Cyber Security Baseline for Consumer Internet of Things Device

Xiaomi AIoT Security Laboratory

December 2021

Contents

04 Device Communication

05 Data Security and Privacy

06 Business Function Logic

Preface

In recent years, with the development of technology and consumers' increasing demand for consumption upgrade, the market of Internet of Things (IoT) devices has also begun to flourish. In terms of the number of products, the quantity of IoT devices is expected to exceed 31 billion in 2021 and more than 75 billion in 2025 according to the research report of Statistics; from the perspective of market value, the IoT market space equals 17 billion dollars in 2019 while expected to exceed 81 billion dollars in 2025 according to the research report of the IoT analystic; from the data generated by IoT devices, the amount of data generated by IoT devices will grow at a compound annual growth rate of 28.7% from 2018 to 2025, reach 79.4 ZB by 2025 as expected according to IDC's research report.

The rapid acceleration of the IoT market means that IoT devices will quickly enter people's daily lives and be widely used in various scenarios. Different from traditional home appliances, consumer IoT devices, which actually are the extension of the Internet in users' various life scenarios, are able to sense objects, transmit and process information intelligently. Therefore, while facilitating people's lives, a large amount of users' personal information and even sensitive information will be collected, transmitted, stored and processed inevitably. Domestic and Global regulatory agencies as well as international standardization organizations have consecutively issued security and privacy laws, regulations and standards for IoT products. At the same time, consumers have increasingly higher requirements for security and privacy protection capabilities of IoT devices.

In case of the rapid increase of the amount of IoT devices, with the sharp increase of the data generated and the ever-increasing attention of supervision, standards organizations and users on security and privacy protection, how companies should ensure the security and privacy of IoT devices, so that users, regulators, and partners can be at ease and trust their IoT products, ensuring product development under compliance, has become the primary problem for companies to solve. However, there is no consumer IoT terminal devices security baseline guides that can be publicly queried and implemented in China, which can help companies improve the security and protection capabilities of their IoT terminal devices.

Xiaomi AIoT Security Laboratory has developed this baseline guide based on years of experience in security testing and compliance with domestic and global laws, regulations and standards. This guide is intended to help domestic IoT companies to deal with the above challenges by providing an open, convenient, and practical knowledge base. When designing and developing consumer IoT terminal products, companies can use this guide to avoid some basic security and privacy protection risks, so as to quickly improve the security and privacy protection capabilities of their products.

Scope

This baseline mainly describes the security baseline requirements for consumer IoT devices.

Normative References

The following documents are necessary for the application of this document. For dated references, only the dated version applies to this document. For undated references, the latest version (including all amendments) applies to this document.

ETSI EN303645

Cyber Security for Consumer Internet of Things: Baseline Requirements

GB/T35273-2020

Information security technology—Personal information security specification

Regulation (EU) 2016/679

General Data Protection Regulation

01

Device Hardware

1.1 Physical debug interface

1.1.1 Debug interfaces disabled by default

Consumer IoT devices (hereinafter referred to as "devices") should disable debug interface such as UART, JTAG, SWD and etc. at the factory by default. If the debug interface needs to be turned on for reasons such as after-sales problem analysis, it should be turned on after special operations (e.g. special key combinations, private USB Dongle serial use, tilting the device to power on, etc.) according to the capability of the device sensors and so on, in order to reduce unnecessary exposure of the physical debug interface and information transmission.

1.1.2 Debug interface silkscreen on the PCB

The device should remove the debug interface silkscreen from the PCB (e.g. visible TX, RX) to prevent reverse engineering.

1.1.3 Debug interface should disable information input by default

When the debug interface is turned on for specific needs, the debug interface should disable information input by default to prevent tampering with the device firmware or reading of locally stored sensitive information.

1.1.4 Debug interface should prevent from printing sensitive data

When the debug interface is enabled for certain needs, only output logs that does not contain sensitive user and device information (sensitive information includes sensitive security parameters such as key, token, and personal sensitive information such as password, Wi-Fi password, etc., see Appendix C Personal Sensitive Information for details) . For full data stream logs output, such information needs to be masked (e.g. : password :********)

1.2 Local data storage

1.2.1 Encryption of sensitive information

Device should encrypt the sensitive information stored in storage chips (flash, nand, emmc, etc.). The encryption scheme can be achieved by using an integrated security chip or by partitioning the operating system for encryption.

1.2.2 Chip read protection

The device MCU should enable the chip read protection mechanism to prevent unauthorised reading of device information via the debug interface. For example, when attempting to read the chip contents/ memory via the debug interface.

1.3 Communication Link Data Transmission

The communication data itself transmitted in the hardware communication link (IIC/SPI) should be encrypted by the device. It's advisable to use a security chip to ensure the security of the encryption key, supplemented by the mechanism of mixing up real data with fake data before transmission in order to prevent attackers from stealing communication key and chip instructions by sniffing through the hardware channel.

1.4 Secure boot

The device chip should support Secure Boot (Secure Boot principle as shown in the figure 2), where the firmware (uboot, kernel, rootfs) or the key partition of the Flash is verified to be legally loaded at boot time to ensure the legality and integrity of the system in the memory chip is verified before it can boot normally.

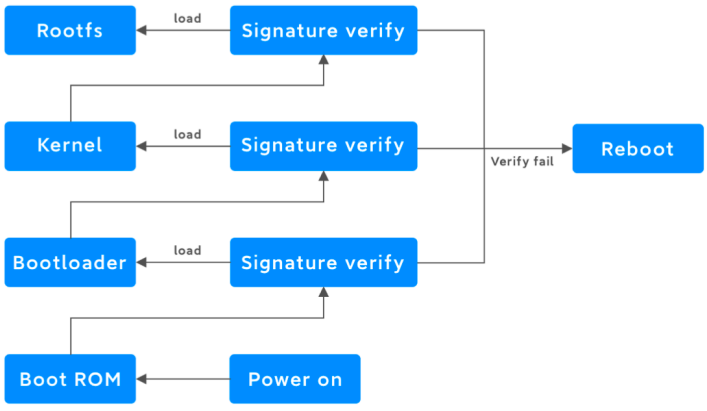


Figure 1–Secure boot

1.5 Protection against physical dismantling

Outdoor (public areas: e.g. outside gates) devices should have protection or warning mechanisms in the event of violent removal or dismantling:

● The enclosure is resistant to removal: Device should not expose screw holes or snaps that can partially or completely remove the device. The enclosure of the device should be designed with a one-piece structure. If screws or snaps must be exposed due to business needs, it is advisable to use special types of fixing screws to fix the device, and use structural adhesives that meet the requirements of the strength and structure to seal and protect the enclosure screws to prevent the device from being easily removed from outside.

● Structural protection of electronic components: Electronic components of the device with control functions (including but not limited to MCU, interfaces, wiring) should be structurally protected with sufficient strength (e.g. designed in the indoor part of the device) to prevent direct exposure if the device is removed from the outside enclosure.

● Dismantling alarms: If the device sensors have the capability to detect the dismantling of the device, it is appropriate to sound an alarm and record and report the dismantling event to the device manager.

Note: Door locks and door bells should meet all requirements of 1.5.

1.6 Protection against EMP Attack

High-security level devices should be equipped with an electromagnetic shield outside the chip to prevent device from EMP Attack, which may cause device logic or operation exception, thereby leading to device crash or circuit burn (such as a EMP Attack to a door lock device can lead to unauthorized unlocking).

1.7 Smart door lock cylinders and door cards

● True cylinder

Smart door locks (true cylinder) should have the clutch assembly placed in the lock body inside the door to ensure that the cylinder from being unlocked by violence or special tools (professional unlocking tools are already available) by enabling the lock cylinder through the external lock body gap. It is strongly recommended that smart door locks are fitted with true cylinders and that Class C cylinders are used.

● Fake cylinder

Smart door locks (fake cylinder) should be fixed with a metal grid between the keyhole and the locking mechanism, and the door should be fully closed with a metal stopper between the lock body and the door to prevent others from opening the lock through the keyhole and the lock body gap by touching the locking mechanism.

● NFC

CPU cards should be used as unlocking cards for smart door locks. ID cards or M1 cards, which can be easily sniffed and copied, should not be used to unlock the door.

Type	Advantages	Disadvantage	Recommendation
CPU Card	High security. Large user space, fast reading speed	Slightly more expensive	Recommended
ID Card	Cheap price	Easy to be copied, low security	Should not be used
M1 Card	Support read and write	Easily sniffed, cracked and copied, slightly more expensive	Should not be used

1.8 Unique device identifiers to prevent tampering

Unique hardware identifiers should be stored on the device and protected from tampering by one-time programming, security chips and other technologies.

02

Device Software

2.1 Software Update

2.1.1 Firmware update package integrity and validity

Before updating the device firmware (OTA from remote or local update), the firmware package integrity hash should be performed to obtain the firmware package digest, and then the digest should be signed and verified for validity using public and private keys (see Nordic Routine[1]) to confirm the package integrity and validity before updating to prevent tampering or replacement of the firmware package.

The firmware update package integrity hash shall be performed using a secure hash algorithm (see 5.1 Hash algorithms) and the integrity credentials shall be transmitted in an encrypted communication channel between the device and the server.

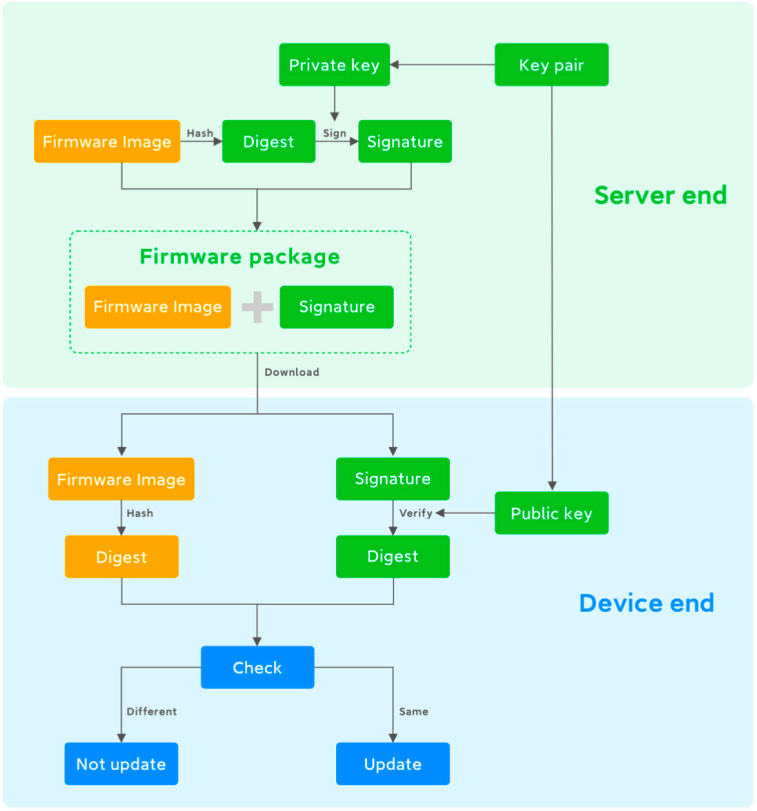


Figure 2-Validity and integrity verification of firmware upgrade package

2.1.2 Anti-firmware downgrade

When updating the device via the OTA function, the device should reject old firmware updates in order to prevent some historical bugs or security risks from being re-exposed. For special needs (e.g. testing, repair, etc.), the device can be flashed after checking the firmware via SD card or wire flashing.

2.1.3 Recovery mechanism after a failed software update

Software updates to the device may encounter special circumstances such as power failure, which may cause the device to fail. Therefore, the device should be equipped with an OTA dual partition backup mechanism to ensure that it can recover to the available software version in the event of an update failure, preventing the update failure from damaging the availability of the device.

2.1.4 LAN OTA update

The device should have the LAN OTA update capability disabled by default, and if it is enabled, it should meet 2.1.1 firmware update package integrity and validity to prevent attackers from performing malicious firmware OTA updates to the device via the LAN.

2.1.5 Third party component updates

Device manufacturers should maintain a list of third party commercial and open source components for product hardware and software during product development and use it to continuously monitor the security of the identified hardware and software components.

Third-party software/libraries used in device should be up-to-date or, if special requirements preclude the use of the latest version, should be subject to a security assessment. Where security vulnerabilities are identified, they should be updated to the latest fixed version.

Note: Software/library vulnerabilities can be found at:
https://www.cvedetails.com/product-list.php

2.1.6 MCU IAP update mechanism

MCUs supporting wired-flash devices should have a secure update mechanism, such as a dedicated USB Dongle package that ensures data encryption or signature verification of the MCU's IAP process to avoid the risk of tampering with the MCU firmware logic.

2.2 Service and ports minimisation

The device should disable high-risk management services or information data services by default, such as FTP, SSH, Telnet, HTTP, ADB, etc. It should also disable ports for IoT SDK control services that are not necessary for data interaction and control implementation.

2.3 Code Repository Management

Device-related code repository should not be uploaded to public code repositories such as Github and Gitee or other public, semi-public services such as Baidu Netdisk without permission to prevent source code leakage.

03

Device OS

3.1 General OS

3.1.1 Bootloader start-up

The device bootloader should not reserve interrupt time before the system boots, and the Delay should be set to 0 to prevent access to the SHELL interface to gain control of the device by modifying the boot parameters.

When the device OS bootloader (Bootloader/U-Boot) starts abnormally, the device OS should automatically reboot to avoid exposing the OS bootloader console interface after an abnormal boot, in order to prevent an attacker from gaining access to the console interface by mistakenly implanting it to gain permission to tamper with the device boot parameters and thus take control of the device.

3.1.2 User account passwords (with login UI)

For devices with a user registration/login interface, if a default password is pre-set for the user account/administration backend of the device, the password should be a randomly generated, per-device, Strong Password, and the user should be forced to change the default password upon first login. If the device does not have a default password for the user account/administration backend where the user registers/logs into the device, the device shall require the user to set their own Strong Password when registering or logging in for the first time.

Strong Password: A password of 10-14 digits in length containing three types of characters: upper case letters, lower case letters, symbols and numbers.

3.1.3 Anti-violence cracking

The device login password should have anti-violence cracking mechanisms, such as human-computer interaction mechanisms (authentication codes), incremental waiting time for failed login authentication and account lockout after a certain number of failed login authentication attempts.

3.1.4 Validating input data

The device should validate data entered through the user interface and API interface to prevent the device from executing external malicious attack code. Data validation methods include filtering data that is outside the scope of processing, filtering and escaping unintended data types, and deploying fuzz tools to detect potential vulnerabilities in the application due to inadequate validation of input data.

3.1.5 Privileged Function Interfaces

The device shall disable by default privileged capabilities or interfaces that provide direct access to the device system (e.g. factory OTA, undisclosed functional interfaces, debugging backdoors, etc.), and shall have a forensic mechanism if really necessary.

3.2 Embedded Linux OS

3.2.1 Serial port binding SHELL

Devices should not bind the system SHELL to a serial debug port such as a UART to prevent wiring to gain control of the device. If there is a special need, the requirements of 3.2.2 must be met.

3.2.2 System default account passwords (without login UI)

Embedded Linux system default users (such as root, admin,etc.) need to set strong passwords and ensure unique-password-per-device. All devices with the same password or empty password are strictly forbidden.

Strong Password: A password of 10-14 digits in length containing three types of characters: upper case letters, lower case letters, symbols and numbers

3.2.3 Root File System Permission

The root file system of the embedded Linux of the device (such as /etc/ and other directories containing boot items) should be set to read-only permission to avoid being forged by malicious attackers.

3.2.4 Externally stored programs and scripts

The embedded Linux operating system of the device should not run programs or scripts in external storage (SD card, USB flash disk, network storage, etc.) by default. If there are special needs, public and private key signature verification should be performed to prevent the system from being implanted with malicious software or scripts.

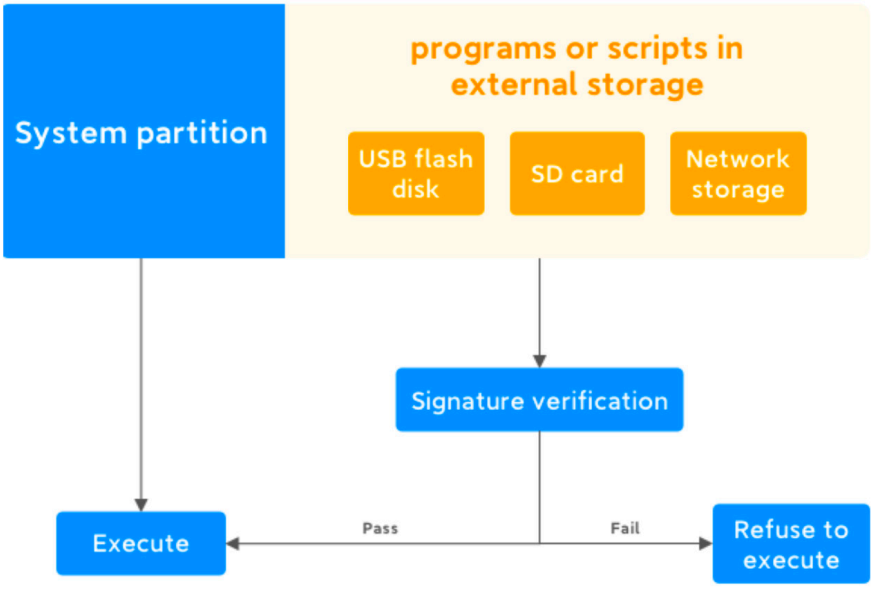


Figure 3-External program and script signature verification

3.2.5 Address Space Layout Randomization (ASLR)

Embedded Linux devices should have Address Space Layout Randomization (ASLR) protection to prevent buffer overflows.

Method: add kernel.randomize_va_space = 2 (Kernel >=2.6.12)in /etc/sysctl.conf

Parameter 2 means that in addition to the library and the stack,the heap is also randomized protection,but it should be noted that ASLR is not responsible for the random protection of the code side and the data side. This work needs to be implemented by the compiler PIE, see: *11.3*

3.2.6 PIE protection

The application should enable the PIE application Address Space Layout Randomization (ASLR) protection option.

Method: gcc compile parameters -pie -fPIE (pay attention to upper and lower case), and need system ASLR support, see:*3.2.5*

3.2.7 Stack Cookie Overflow Protection

When compiling the Linux program code, the CANARY stack overflow protection option of the Linux program should be enabled.

Method: add gcc compilation parameter -fstack-protector-all

3.2.8 Executable Space Protection

When compiling the Linux program code, the Executable Space Protection option of the Linux program should be enabled.

Method: add gcc compilation parameter -z noexecstack

3.2.9 Delete debug symbol table

When compiling the Linux program code, strip function should be applied to drop the debug symbol table to increase the difficulty of reverse analysis and reduce the size of the program.

04

Device Communication

4.1 General Communication

4.1.1 Hard-Coded Key

The devices should not hard-code the key used for transmission encryption or authentication in program code, and measures like unique-password-per-device or generation from PSK should be applied to generate keys.

4.1.2 Communication Channel Encryption

The devices should encrypt the communication channel with other devices or applications, and destroy the session key in time at the end of the session. Refer to 4.2, 4.3, 4.5 for encryption schemes for different communication methods.

4.1.3 Two-way authentication of communication

Two-way authentication should be carried out before the device communicates to verify that both parties' true identities are validate and to check that control rights match those of the identities to prevent overstepping or unauthorised control. The device should use the Mijia standard two-way authentication communication protocols listed below, or undergo a security assessment if it is necessary to use a non-standard protocol for special reasons.

- Device verify the server:

Ethernet devices communicate with the server via the Mijia OTS protocol (based on TLS/DTLS) by first checking with the CA to see if the server certificate is validate.

The Bluetooth device verifies the validity of the server/ APP by using the OOB information in the Mijia Bluetooth Secure Authentication Protocol binding process.

● Server verify the device:

The server side verifies the device's triad information (medium and low security devices) or the unique device identifier in the Mi Home Security Chip (high security devices).

4.1.4 Anti-replay

Device communications should use a rolling code or counter mechanism that allows the device to execute an action command when the requested action count is greater than the device count, to prevent others from replaying control requests by grabbing packets to take unauthorized control of the device.

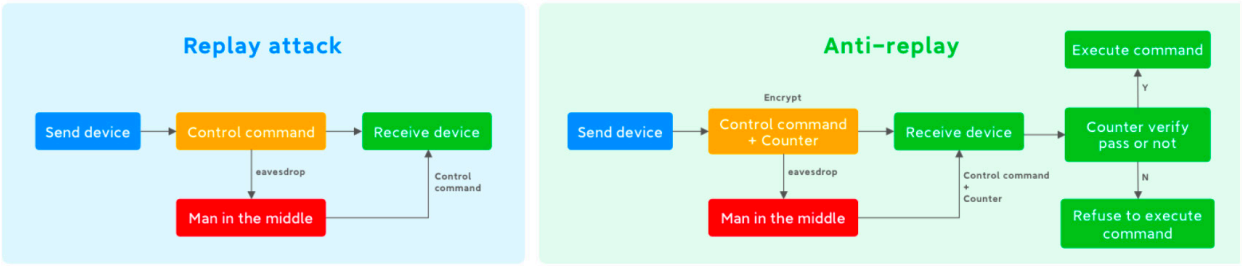


Figure 4-Replay attack and anti-replay

The following solutions can be used to prevent replay under different communication protocols:

● ZigBee: The device should enable the Zigbee protocol frame counter. Enabling method: Set nwkAllFresh to TRUE.

- RF: The device should use rolling code to communicate, refer to keeloq, DST40, Hitag2 and other mature solutions in the industry.[2]
- Wi-Fi: Use HTTPS TLS1.2+ protocol to transmit packet sequence number counters to prevent replay by default.
- Bluetooth: frame counter and encryption with default support of the Mijia Bluetooth authentication protocol or packetCounter and link layer encryption with the standard BLE SM (Security Manager) module.

4.1.5 Unauthorised communication protocols

The communication protocols used by the device should be assessed for security. Unauthorised communication protocols should not be used in order to prevent security vulnerabilities or backdoors in unauthorised protocols from affecting the security of the devices' communications.

4.2 Ethernet

4.2.1 Communication channel encryption

The device should use encrypted transmission protocols to encrypt communications and should use a secure encryption suite of TLS (1.2+) transmission protocols. Avoid the risk of information leakage or tampering due to the use of plaintext transport protocols such as MQTT, HTTP, etc.

Note: The security of the TLS encryption suite can be found at https://ciphersuite.info/cs/

4.2.2 Encryption of sensitive data transmission

The device should use a secure encryption algorithm to additionally encrypt sensitive data when transmitting it.

Note: The security of encryption algorithm can be referred to 5.1 Encryption and Hash Algorithm.

4.2.3 HTTPS Certificate Verification

When the device uses the HTTPS protocol, strict certificate verification should be carried out and the check should not be ignored.

- Linux System

The devices should strictly verify the validity of the server certificate. Using parameters to skip certificate and ignoring certificate verification errors are forbidden.

Tools/Lib	Recommendations for using parameters
curl	Shouldn't use -k parameter
wget	Shouldn't use --no-check-certificate parameter
libcurl	Should set CURLOPT_SSL_VERIFYPEER and CURLOPT_SSL_VERIFYHOST as True

● Android System

Server and client certificates should be strictly verified when devices applications apply SSL encrypted communication services. The devices should not trust any certificates, should not ignore abnormal events (such as empty return or null). If SSLx509 TrustManager is needed to be customized with the checkServerTrusted method rewritten, the server certificate verification must be strictly determined in the method to prevent the communication content from being hijacked, which will lead to communication data leakage or tampering.

4.2.4 Wi-Fi Access Point Password

The function of connecting the device using the Wi-Fi Direct access point to the control application should be strictly evaluated. If such function is necessary, the Wi-Fi Direct access point password shouldn't be fixed or empty, which should follow unique-password-per-device principle or generate password with real random number function every time you use it, then display the password on the screen (devices with screen) or set and store it in advance by the user during binding (devices without screen) to prevent WiFi from unauthorized access.

4.2.5 Wi-Fi Access Point Usage

The Wi-Fi access point established by the device should only allow communication with the device itself, and should not access the device's external network (such as the home network or the

Internet) to prevent attackers from penetrating the home network through the device's Wi-Fi.

4.3 BLE

4.3.1 BLE Pairing

BLE devices with physical buttons should perform the bind confirm via physical buttons or enable bind window to avoid the risk of repeated binding or being bound by others without users consent.

4.3.2 Validity Verification of Control Instructions

For devices that support BLE, the session key should be negotiated every time when users log in and it should be used to encrypt transmitting control instructions between the device and the control applications.

4.3.3 BLE Advertising of Sensor Devices

Some devices can advertise the data collected by sensors through BLE, which will be analyzed through BLE gateways for devices linkage or perform as control instructions, thereby affect other devices. The BLE advertising of such devices should apply a secure communication protocol as well as encrypt the advertising data with the beaconkey generated during binding process.

4.3.4 BLE mesh protocol

Bluetooth mesh devices should use secure communication protocols for two-way authentication with mesh devices, gateways and mobile phones, and use session key encryption for sensitive data transmitted.

4.3.5 BLE Protocol Version

BLE devices (such as mouse and speakers) shouldn't apply BLE protocol version less than 4.2 when performing BLE link layer encryption to prevent the device from leaking the BLE Long Term Keys (LTK), which may lead to risks of privacy leakage or device forgery.

4.3.6 BLE Control Instructions Authentication

The devices should detect the BLE binding status of the control application before using the OTA routine instructions provided by the BLE chip manufacturer. Which is intended to prevent the device from denial of service (such as chip reboot, switching workspace, enabling DFU update mode, etc.) due to the logic that may support unauthorized instructions to control the device in the routine.

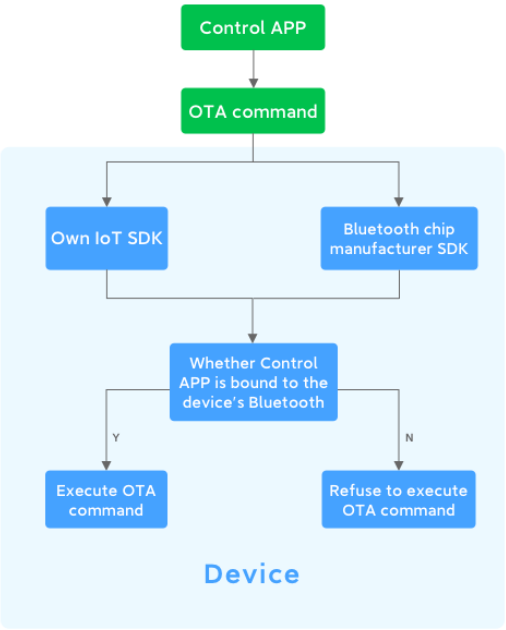


Figure 5-BLE control command authentication

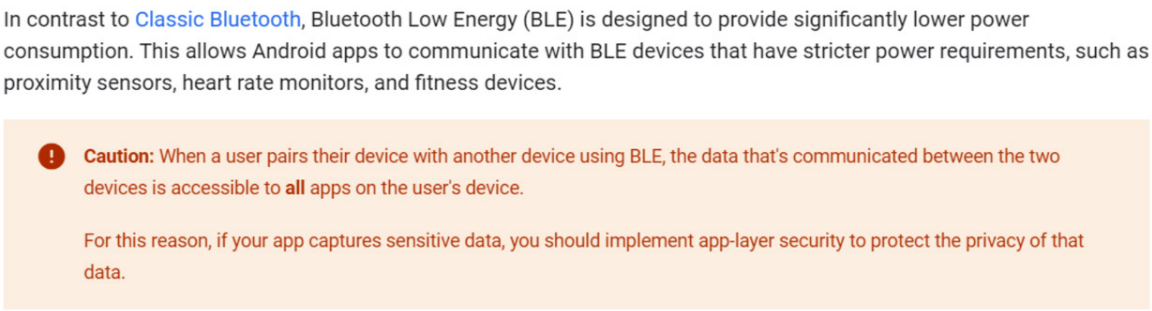
4.3.7 BLE Advertising Anti-tracking Mechanism

The device should apply a random BLE MAC address and encrypt the BLE advertising content; if there's need to advertise identifiable information through the beacon, the device BLE MAC address should change regularly to prevent that others may analyze the advertising content by deploying enough detection devices and track the movement of the device.

4.3.8 BLE Sensitive Information Communication

When the devices communicate with the control application (Android) through BLE, the sensitive information content itself should be encrypted at the application layer to prevent that all applications on the phone can access the data transmitted between the two devices when user pair the phone with other smart devices for data transmission through BLE.

For the risk description, please refer to the development document2 when Android 4.3 introduced BLE[3]:



In contrast to Classic Bluetooth, Bluetooth Low Energy (BLE) is designed to provide significantly lower power consumption. This allows Android apps to communicate with BLE devices that have stricter power requirements, such as proximity sensors, heart rate monitors, and fitness devices.

Caution: When a user pairs their device with another device using BLE, the data that's communicated between the two devices is accessible to **all** apps on the user's device.

For this reason, if your app captures sensitive data, you should implement app-layer security to protect the privacy of that data.

4.4 Zigbee

4.4.1 Zigbee protocol version

Zigbee devices should use the Zigbee 3.0 protocol stack version and use the official install code scheme[4] to prevent attackers from using the default TCLK key from older protocol versions.

4.5 RF

4.5.1 RF Communication Packet Sequence Number

The radio frequency communication data packet of the device should use four bytes as sequence number variable space. A short sequence number is not recommended. This requirement is to avoid the risk that the communication may be brute-forced in a short time.

4.5.2 Hard-coded Communication Key

The radio frequency communication key of the device should be generated by paring and exchange of the transmitter and the receiver without preset in the code.

4.5.3 Communication Frequency

The device should apply the frequency hopping mechanism for communication to prevent channel congestion caused by the fixed radio frequency communication, resulting in abnormal communication of the device.

05

Data Security and Privacy

5.1 Encryption and hash algorithms

● Encryption algorithm:

The device should use a secure encryption algorithm with a key length that meets the minimum requirements of the corresponding algorithm (see the table below), and should not use insecure encryption algorithms such as DES, TDES, RC4, etc.

● Hash algorithm: The device should use a secure hash function of 256 bits or more, such as sha256, sha512, etc. It should not use a hash function with existing security risks, such as md5 or sha1.

Algorithms	Key/Hash Length (bit)
AES	128 / 192 / 256
ECDSA / ECDH	256 / 384 / 512
RSA	2048 /3072/ 4096
DSA	Prime P: 2048 / 3072 / 4096
	Prime Q: 256 / 384 / 512
SM4	128
SHA	256 / 384 / 512

5.2 Random Number Generation Function

Device system should use true random algorithms to generate random numbers required for authentication or encryption.

Type	Generation Method	Recommendation
True / Strong Pseudorandom	/dev/urandom Entropy random ability supported by the chip	Recommendated
Pseudorandomn	srand()、rand() with time(0) as the seed	Not Recommendated

5.3 Telemetry Upload

Telemetry printed by devices, systems and applications should not upload any plaintext or encrypted sensitive information (such as Wi-Fi SSID, password, mobile phone IMEI, geographic location and other sensitive information). If it needs to be reported for use according to evaluation, the role and storage method should be clearly stated in the privacy policy.

5.4 Cross-border Network Requests

The device or control applications should identify the country or region of the current user, and send network requests based on the privacy compliance requirements of different countries or regions to prevent privacy risks caused by cross- border data transmission.

5.5 Cloud storage security

Sensitive user data (video, audio, images, documents, etc.) collected through the device should be stored encrypted on the cloud storage server.

5.6 Cloud data deletion function

The IoT platform should provide users with a convenient function to delete data in the cloud. The scope of deletion includes personal information generated and stored in the cloud by users during the use of the device and its associated services.

5.7 Restore Factory Settings

After the device is restored to factory settings, all user data and settings in the device (such as user usage records, settings, NFC door cards, eSIM card records, etc.) should be cleared completely.

06

Business Function Logic

6.1 Device Binding Status

If the unbound device is not bound after being powered on for 30 minutes, the binding status should keep shut off before it being repowered to avoid the risk that the device always accepts binding requests which may lead to malicious binding.

6.2 Binding Confirmation

When the device is being bound, the user should be asked to confirm of binding on device (OOB) when the product capability permits. Vehicles and high security level devices (such as scooters, doorbells, door locks, outdoor monitoring) should be mandatory to require OOB to prevent the device from being maliciously bound.

6.3 Anti-rebinding

For devices with reset capability, they should be reset before accepting the binding request again to prevent repeated binding or malicious binding and control by other users.

6.4 Strong Binding Relationship

Outdoor devices, high security level devices should establish a strong binding relationship between the device and the account in the cloud, that is, the cloud records the binding relationship between the device ID and user ID.

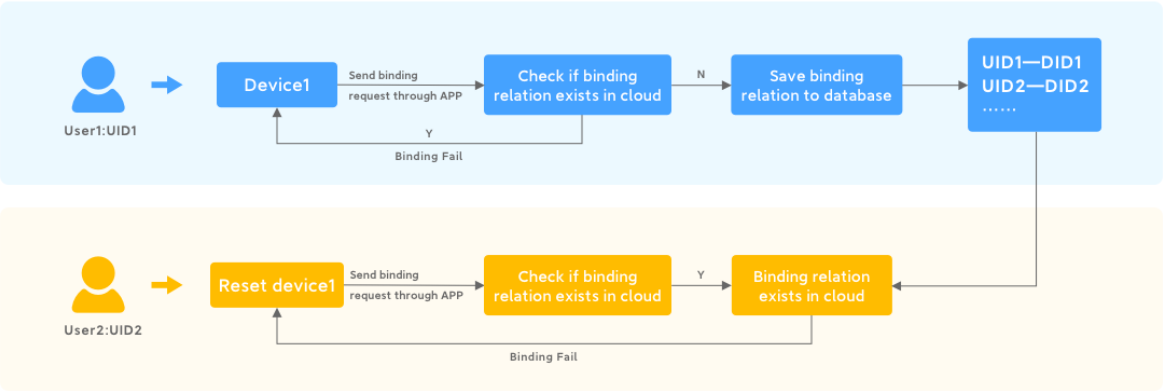


Figure 6–Strong binding

6.5 Device log security monitoring

The IoT platform should collect key logs from the device side (e.g. device networking status, usage status, upgrade status, etc.) and such device logs should be used for security monitoring. Anomalies are identified through statistics and analysis of log information to detect potential device or IoT platform security risks.

6.6 Guidance on security settings

If the device is equipped with security features, one of the following methods should be used to help the user easily enable these security features.

1) Default setting:

The device automatically turns on the necessary safety features by default when it is initialised.

2) Software guidance:

The device software or control application provides switches and instructions for the relevant security features via the software UI, or guides the user to enable them via pop-up windows, etc.

3) Instruction manual:

The user is informed of the way and steps to enable the security features of the product through a paper or electronic version of the product security setup guide.

Terms and Definitions

The following terms and definitions apply to this document

1 Consumer IoT Device

Network connected devices (referred to as “devices” in the article), consumer IoT terminal devices in this specification refer to wireless(Wi-Fi, BLE/Mesh, Bluetooth classic, Zigbee, NFC, RF), wired (RJ45) connectivity and network organization capability, or any smart terminal product with firmware logic update capabilities.

Note 1: Consumer IoT devices are also commonly used in commercial environments. These devices are still classified as consumer IoT devices.

Note 2: Consumer IoT devices are usually available for purchase by consumers in retail environments. Consumer IoT devices can also be commissioned and/or professionally installed.

2 User

Natural person or organization

3 Critical security parameter

critical security parameter: security-related secret information whose disclosure or modification can compromise the security of a security module.

EXAMPLE: secret cryptographic keys, authentication values such as passwords, PINs, private components of certificates.

4 Public security parameter

security related public information whose modification can

compromise the security of a security module

EXAMPLE 1: A public key to verify the authenticity/integrity of software updates.

EXAMPLE 2: Public components of certificates.

5 Sensitive security parameters

critical security parameters and public security parameters

6 Personal data

any information relating to an identified or identifiable natural person

7 Sensitive information

Sensitive information is a collective term for sensitive security parameters and personal information.

8 Debug interface

physical interface used by the manufacturer to communicate with the device during development or to perform triage of issues with the device and that is not used as part of the consumer-facing functionality

EXAMPLE: Test points, UART, SWD, JTAG.

9 Logical interface

software implementation that utilizes a network interface to communicate over the network via channels or ports

10 Network interface

physical interface that can be used to access the functionality of consumer IoT via a network

11 Physical interface

physical port or air interface (such as radio, audio or optical) used to communicate with the device at the physical layer

EXAMPLE: Radios, ethernet ports, serial interfaces such as USB, and those used for debugging.

12 Security update

software update that addresses security vulnerabilities either discovered by or reported to the manufacturer

NOTE: Software updates can be purely security updates if the severity of the vulnerability requires a higher priority fix.

13 Telemetry

data from a device that can provide information to help the manufacturer identify issues or information related to device usage

EXAMPLE: A consumer IoT device reports software malfunctions to the manufacturer enabling them to identify and remedy the cause.

14 Authentication mechanism

method used to prove the authenticity of an entity

NOTE: An "entity" can be either a user or machine.

EXAMPLE: An authentication mechanism can be the requesting of a password, scanning a QR code, or use of a biometric fingerprint scanner.

15 Flash

Flash is a kind of a memory chip, and the data in it can be modified through a specific program. FLASH often means Flash Memory in the fields of electronics and semiconductors, that is,

"flash memory", the full name is Flash EEPROM Memory.

16 Bootloader

In an embedded operating system, BootLoader runs before the operating system kernel runs. It can initialize hardware devices and establish a memory map to bring the system's software and hardware environment to a suitable state, so that it is ready for the final call to the operating system kernel.

17 U-boot

U-Boot is a boot loader mainly used for embedded systems. It can support a variety of different computer system structures, including PPC, ARM, AVR32, MIPS, x86, 68k, Nios and MicroBlaze.

18 Kernel

Embedded kernel is an abstraction layer between embedded hardware and software. It is called "kernel" in Linux terminology and can also be called "core". The main modules (or components) of the Linux kernel are divided into the following parts: storage management, CPU and process management, file system, device management and drivers, network communication, and system initialization (boot), system calls, etc.

19 Beacon

The BLE Beacon is a dvertising protocol based on the BLE protocol.

20 ASLR

Address Randomization (ASLR) is a security protection

technology against buffer overflow. By randomizing the layout of linear areas such as heap, stack, and shared library mapping, it increases the difficulty for attackers to predict the destination address, which can further prevent overflow attacks.

21 Identification Card

The full name of ID card is Identification Card. It is a non-writable proximity card with a fixed number. It mainly includes EM format of Taiwan SYRIS and HIDMOTOROLA of the United States. The ID card and other types. like the magnetic card, only uses the "card number". Except for the card number, there is no confidentiality function inside the card, and the "card number" is public and exposed. So ID card is "inductive magnetic card".

22 M1 card

M1 chip card refers to the abbreviation of the chip produced by NXP, a subsidiary of Philips. The full name is NXP Mifare1 series. Two types including S50 and S70 are commonly used. Common ones are card type and keychain type.

23 Routine

A routine is a collection of functional interfaces or services provided externally by a certain system. For example, the APIs and services of the operating system are routines.

Abbreviation

The following abbreviations apply to this document.

API	Application Programming Interface
IoT	Internet of Things
IP	Internet Protocol
JTAG	Joint Test Action Group
OTA	Over The Air
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus
SSH	Secure Shell
MAC	Media Access Control
TLS	Transport Layer Security
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
FTP	File Transfer Protocol
UID	User ID
DID	Device ID
SWD	Serial Wire Debug
NAND	Not And
EMMC	Embedded Multi Media Card
MQTT	Message Queuing Telemetry Transport
TCLK	Trust Center Link Key
ASLR	Address Space Layout Randomization·
NX Stack	No Execute Stack
PIE	Position Independent Executables
OOB	Out of Band
XML	eXtensible Markup Language
AES	Advanced Encryption Standard
DES	Data Encryption Standard
RSA	An asymmetric encryption algorithm named after the first letter of the last name of the three co-authors of the algorithm
TX	transmit
RX	Receive
PSK	Pre-Shared Key
ADB	Android Debug Bridge
RF	Radio Freqency
JSAPI	JavaScript Application Programming Interface

Appendix A

Product Security Levels:

IoT products are classified into three security levels: high, medium and low, based on the degree of maximum possible impact on the device, the user and the IoT platform if the confidentiality, integrity and availability of the product are compromised. The core concerns and corresponding impact levels for determining the security level of a device are shown in the table below.

Security attribute	Concerns	Level of impact			Remarks
		Low	Medium	High	
Confidentiality	The higher the sensitivity of the personal information collected by the device, the greater the impact on the user if confidentiality is compromised.				
	Basic personal information	√			
	Personal Education & Work Information	√			
	Personal information on commonly used devices	√			
	Personal communication information		√		
	Contact information		√		
	Personal identification information			√	
	Personal biometric information			√	
	Web identifier information			√	
	Personal health and physical information			√	
	Personal property information			√	
	Personal Internet Records			√	
	Personal location information			√	
	Other sensitive information			√	

Integrity	The extent of the potential impact on the user's personal safety, property security and spirit in the event of possible malicious tampering or other damage of integrity of the comprehensive device or APP terminal				
	Minor impact	√			
	Limited impact		√		
	Severe impact			√	
Availability	The extent of the impact on users or on public opinion in the event of complete unavailability of the device and associated services.				
	Minor impact	√			
	Limited impact		√		
	Severe impact			√	

See Appendix C for definitions and examples of various types of personal and sensitive information.

Appendix B

The requirements to be met for device of different safety levels are as follows.

Chapter	Title	Title	Medium	High	Applicable conditions
Device Hardware	1.1 Physical debug interface				
	1.1.1 Debug interfaces disabled by default		√	√	
	1.1.2 Debug interface silkscreen on the PCB			√	
	1.1.3 Debug interface should disable information input by default		√	√	
	1.1.4 Debug interface should prevent from printing sensitive data		√	√	
	1.2 Local data storage				
	1.2.1 Encryption of sensitive information		√	√	
	1.2.2 Read Protection for chips			√	
	1.5 Protection against physical dismantling			√	Smart doorlocks, doorbells
	1.6 Protection against EMP Attack			√	Smart doorlocks
	1.7 Smart door lock cylinders and door cards			√	Smart doorlocks
	1.8 Unique device identifiers to prevent tampering			√	
Device Software	2.1 Software Update				
	2.1.1 Firmware update package integrity and validity	√	√	√	
	2.1.2 Anti-firmware downgrade			√	
	2.1.3 Recovery mechanism after a failed software update		√	√	
	2.1.4 LAN OTA update	√	√	√	
	2.1.5 Third party component updates		√	√	
	2.1.6 MCU IAP update mechanism			√	
	2.2 Service and ports minimisation	√	√	√	
	2.3 Code Repository Management	√	√	√	

Chapter	Title	Title	Medium	High	Applicable conditions
Device OS	3.1 General OS				
	3.1.1 Bootloader start-up		√	√	
	3.1.2 User account passwords (with login UI)	√	√	√	
	3.1.3 Anti-violence cracking	√	√	√	
	3.1.4 Validating input data	√	√	√	
	3.1.5 Privileged Function Interfaces		√	√	
	3.2 Embedded Linux OS				
	3.2.1 Serial port binding SHELL		√	√	
	3.2.2 System default account passwords (without login UI)			√	
	3.2.3 Root File System Permission		√	√	
	3.2.4 Externally stored programs and scripts		√	√	
	3.2.5 Address Space Layout Randomization (ASLR)		√	√	
	3.2.6 PIE protection		√	√	
	3.2.7 Stack Cookie Overflow Protection		√	√	
	3.2.8 Executable Space Protection		√	√	
	3.2.9 Delete debug symbol table	√	√	√	
Device Communication	4.1 General Communication				
	4.1.1 Hard-Coded Key		√	√	
	4.1.2 Communication Channel Encryption	√	√	√	
	4.1.3 Two-way authentication of communication			√	
	4.1.4 Anti-replay		√	√	
	4.1.5 Unauthorised communication protocols	√	√	√	
	4.2 Ethernet				
	4.2.1 Communication channel encryption	√	√	√	
	4.2.2 Encryption of sensitive data transmission		√	√	
	4.2.3 HTTPS Certificate Verification	√	√	√	
	4.2.4 Wi-Fi Access Point Password		√	√	Relies on devices to establish Wi-Fi hotspots for connection control (e.g. car recorders, sports cameras, etc.)
	4.2.5 Wi-Fi Access Point Usage		√	√	

Chapter	Title	Title	Medium	High	Applicable conditions
Device Communication	4.3 BLE				
	4.3.1 BLE Pairing	√	√	√	
	4.3.2 Validity Verification of Control Instructions	√	√	√	
	4.3.3 BLE Advertising of Sensor Devices	√	√	√	
	4.3.4 BLE mesh protocol	√	√	√	
	4.3.5 BLE Protocol Version	√	√	√	
	4.3.6 BLE Control Instructions Authentication	√	√	√	
	4.3.8 BLE Sensitive Information Communication			√	
	4.4 Zigbee				
	4.4.1 Zigbee protocol version	√	√	√	
	4.5 RF				
	4.5.1 RF Communication Packet Sequence Number	√	√	√	
	4.5.2 Hard-coded Communication Key		√	√	
	4.5.3 Communication Frequency			√	
Data Security and Privacy	5.1 Encryption and hash algorithms	√	√	√	
	5.2 Random Number Generation Function		√	√	
	5.3 Telemetry Upload	√	√	√	
	5.4 Cross-border Network Requests	√	√	√	
	5.5 Cloud storage security	√	√	√	
	5.6 Cloud data deletion function	√	√	√	
	5.7 Restore Factory Settings	√	√	√	
Business Function Logic	6.1 Device Binding Status	√	√	√	
	6.2 Binding Confirmation			√	
	6.3 Anti-rebinding	√	√	√	
	6.4 Strong Binding Relationship			√	Outdoor device
	6.5 Device log security monitoring	√	√	√	
	6.6 Guidance on security settings			√	

Appendix C

Personal information

Personal information refers to various information recorded electronically or in other ways that can identify a specific natural person alone or in combination with other information or reflect the activities of a natural person.

Table A.1 Examples of personal information	
Basic personal information	Name, date of birth, gender, ethnic group, nationality, family relation, address, personal phone number, email address, etc.
Personal identity information	ID card, military officer certificate, passport, driver's license, employee ID, pass, social security card, resident certificate, etc.
Personal biometric information	Personal gene, fingerprint, voice print, palm print, auricle, iris, and facial recognition features, etc.
Online identity information	A PI subject's account, IP address, and personal digital certificate.
Physiological and health information	Records generated in connection with medical treatment, such as pathological information, hospitalization records, physician's instructions, test reports, surgical and anesthesia records, nursing records, medication administration records, drug and food allergy, fertility information, medical history, diagnosis and treatment, family illness history, history of present illness, and history of infection, and personal health information such as weight, height, and lung capacity.
Personal education information	Personal occupation, position, work unit, educational background, academic degree, educational experience, work experience, training records, transcript, etc.
Personal property information	Bank account, authentication information (password), bank deposit information (including amount of funds, payment and collection records), real estate information, credit records, credit information, transaction and consumption records, bank statement, etc., and virtual property information such as virtual currency, virtual transaction and game CD Keys.
Personal communication information	Communications records and content, SMS, MMS, emails, data that describe personal communications (often referred to as metadata), etc.
Contact information	Contacts, friend list, list of chat groups, email address list, etc.
Personal web surfing record	Refers to records of a PI Subject's operations stored in the logs, including web browsing records, software use records, click records, and favorites.
Information of often used equipment	Refers to the information describing the general conditions of the equipment often used by an individual, including hardware serial number, equipment MAC address, list of software, and unique equipment identifier (e.g. IMEI/Android ID/IDFA/Open UDID/GUID, SIM card IMSI information).
Personal location information	Including records of whereabouts, precise location information, accommodation information, longitude and latitude.
Other information	Marriage history, religious preference, sexual orientation, undisclosed criminal records, etc.

[Source: Appendix A (Informative) Examples of personal information – GB/T35273–2020]

Personal sensitive information

Personal sensitive information refers to personal information that, once leaked, illegally provided, or misused, may endanger personal and property safety, and easily lead to personal reputation, physical and mental health damage, or discriminatory treatment.

Table B.1 Examples of sensitive personal information	
Personal property information	Bank account, authentication information (password), bank deposit information (including amount of funds, payment and collection records), real estate information, credit records, credit information, transaction and consumption records, bank statement, etc., and virtual property information such as virtual currency, virtual transaction and game CD Keys.
Physiological and health information	The records generated in connection with medical treatment, including pathological information, hospitalization records, physician's instructions, test reports, surgical and anesthesia records, nursing records, medicine administration records, drug and food allergy, fertility information, medical history, diagnosis and treatment, family illness history, history of present illness, history of infection.
Personal biometric information	Personal gene, fingerprint, voice print, palm print, auricle, iris, and facial recognition features, etc.
Personal identity information	ID card, military officer certificate, passport, driver's license, employee ID, social security card, resident certificate, etc.
Other information	Sexual orientation, marriage history, religious preference, undisclosed criminal records, communications records and content, contacts, friends list, list of chat groups, records of whereabouts, web browsing history, precise location information, accommodation information, etc.

[Source: Appendix B (Informative) Identification of personal sensitive information – GB/T35273-2020]

Special categories of personal data

Personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, and the processing of genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural person's sex life or sexual orientation.

[Source: General Data Protection Regulation – Article 9]

Reference

【1】Nordic OTA Routines

https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.sdk5.v15.3.0%2Fexamples_bootloader.html&cp=5_1_4_4

【2】Rolling Code

Rolling codes are used to authenticate device communication and prevent replay attacks.

https://ww1.microchip.com/downloads/en/Appnotes/Atmel-2600-AVR411-Secure-Rolling-Code-Algorithm-for-Wireless-Link_Application-Note.pdf

- Industry proven solutions.

Keeloq: https://blog.csdn.net/kangweijian/article/details/43491047

DST40: https://en.wikipedia.org/wiki/Digital_signature_transponder

Hitag2: https://blog.csdn.net/spenghui/article/details/71930428

【3】Developer documentation for the introduction of BLE in Android 4.3:

https://developer.android.com/guide/topics/connectivity/bluetooth-le

【4】Zigbee 3.0 official Install code scheme:

https://zigbeealliance.org/solution/zigbee/