern in the deployment and management of these devices. This document aims to establish the security guidelines for the IoT devices connecting to the SmartThings cloud with the goal to protect the cloud services and the end users.

These guidelines form the basis for a secure connection to the SmartThings cloud. Vendors must implement the necessary security functions on their platforms that connects to the SmartThings cloud.

We will first discuss threats and attack vectors in general. Next, we will describe the security guidelines for the IoT devices to mitigate these threats and attacks. In these guidelines, we use the most common words such as “must” and “should” as defined by RFC 2119.

DISCLAIMER:

SAMSUNG OR SMARTTHINGS DOES NOT REPRESENT OR WARRANT THAT FOLLOWING THESE GUIDELINES WILL ENSURE THE SECURITY OF ANY DEVICES OR SERVICES. SAMSUNG OR SMARTTHINGS DOES NOT HAVE ANY LIABILITY RELATED TO OR ARISING OUT OF ANY SECURITY BREACH ON ANY OF VENDOR’S DEVICES OR SERVICES. THESE GUIDELINES ARE INTENDED TO BE USED AS A REFERENCE ONLY. VENDORS ARE SOLELY RESPONSIBLE FOR THE USER PROTECTION AND THE SECURITY OF THEIR DEVICES AND SERVICES.

THESE GUIDELINES MAY BE CHANGED TO IMPROVE THE SECURITY FOR THE CLOUD AND SERVICES WITHOUT ANY NOTICE. THE UPDATED GUIDELINES ARE NOT APPLICABLE TO THOSE DEVICES THAT ARE ALREADY CERTIFIED BY WWST CERTIFICATION PROGRAM BASED ON A PREVIOUS VERSION OF THE GUIDELINES.

THERE MAY BE ADDITIONAL GUIDELINES OR TEST CASES REQUIRED BY THIRD PARTY EVALUATION LABS FOR CERTIFICATION PURPOSES, IN ADDITION TO THE REQUIREMENTS DESCRIBED IN THIS DOCUMENT.

Assumptions & Threats

Assumptions: We assume that any software running on a device can be compromised. We also assume that attackers can gain physical access to a device, and that a device is operating in an open environment without proper access control. We assume that attackers may have full access to the network between a device and the cloud, and that attackers may eavesdrop on any communications between the device and the cloud.

Threats: We explore the classes of threats in general. More specific threats can be developed depending on the service and business scenarios.

Threat 1 – Device control hijack:

Attackers may try to hijack the control of a device in order to manipulate it to their benefits. Devices for the home security, or the ones equipped with a camera such as a connected door lock, a security camera, or a robot vacuum cleaner can be an attractive target. Such devices can be exploited to break-in or spy on the inside of others' house.

Attackers can also use compromised devices to attack other services (for example, DDoS attacks) or to do something else (e.g., bitcoin mining) by controlling them as a botnet. When a device connected to the SmartThings cloud is compromised, it can also attack other devices registered to the same user and abuse cloud resources for nefarious reasons (e.g., storing illegal content in the user's cloud storage), amplifying the impact of the attack.

Threat 2 – Information disclosure:

The data associated with a device may be valuable to some classes of attackers. By breaching either the service or the local IoT network, it may be possible for an attacker to gain access to the user's private information (e.g., the user's physical location or billing information for the service, or perhaps the user's home network by pivoting through the IoT hub).

Threat 3 – Service disruption:

Attackers may try to attack the service end-points in the cloud to steal the credentials or to disrupt the service itself. Well-resourced attackers can launch DoS attacks against the infrastructure to cause severe outage.

Security Guidelines for WWST Devices

Security Certification Levels:

We define *three* levels of security requirements to cover a wide spectrum of data sensitivity (e.g. indoor video camera, presence sensors, level of dimness etc.) and breach impact (e.g. unable to view live video feed, getting locked out of home, fake motion events etc.) these devices can make. A higher security level adds more requirements over those needed to meet the lower levels.

Level	L1 (Level 1)	L2 (Level 2)	L3 (Level 3)
Description	A level for minimum security Appropriate for devices that operate in low impact environments and do not involve storage, processing or transmission of any sensitive, PII[1], or regulated data.	Enhanced level of security Appropriate for devices that do not operate in security focused functions, health care, or other regulated environments but may deal with moderately sensitive information.	High level of security Appropriate for devices that deal with highly sensitive information; or operate in security focused functions, health care, or other regulated environments. For regulated domains, each device must be compliant with regulations of the specific domain. The requirements described in this documentation may be superseded by domain-specific regulation.

[1] Personally identifiable information

Device Security Guidelines:

To protect the SmartThings cloud services and their end users under the general threats model, all IoT devices directly connecting to the SmartThings cloud must be installed with device credentials (e.g. an X.509 certificate issued by Samsung IoT Device CA or third-party CA accredited by Samsung or raw key pair). Mutual authentication between the IoT device and the SmartThings cloud is based on these device-unique credentials. The storage mechanism of those credentials on the device and other mechanisms (for example, secure boot/update, device integrity protection, etc.) needed to mitigate various threats to the device and service must be implemented to meet the requirements, followed by the security levels in parentheses, described in this document.

1. Hardware Root-of-Trust
   - 1-1 (Secure Boot): The first bootloader to get executed by the processor on power-on must be implemented on ROM and the provision of the secure boot parameters must be implemented in hardware during the manufacturing process. (L2, L3)

2. Trust Chain

   Trust chain can be accomplished through either X.509 certificates or raw key pairs.

   A. X.509 Certificate

   If device will be using X.509 certificates as device credentials:

   - 2-1A (Trust CA – L3): Each device must be equipped with an X.509 certificate issued by Samsung IoT Device CA or third-party CA accredited by Samsung. (L3)

     - ■ 2-1A-1: The third-party CA must submit an audit result before its CA certificate is initially registered to the cloud. (L3)

   - 2-2A (Trust CA – L1/L2): Each device must be equipped with an X.509 certificate issued by the third-party CA that meets the following requirements. (L1, L2)

     - ■ 2-2A-1: The third-party CA must issue certificates for end-entities only, and must not issue certificates for CAs. (L1, L2)
     - ■ 2-2A-2 (CA Operation): The operation of the third-party CA must be conducted in a secure manner to be compliant with the industry standard. (L1, L2)
     - ■ 2-2A-3 (CA Operation): If device key pair is generated outside device and later injected, the generated private key must only be maintained outside the device in encrypted form and should be deleted after successful injection. (L1, L2)
     - ■ 2-2A-4 (Certificate Issuance): The device certificates generated by third-party CA must be globally uniquely identified independently based on its subject names and certificate serial number. (L1, L2)

- 2-2A-5 (Certificate Issuance): The certificate subject names issued by the third-party CA must be unique. Device certificates must include a unique device identifier in the CN field as part of the subject name, and must also have unique <certificate serial number, issuer name> pairs. (L1, L2)
- 2-2A-6 (Revocation List): The third-party CA must provide a way to check the validity of each certificate for known compromised devices either through a certificate revocation list (CRL) or OCSP. (L1, L2)

B. Raw key pair

If device will be using raw key pair as device credentials:

- 2-1B (Trust Key Management – L3): Each device must be equipped with raw key pair (public/private) issued by third-party key management facility accredited by Samsung. (L3)

  - 2-1B-1: The third-party key management facility must submit an audit result before first device from this vendor will be connected to the Samsung cloud. (L3)

- 2-2B (Trust Key Management – L1/L2): Each device must be equipped with raw key pair (public/private) issued by the third-party key management facility that meets the following requirements. (L1, L2)

  - 2-2B-1 (Key Management Operation): The operation of the third-party key management facility must be conducted in a secure manner. (L1, L2)
  - 2-2B-2 (Key Management Operation): If device key pair is generated outside device and later injected, the generated private key must only be maintained outside the device in encrypted form and should be deleted after successful injection. (L1, L2)
  - 2-2B-3 (Key Issuance): The third-party generated key pairs must be globally uniquely identified independently based on its identifiers. (L1, L2)
  - 2-2B-4 (Key Issuance): Key blob must include a unique device identifier in the description field, include unique <key serial number, key issuer name> pairs, and signed using the manufacturer’s certificate. (L1, L2)
  - 2-2B-5 (Revoked Key List): The third-party key management facility must provide a way to check the validity of each key pair for known compromised devices. (L1, L2)

3. Device Integrity Protection and Detection

- 3-1 (Secure Boot): Each device must provide secure boot to validate the integrity of the security critical executables and stop the boot process if any integrity of security critical executables is compromised. The hardware Root-of-Trust must be based for validation of the integrity. (L2, L3)
- 3-2 (Integrity Validation – L1): Each device must provide the integrity validation mechanism to validate the integrity of the security critical executables. The key used for validating the integrity must be pinned at the least to the validating software module. (L1)
- 3-3 (Secure Update): Each device must be able to update its firmware; and the hardware Root-of-Trust must be based for validation of the integrity and authenticity of signed updates. (L2, L3)
- 3-4 (Secure Update – L1): Each device must be able to update its firmware; and the key used to validate the integrity and authenticity of signed updates must be pinned at the least to the validating software module. (L1)
- 3-5 (Roll-back prevention): Device should provide security controls to prevent firmware rollback
- 3-6 (Known vulnerability protection) Device should not be running old versions of libraries and system software with known vulnerabilities that could likely compromise the device and its ecosystem.
- 3-7 (Failsafe measures): Device should implement failsafe measures to ensure its integrity is not compromised during an error state.
- 3-8 (Input Sanitization): Device should perform all input and output data validation/sanitization to prevent common injection attacks, including command injection, SQL injection, XSS etc.
- 3-9 (OS Hardening): Device OS should be hardened to minimize the attack surface and ensure the principle of least privilege. All interactive OS accounts are either removed or disabled, proper access control is implemented, nonessential services/interfaces are removed, applications are run at lowest privilege possible (not at root level), etc.

4. Protected Communication
   - 4-1 (Protected Communication): All network communications by the device must be protected to provide confidentiality and integrity. (L1, L2, L3)

5. Protected Storage
   - 5-1 (Secure Storage): Each device must provide secure storage to protect the confidentiality and integrity of data from any unauthorized access. Hard coded credentials should not be used in device software. (L1, L2, L3)
   - 5-2 (HW Protection): Secure storage of sensitive data must be hardware backed and all cryptographic keys must be stored encrypted in hardware backed secure storage (e.g. eSE, TEE). (L3)
   - 5-3 (SW Protection): Secure storage of sensitive data must be protected at the least in software; and all cryptographic keys (except the keys used to protect the secure storage) must be stored encrypted in a secure storage. Keys used to protect the storage must be protected by other technical mechanisms (e.g. access control). (L1, L2)
   - 5-4 (Encryption Algorithms and Libraries): Device should use current industry standard algorithms/modes and widely accepted implementations. E.g., AES-128/AES-256 CBC/GCM/XTS-

AES modes using BouncyCastle or OpenSSL libraries at the time of this writing. For the most current security recommendations refer to NIST 800-131A or OWASP.

- 5-5 (TEE): Sensitive software components such as cryptographic processes should be isolated by implementing a Trusted Execution Environment, or assigned to a higher privilege than other software components.
- 5-6 (Biometric Data Protection): If the device handles biometric data (for example, facial images, fingerprints, iris scans, and/or other biometry), it must process the data on the device and send only a result that does not contain biometric PII to SmartThings Cloud, and store them in an encrypted form. (L3)

6. Authentication/Authorization
   - 6-1 (Device Identity): Each device must be identified uniquely. (L1, L2, L3)
   - 6-2 (Mutual Authentication): Each device must provide device credentials-based mutual authentication with SmartThings cloud. (L1, L2, L3)
   - 6-3 (Default Credential): Any default credentials that device ships with either must be disabled on successful provisioning/transition to normal working state, or default credentials must be unique to each device meeting minimum strength depending on the credentials. If passwords unique to each device are pre-installed, these should be generated, with respect to their length and complexity, in a way to reduce the risk of automated/brute-force attacks. (L1, L2, L3)
   - 6-4 (Development credentials): Production version must not use the same keys, passwords etc. as used for test/dev version. (L1, L2, L3)

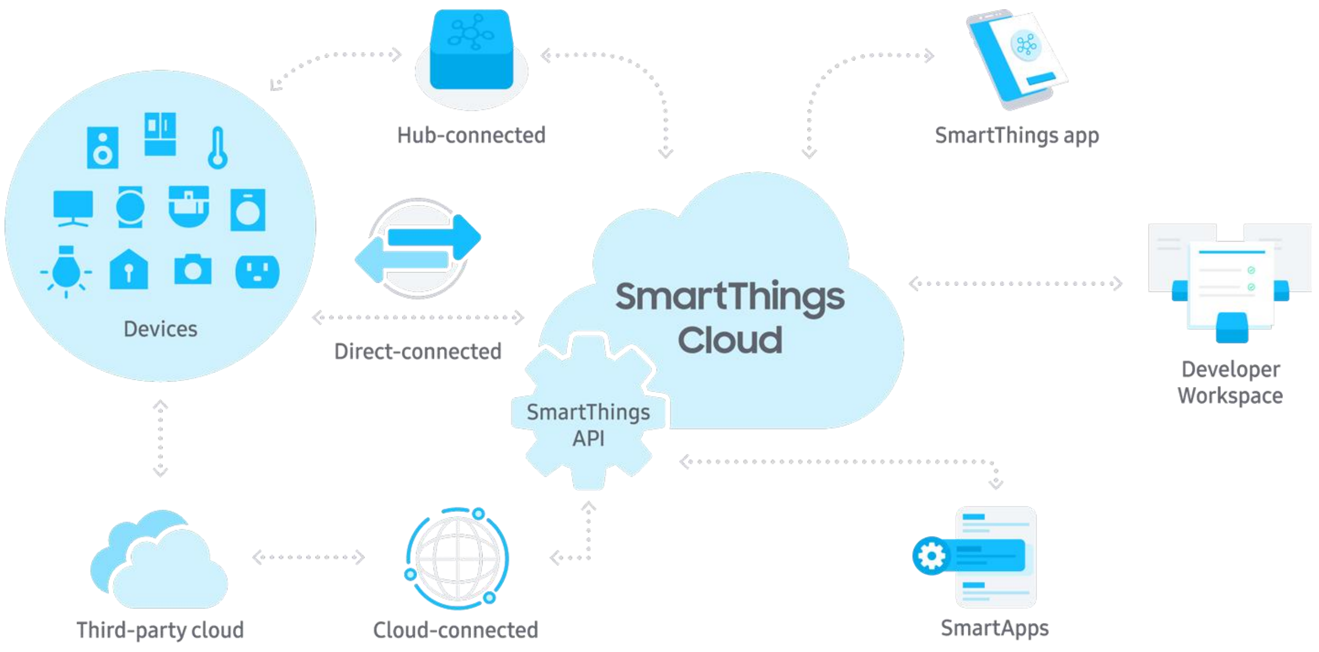
7. Hardware Security
   - 7-1 (Protected Local Ports): Device's local ports (for example, I/O and debug ports including USB, serial, JTAG etc.) must be disabled or protected from unauthorized use. When a port is used for field diagnostics, the port input must be deactivated and the output must not provide any sensitive information (for example, Wi-Fi password/keys, tokens) which could compromise the device. (L1, L2, L3)
   - 7-2 (Tamper-resistance): Manufacturers should implement tamper evident measures (E.g. plastic seal around the case of the device) to provide a visual indication of tampering.
   - 7-3 (TRNG): Devices should incorporate True Random Number Generator with hardware based source of entropy for key generation.

8. Device Management
   - 8-1 (Device Management): Access to any client application that provides device management interface should only be granted after successful user authentication and authorization. The client application should be able to lock an account or to delay additional authentication attempts after too many failed authentication attempts.

- 8-2 (Administration): Once device has been provisioned successfully, it should not allow any local admin interface. Any device management should be accomplished through the official client applications that first successfully authenticate user in the cloud.
- 8-3 (Provisioning): Provisioning should be accomplished in a way that protects from attacks that can be carried out by an attacker without gaining physical access to the device. Provisioning should protect against threats that would allow someone in the proximity to trick either the device, user or the provisioning application to trust the entity controlled by the attacker (E.g. another device, rogue application, attacker’s account).
- 8-4 (Device Reset): Factory reset should remove all sensitive data stored on the device during or after provisioning and restore it to the factory state.
- 8-5 (Vulnerability Management): Manufacturer should have a defined process for Security Vulnerability Management. Disclosed vulnerabilities should be acted on in a timely manner.

Works with SmartThings (WWST) Devices

SmartThings IoT Devices can connect to the SmartThings cloud in a several ways. First, a Direct-connected Device can connect *directly* to the SmartThings cloud; second, a Cloud-connected Device can connect *indirectly* through a third-party cloud by way of the SmartThings Connector. Thirdly, a Hub-connected Device can connect through a *hub*. More details on SmartThings ecosystem can be found at https://smartthings.developer.samsung.com/docs/platform-basics.html



- Direct-connected Devices:

  The Device Security Requirements must be applicable to the Direct-connected Devices for securely connecting to the SmartThings cloud. Additionally, vendors are responsible to implement the necessary security controls to protect their services.

- Hub-connected Devices:

  We assume secure local connections using ZigBee, Z-Wave, Bluetooth, have been established between the Hub and the Things. The Hub device must follow the Device Security Requirements. The Hub vendor is responsible for any additional necessary security controls for their services.

- Cloud-connected Devices:

  We assume a secure connection has been established from a device to the third-party cloud for Cloud-connected Devices. However, we still require to establish a secure connection between the SmartThings cloud endpoints and the third-party cloud endpoints. For the cloud-to-cloud connections, mutual TLS/IP whitelisting must be used; or alternatively, server-to-server tokens must be used to access to the SmartThings cloud.

Appendix A: Implementation Notes

This section explores the requirements with a few implementation examples (using X.509 certificates as device credentials). We further discuss additional recommendations to implement some security features so that a device meets the security requirements.

1. Hardware Root-of-Trust (RoT)
   A Root-of-Trust contains implicitly trusted elements, such as a hardware computing engine, a binary executable and some configuration data and/or cryptographic keys. This Root-of-Trust is essential to many security functions. Among them, secure boot validates the integrity of the modules during the booting process, and allows the applications providing services the trust that the booting/execution environments are authentic.

   However, even if a device is protected with validated boot, if the Root-of-Trust is tampered with a software attack, then it can be compromised. This can pose risks on the applications/services.

   To establish the foundation of the security on a device, the Root-of-Trust must be implemented in hardware during the manufacturing process. For example, the secure boot key (or its hash), used to verify the integrity of the secondary bootloader, must be provisioned on the processor (or alternatively, Secure Elements, or TrustZone) by using a kind of fusing at the factory.

2. Trust Chain
   Two kinds of trust relationships must be established:

   1) when a sequence of binary executables is validated on a device, trust of each validation must be chained to the Root-of-Trust on the device, and

   2) a trust relationship must be established between a device and the cloud when each mutually authenticates itself to the other party.

   A chaining order of trust for each validated modules during secure boot may differ, depending on the implementation, but a trust of each validation must be chained back to the Root –of -Trust.

   To establish a trust relationship between a device and the cloud, each device must be equipped with an X.509 certificate, issued by Samsung CA (Certificate Authority), or a third-party CA accredited by Samsung; and a corresponding key. Also, each device must include the Samsung CA's root certificate by a kind of certificate pinning, to validate server certificates.

If the corresponding key on a device is compromised, the key/certificate will be revoked, and the device will not be allowed to connect.

3. Device Integrity Protection and Detection
   Any software running on a device can be compromised and modified in an unauthorized way by a malware, or by an attacker gaining physical access to the device.

   Each device must provide secure boot to validate the integrity of the security-critical executables; and stop the boot process if any compromise of the integrity is detected. The implementation of secure boot may determine the modules included in the secure boot chain. Bootloaders and the OS are strongly recommended to be included.

   If there are any security-critical executables, such as cryptographic modules or secure service applications, not included in the secure boot chain, they must be validated before their execution. If it detects a failed integrity, it should generate notifications, or be securely logged on a device or in the cloud.

   Also, each device must be able to securely update its firmware. The firmware can be delivered to the device in a binary format preferably by OTA (over-the-air); or by other ways, such as USB, UART, etc., using a variety of protocol mechanisms (e.g., CoAP, HTTP).

   The hardware Root-of-Trust must be based for validation of the integrity and authenticity of the signed updates. The secure update/boot should provide a way to detect a firmware downgrade and stop if any downgrades have been detected, especially in case that the downgrade has any security vulnerability.

   In addition to the secure update of the firmware, the vulnerabilities of the software running on a device should be handled by a process of security vulnerability management. Once vulnerabilities have detected, the vendor should be able to provide timely security patches, and the device should be able to update its software, to mitigate the threats posed by the vulnerabilities.

4. Protected Communication
   An attacker may try to steal or modify either the data (for example, tampering with the streaming of a security camera), or the user's credentials such as Wi-Fi credentials or access tokens being transmitted, by having access to the network between a device and the cloud. Also, an attacker may try to send unauthorized commands or a false event (e.g., fire alarm) to the victim's devices.

   All transmitted data between each device and the SmartThings cloud must be protected for confidentiality and integrity. The TLS 1.2 and industry standard cryptographic algorithms should be used to protect the communication. If open source crypto implementation is used, even though using validated crypto implementation is strongly recommended for cryptographic operations such as key generations, random number generations, encryptions, and hashing, the latest version applied with all security patches should be used.

5. Protected Storage
   Malware or unauthorized code may try to steal keys, access tokens, or sensitive user data stored on a device. To mitigate those threats, each device must provide secure storage to ensure the confidentiality and integrity of the data.

   Secure storage for L3 must be hardware backed and all cryptographic keys must be stored encrypted in hardware backed secure storage. If a key is used to encrypt the data in secure storage, a unique key must be provisioned at the factory and used on each device. Some examples of hardware protections of keys for secure storage are: using a TPM (Trust Platform Module), a Secure Element, TrustZone, a PUF (Physically Unclonable Function), or processor's internal fuses.

   Also, secure storage for L1 and L2 must be protected at the least in software and all cryptographic keys must be stored encrypted in a secure storage. If a key is used to encrypt the data in secure storage, a unique key should be provisioned at the factory and used on each device like for key derivations. And white box cryptography or obfuscation techniques should be considered to protect the keys.

6. Authentication
   An attacker may try to register a rogue device not certified, in order to hijack sensitive events or send an event for his benefits. Or an attacker may try to trick a device to connect to a spoofed server.

   Each device must be identified with a device ID. A uniqueness of a device ID must be established during the first time the device registration to the cloud occurs. A unique device ID can be used to detect a device cloning using the same device ID.

   Each device must provide a certificate-based mutual authentication with the cloud, and should be able to establish a TLS connection based on the mutual authentication.

7. Protected Local Ports
   An attacker may modify the device's configuration and codes using the hardware debug ports. For examples, an attacker can reverse engineer the binary code, reload it with physical access to the device, analyse the protocols to the cloud, and obtain credentials such as access tokens, using the hardware debug ports and debugging tools.

To mitigate the threats, the device’s local ports (for example, I/O and debug ports including USB, serial, JTAG etc.) must be disabled or protected from unauthorized use. When a port is used for field diagnostics, the port input must be deactivated and the output must not provide any sensitive information (for example, Wi-Fi password/keys, tokens) which could compromise the device. Also, unnecessary network ports on a device must be disabled or properly protected.