消费级物联网安全基线

小米AIoT安全实验室

2020.12

目　录
Contents

第九章 Android 应用安全

第十章 通用编码安全

第十一章 Linux 应用编码安全

第十二章 业务逻辑安全

第十三章 数据安全

前　言

近年来，随着科技的发展，以及消费者对智能生活日益增长的消费升级需求，物联网（IoT）设备市场也开始蓬勃发展。从产品数量来看，根据 Statistics 的调研报告，预计 2020 年 IoT 设备数量将超过 300 亿，2025 年将超过 750 亿；从市场价值来看，根据 IoT analystic 的调研报告，2019 年，IoT 市场空间 170 亿美元，预计到 2025 年，IoT 市场空间将超过 810 亿美元；从 IoT 设备所产生的数据来看，根据 IDC 的调研报告，2018–2025 年间，由 IoT 设备产生的数据量将以 28.7% 的复合年增长率增长，预计到 2025 年，IoT 设备数据将达到 79.4ZB。

IoT 市场如此迅猛的增速，意味着物联网设备将迅速进入人们的日常生活并广泛应用于各种场景。不同于传统家电，消费级物联网设备具有感知物体、信息传输、智能处理的特点，其本质是互联网在用户各种生活场景的延伸，因此在方便人们生活的同时，也不可避免的会收集、传输、存储、处理大量的用户个人信息甚至个人敏感信息。国内外的监管机构与国际标准化组织也相继发布了针对 IoT 产品的安全与隐私法律法规和标准规范；与此同时，消费者对 IoT 设备的安全与隐私保护能力的要求也越来越高。

在 IoT 设备数量急速增加，所产生的数据急剧增加，监管，标准组织和用户对产品安全与隐私保护的关注度不断增加的背景下，企业应如何保证物联网设备的安全与隐私，让用户，监管和合作伙伴放心，信任其物联网产品，保障产品合规发展，成了企业首要解决的问题。但当前国内缺少一份可被公开查询、可落地的消费级物联网终端安全基线指南，来帮助企业提升其 IoT 终端产品的安全与保护能力。

小米 AIoT 安全实验室根据多年的安全测试经验积累、并结合国内外法律法规和标准规范，制定了本基线指南。本指南意在帮助国内物联网企业，在应对以上安全挑战时，可以有一个开放，便捷，落地的知识库，让企业在设计和开发消费级物联网终端产品时，可以对照此指南，规避一些基本的安全与隐私保护风险，以快速提升产品的安全与隐私保护能力。

范围

本基线主要描述了针对消费级物联网终端设备的安全基线要求。

规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

ETSI EN303645 《消费级物联网信息安全基本要求》

Cyber Security for Consumer Internet of Things: Baseline Requirements

GB/T35273-2020《信息安全技术 – 个人信息安全规范》

Information security technology—Personal information security specification

01

硬件安全

1.1 物理调试接口

- **设备调试接口应设置为默认关闭**

消费级物联网设备（下文简称“设备”）应在出厂时默认关闭 UART、JTAG、SWD 等调试接口。如因售后问题分析等原因需开启调试接口，应按需飞线通联或根据设备传感器等能力，做出特殊操作后开启（如特殊按键组合、私有 USB Dongle 串接使用、设备倾斜通电等），以减少不必要的物理调试接口暴露和信息传输。

- **设备宜去除 PCB 板上的调试接口丝印**

物联网终端设备宜去除 PCB 板上的调试接口丝印（如明显的 TX、RX），以防止逆向工程。

- **设备调试接口信息传输应遵循最小化原则**

调试接口在某些特定需要而开启后，默认不应开放调试接口的信息输入，并仅允许进行不包含用户与设备敏感信息的 log 输出（敏感信息包含敏感安全参数如 key、token，和个人敏感信息如 password、Wi-Fi 密码等，详见附录 A 个人敏感信息），如需输出完整数据流日志，需要将此类信息遮蔽展示（如：password：********）

1.2 本地数据存储

- **敏感信息加密**

设备应将敏感信息加密存储在存储芯片（flash、nand、emmc 等），加密方案可通过采用集成安全芯片或操作系统分区加密来实现。

- **设备应开启芯片读保护**

设备应开启芯片读保护机制，以防止通过调试接口读取固件。

1.3 通信链路数据传输

设备宜对硬件通信链路（IIC/SPI）中传输的通信数据本身进行加密。宜采用安全芯片来确保加密密钥的安全性，并结合真实与伪数据融合发送的机制，以防止攻击者通过硬件通道嗅探得到协商密钥以及芯片指令。

图 1 – 安全芯片

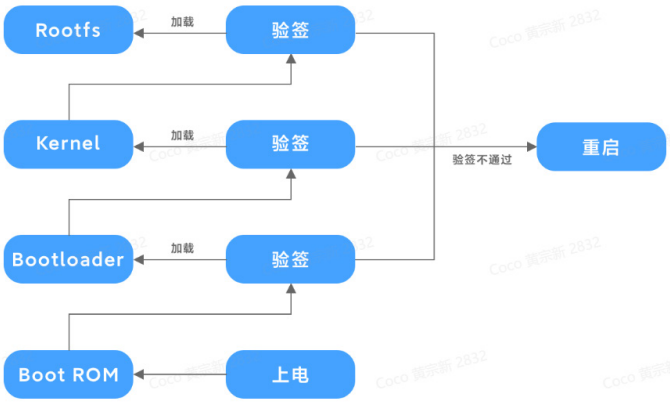

图 2 – 安全启动

1.4 安全启动

设备芯片宜支持安全启动（SecureBoot 原理如图），启动时对固件（uboot、kernel、rootfs）或 Flash 关键分区进行合法加载校验，确保当安全校验通过后系统才能正常启动。

1.5 启动异常

当设备操作系统引导加载程序（Bootloader/U-Boot）启动异常后，设备操作系统应自动重启，避免启动异常后暴露操作系统引导加载控制台（Console）界面，以防止攻击者通过错误植入进入到控制台界面，来获得篡改设备启动参数的权限，从而控制设备。

1.6 MCU IAP 更新机制

部分电源控制类设备 MCU 支持 Type-C CC 引脚的 IAP 能力，这类设备的 MCU 应具备安全更新机制，如封装专用的 USB Dongle 确保对 MCU 的 IAP 过程进行了数据加密或签名验证，以避免 MCU 固件逻辑被篡改导致的设备供 / 用电风险。

1.7 防强电磁攻击

高安全等级设备应在芯片外加装电磁防护罩，以防止强电磁脉冲（EMP）攻击造成的设备逻辑或运行异常，进而导致逻辑错误，设备宕机或电路烧毁（如门锁设备遭遇强电磁攻击后可导致非授权开锁）。

1.8 防物理拆除

室外（公共区域：如大门外）设备宜具备在受到暴力移除或拆卸时的防护或预警机制，如采用非常用类型固定螺丝来固定设备，或设备在检测到被拆解时可发出警报（如门铃）。

1.9 智能门锁开锁

- **真插芯**

智能门锁（真插芯）应将离合组件放置于门内锁体中，以保证即使外部锁体被破坏或舵机电平被控制时，锁芯依然空转无法解锁。防止智能门锁锁芯被暴力或专用工具（已有专业开锁工具）通过外部锁体缝隙劫持锁芯开锁。强烈建议智能门锁产品采用真插芯。

- **假插芯**

智能门锁（假插芯）应在钥匙孔与锁芯机关间用金属格挡并固定，门外锁体与门之间应用金属挡板全部封闭，以防止他人通过钥匙孔、锁体缝隙触动开锁机关开锁。

- **NFC**

智能门锁应使用 CPU 卡作为开锁门卡，不应使用易被嗅探、破解复制的 ID 卡或 M1 卡开锁。

卡类型	优点	缺点	推荐
CPU 卡	安全性高，用户空间大，读取速度快	价格稍贵	推荐使用
ID 卡	价格便宜	容易复制，安全性低	不应使用
M1 卡	可读可写	易被嗅探、破解并复制，价格稍贵	不应使用

02

通用通信安全

2.1 特权功能接口

设备应默认关闭可直接进入设备系统的特权能力或接口（如工厂OTA、未公开功能接口、调试后门等），如实属业务必要，应具备鉴权机制。

2.2 密钥硬编码

设备不应将用于传输加密或鉴权的密钥硬编码在程序代码中，应采用一机一密（PSK）或通过 PSK 密钥导出等方式生成密钥。

2.3 通信信道加密

设备应将与其他设备或应用通信的信道进行加密，并在会话结束时及时销毁会话密钥。针对不同通信方式的加密方案可参考第三、四、六章。

2.4 通信鉴权

设备通信时应在数据传输之前进行双向认证，验证双方真实身份是否合法，检查控制权限是否与身份匹配，以防止越权或非授权控制。

2.5 防重放

设备通信应使用滚动码或计数器机制，当请求操作计数大于设备计数才准许设备执行该操作指令，以防止他人通过抓包重放控制请求来对设备进行非授权的控制。针对不同通信协议下的防重放方案，可参考下述通信协议安全章节的内容。

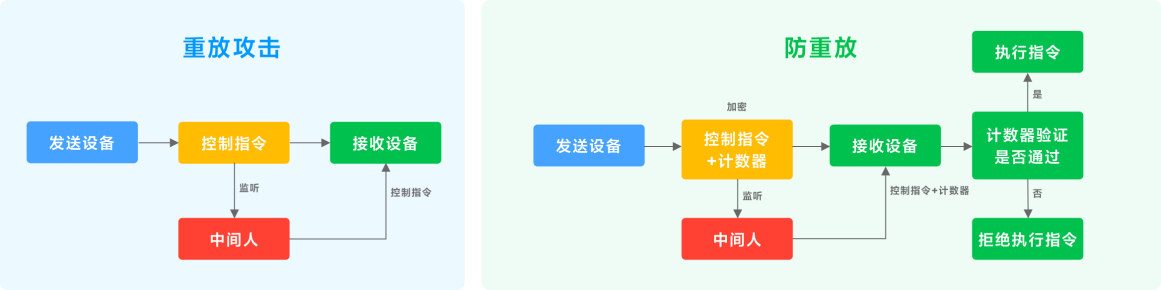

图 3 – 重放攻击和防重放

03

以太网
通信安全

本章（第三章）内容适用于使用 RJ45、WLAN 网络接入方式的设备

3.1 数据传输加密

设备应使用加密的传输协议对通信进行加密，在数据传输时须对敏感信息进行额外的加密（密钥算法及长度要求参见 13.1 加密算法）。设备宜采用 TLS（1.2+）安全传输协议。避免因使用 MQTT、HTTP 等明文传输协议而导致信息泄露或被篡改的风险。

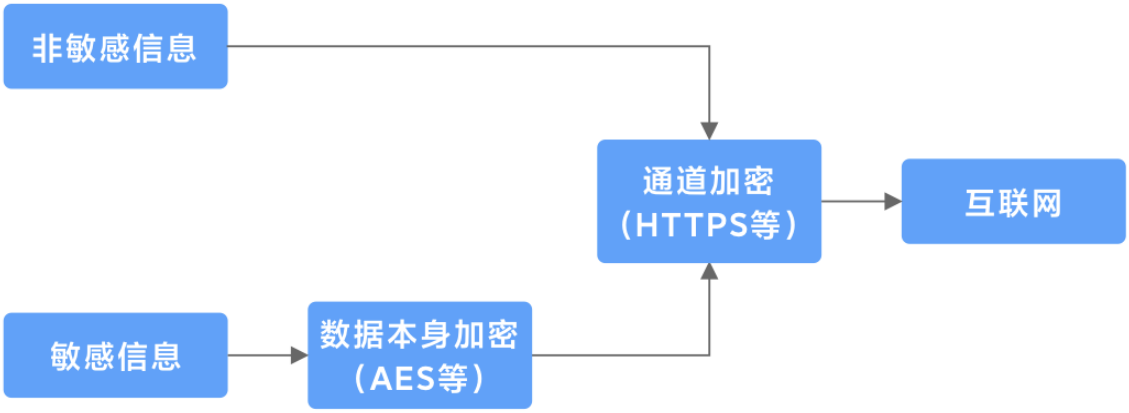

图 4 - 数据传输加密

3.2 HTTPS 证书校验

设备使用 HTTPS 协议时，应进行严格的证书校验，不能忽略检查。

- Linux 系统

设备应严格验证服务端证书合法性，不应使用参数跳过证书验证或忽略证书验证错误。

工具 / 库	参数使用建议
curl	不应使用 -k 参数
wget	不应使用 --no-check-certificate 参数
libcurl	应将 CURLOPT_SSL_VERIFYPEER 和 CURLOPT_SSL_VERIFYHOST 设置为 True

- Android 系统

设备应用使用 SSL 加密通信服务应严格校验服务端和客户端证书，不应信任任意证书，不应忽略异常事件（如 return 空或者 null）；如需自定义 SSLx509 TrustManager，重写 checkServerTrusted 方法，方法内必须严格判断服务端的证书校验，以防止通信内容被劫持导致通信数据泄漏或被篡改。

3.3 设备服务端口

设备应仅开启业务必要的 IoT SDK 控制服务端口进行数据交互与控制实现。

3.4 WiFi 接入点口令

设备使用 WiFi Direct 接入点与控制应用连接的功能应严格评估必要性，如确有必要，设备端 WiFi Direct 接入点口令不应使用固定密码或空密码，应遵循一机一密或者每次使用真随机生成密码，并显示在屏幕上（带屏设备）或在绑定时由用户提前设置并存储（无屏设备），以防止 Wi-Fi 被非法接入。

04

低功耗蓝牙(BLE)
通信安全

4.1 蓝牙 SoC SDK 版本

设备应定期检查并升级蓝牙 SoC SDK 至官方最新版本，不应使用存在安全问题的 SDK 版本（如协议栈漏洞 SweynTooth[1]）。

4.2 蓝牙配对

有物理按键的蓝牙设备应通过物理按键进行绑定确认或开启绑定窗口，以避免设备重复绑定或在用户不知情的情况下被他人绑定的风险。

4.3 控制指令合法性校验

对于支持蓝牙的设备，应每次登录协商会话密钥，设备与控制应用间应使用会话密钥加密传输控制指令。

4.4 传感器设备蓝牙广播

部分设备通过蓝牙将传感器采集的数据进行广播传输，通过蓝牙网关解析广播内容进行设备联动，或根据广播数据作为控制指令，从而影响其他设备。此类设备的蓝牙广播应使用安全的通信协议，并使用绑定时产生的 beaconkey 对广播数据内容进行加密。

4.5 蓝牙协议版本

低功耗蓝牙 BLE 设备（如鼠标、音箱）采用蓝牙链接层加密时应使用 4.2 及以上低功耗蓝牙协议版本，以防止设备在绑定阶段泄露蓝牙连接层密钥（LTK），从而导致隐私泄露或设备伪造的风险。

4.6 蓝牙控制指令鉴权

设备使用蓝牙芯片厂商提供的 OTA 例程指令前应判断控制应用的蓝牙绑定状态，防止因例程中可能支持未授权指令控制设备的逻辑，从而导致设备拒绝服务（例如芯片重启、切换工作空间、进入 DFU 升级模式等等）的风险。

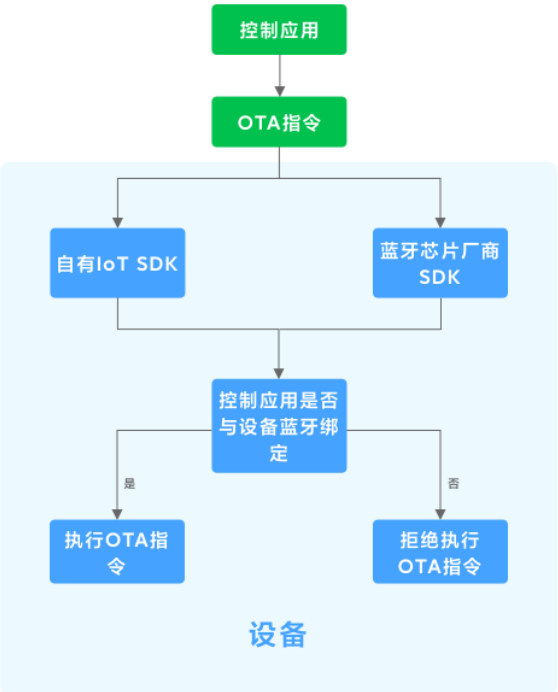

图 5 – 蓝牙控制指令鉴权

4.7 蓝牙广播防追踪机制

设备宜使用随机蓝牙 MAC 地址，并对蓝牙广播内容进行加密；如需通过 beacon 信标广播可识别身份的信息，应定时变换设备蓝牙 MAC 地址，以防止他人通过部署足够多的探测设备对广播内容进行分析，并跟踪设备的移动轨迹。

4.8 蓝牙敏感信息通信

设备在使用 BLE 与控制应用（安卓）通信时，应对敏感信息内容本身进行应用层加密，以防止用户使用 BLE 将手机与其他智能设备配对进行数据传输时，手机上的所有应用都可以访问这两个设备间传输的数据。

风险描述参见安卓 4.3 引入蓝牙 BLE 时的开发文档[2]：

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.

05

设备 Zigbee 通信安全

5.1 默认 TCLK

支持 Zigbee 1.2 协议栈的设备不应使用 Zigbee 联盟默认的 TCLK(Trust Center Link Key)：5A6967426565416C6C6961 6E63653039 (ZigBeeAlliance09)，出厂前应对协调器与节点预制非默认 TCLK。

支持 Zigbee 3.0 协议栈的设备应使用官方 install code 方案[3]。

5.2 防重放

设备应开启 Zigbee 协议帧计数器（framecounter），以具备防重放攻击保护能力。

```
开启方法：设置 nwkAllFresh 为TRUE
```

5.3 Rejoin 功能

设备使用 Rejoin 功能时应设置 apsUseInsecureJoin 为 FALSE（默认为 TRUE），以防止 Zigbee 在未满足 3.5.1 的情况下，即使未开启 join 窗口，依然可通过 rejoin 攻击[4]，利用默认 TCLK 解密 Network Key 控制该网络的风险。

06

设备射频通信安全

6.1 防重放

设备应使用滚动码方式进行通信，防止重放与伪造，可参考keeloq、DST40、Hitag2 等业界成熟方案[5]。

6.2 射频通信数据包序列号

设备的射频通信数据包应使用四字节作为序列号变量空间，避免使用较短序列号长度，以防止通信内容可在短时间暴力破解的风险。

6.3 通信密钥硬编码

设备射频通信密钥应通过发射器和接收器配对交换生成，不应预置密钥在代码中。

6.4 射频通信数据包序列号

设备宜使用跳频机制进行通信，以防止因射频通信频点固定造成信道拥堵，导致设备无法正常通信。

07

设备通用
系统安全

7.1 固件升级包完整性与合法性

设备固件升级（OTA 远程或本地升级）前应先对固件包进行完整性哈希得到固件包摘要，再使用公私钥方式对摘要进行合法性签名和验签（可参考 Nordic 例程[6]），确认升级包完整性与合法性再进行更新，以防止固件包被篡改或替换。

固件升级包完整性哈希宜采用安全的哈希算法（详见 13.2 哈希算法），完整性凭据应在设备与服务端的加密通信通道内传输。

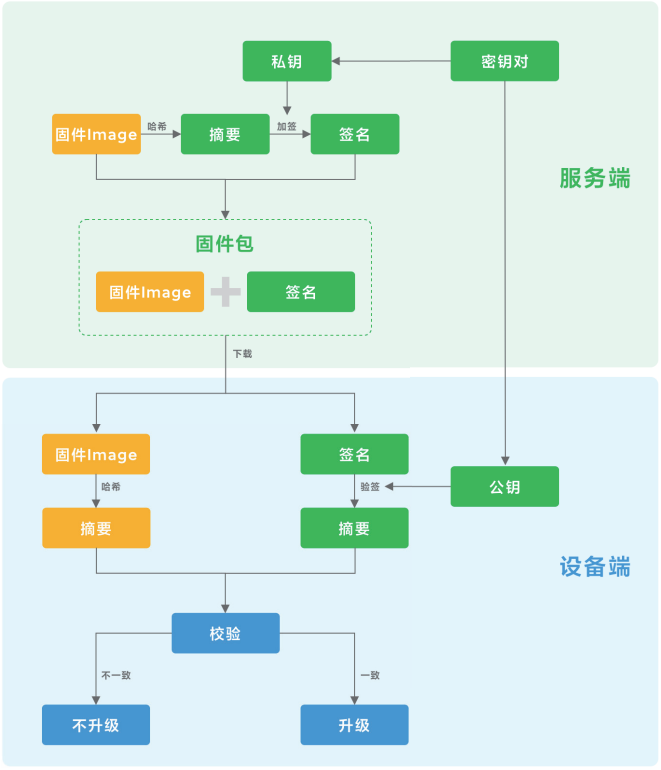

图 6 – 固件升级包合法性与完整性校验

7.2 固件降级

通过 OTA 功能进行设备更新时，设备拒绝旧版固件更新，以防止一些历史 BUG 或安全风险重新暴露被利用。特殊需求下（如：测试、维修等），可以通过 SD 卡或线刷等方式对固件验签后刷机。

7.3 高风险网络服务

设备应默认关闭 FTP、SSH、Telnet、HTTP、ADB 等高风险管理服务或信息数据服务。

7.4 OTA 升级指令

设备应仅执行云端授权来源的 OTA 升级指令，宜关闭局域网更新能力，以防止攻击者通过局域网对设备进行恶意固件 OTA 升级。

08

嵌入式设备 Linux 系统安全

8.1 地址空间布局随机化

嵌入式 Linux 系统的设备应开启地址空间布局随机化（ASLR）保护措施，以防止缓冲区溢出。

```
开启方式：在 /etc/sysctl.conf 中添加kernel.randomize_va_space = 2（Kernel >=2.6.12）
参数 2 代表除了库与栈外，堆也进行随机化保护，但需要注意，ASLR不负责代码端以及数据端的随机保护，
该项工作需要通过编译器PIE 实现，参见： 11.3
```

8.2 Bootloader 启动

设备 BootLoader 不应在系统启动前预留中断时间，Delay 需设置为 0，防止通过修改启动参数进入 SHELL 界面获得设备控制权。

8.3 串行端口

设备不应将系统 SHELL 绑定在 UART 等串行调试接口，防止通过接线获取设备控制权。如有特殊需要，需满足 8.4 要求。

8.4 系统默认用户密码

嵌入式 Linux 系统默认用户（如 root、admin 等）需设置高强度密码，并且保证一机一密，不应在代码中为所有设备写入相同密码或空密码。

```
高强度密码：密码应为10-14位，必须包含字母、大写字母、符号、数字四类字符
```

8.5 基础文件系统权限

设备嵌入式 Linux 的基础文件系统（如 /etc/ 等存有启动项的目录）宜使用只读文件系统（如 squashfs），以防止攻击者在篡改运行状态的操作系统。

8.6 外部存储的程序和脚本

设备嵌入式 Linux 操作系统默认不应运行外部存储（SD 卡、U 盘、网络存储等）中程序或脚本。如有特殊需要，应进行公私钥签名验证，以防止系统被植入恶意软件或脚本。

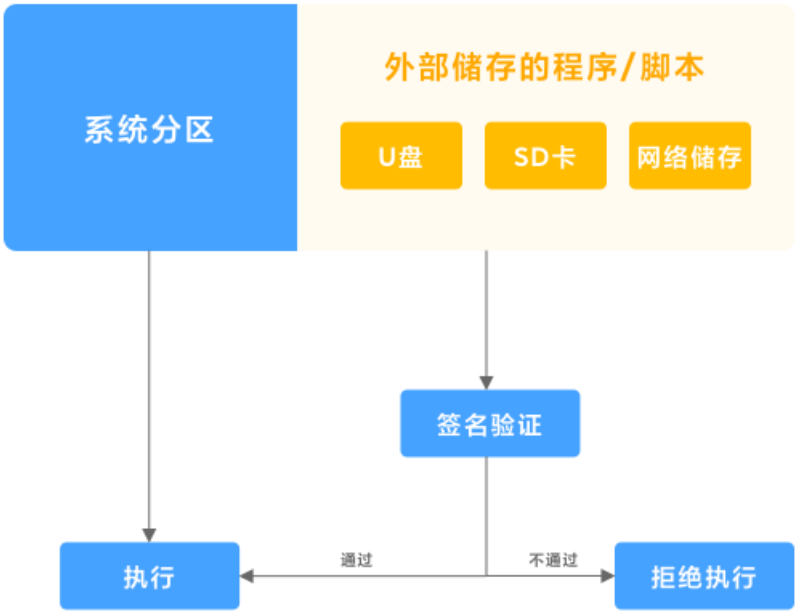

图 7 – 外部程序和脚本签名验证

09

Android
应用安全

9.1 Socket 端口请求

应用开放 Socket 网络端口和功能时，应对端口请求鉴权，并校验通信数据合法性，以防止其他本地应用或远程攻击者通过网络直接连接端口，恶意进行功能调用。

9.2 私有目录权限

应用私有目录下的文件和文件夹，other 用户不应有读写和执行权限（如：应设置为 "-rw-rw----"，不应设置为 "-rw-rw-rw-"），以防止相关权限引起的程序逻辑被篡改或用户隐私信息泄露。

9.3 本地信息存储

应用不应将敏感信息明文存储在设备本地文件，包括文本文件、二进制文件、SharedPreference 等 XML 文件、WebView 等数据库文件，以防止应用数据目录中的恶意文件读取并泄露存放在本地的敏感信息。

9.4 外部可执行文件

需要从外部动态加载的文件不应存储在 sd 卡这类任意进程可写的文件位置，应放在私有目录同时验签。如因特殊原因需要存储在 sd 卡等外部存储中,应用应在加载该文件前验签,以防止该外部可执行文件被篡改。

9.5 解压文件

应用使用 ZipEntry.getName() 解压 zip 压缩文件时，应对上级目录字符串（../）进行过滤校验，仅允许文件名不包含（../）特殊字符的压缩包进行解压，以防止解压目录跳转并覆盖其他目录关键文件，导致任意外部代码被执行。

应用应对重要的 Zip 压缩包文件进行数字签名校验，应仅允许校验通过的压缩包文件进行解压。

9.6 XML 配置

应用应关闭应用备份配置和可调试配置，以防止能接触用户设备的恶意攻击者短时间内启动设备 USB 调试功能，在不依赖设备 ROOT 权限的情况下通过 adb 调试工具窃取应用备份数据或篡改应用逻辑，造成用户隐私泄露甚至财产损失。

```
配置方法：
应用备份：AndroidManifest.xml文件中指定android:allowBackup="false"。
可调式：保持AndroidManifest.xml中debugable的默认配置：android:debugable="false"
```

9.7 防逆向工程

- **应用宜进行反编译保护**

应用启动时宜拒绝执行 su 命令或查找 su 文件，应用收到此类指令时宜抛出异常并停止运行，以防止应用在已 ROOT 的设备上运行被反编译。

- **应用应使用代码混淆**

应用应使用 Proguard 或其他加固方案对应用 APK 程序中 Java 代码进行混淆，以防止应用被反编译后核心代码被窃取。

9.8 应用完整性

应用宜对 dex 及 apk 包进行完整性校验，避免客户端文件被篡改导致盗版、内购破解、植入广告和恶意代码。

9.9 安装包签名

应用应使用从属方数字证书进行签名后再发布，不应使用第三方开发商的证书进行签名，以防止与第三方开发商合作终止需要切换为应用从属方签名时，新旧版本签名不一致，应用无法覆盖安装，导致应用从属方失去对版本管理的主动权。

9.10 安卓组件

- **安卓组件权限控制**

Android 应用四大组件可通过导出属性设置为导出或不导出，不导出组件仅允许应用内部访问，导出组件可被任意程序的任意组件访问。

- 应用四大组件均应主动明确地设置导出属性为导出或不导出，以更明确地表达该组件是否可被外部访问的意图。不应使用默认的导出规则，以防止在开发人员对 Android 不同版本还有不同组件类型默认导出规则不熟悉的情况下，原本不希望被外部访问的组件被意外导出并被外部恶意访问。

- 仅应用内部使用的组件应强制设置为不导出，以防止不必要的外部访问。

- 需要对相同签名的其他应用开放的组件，应仅把有业务需求的组件设置为导出，并使用自定义 permisssion，设置权限级别为 signature[7]，以防止组件接口被恶意使用。

设置方法：
在AndroidManifest.xml里通过导出属性"android:exported"设置为不导出（"false"）或导出（"true"）

- **Activity 防劫持**

应用宜采取以下两种保护措施以防止应用启动页面 Activity 被劫持为钓鱼工具：

- 界面切换

应用宜在 Activity 界面切换到后台时弹出警告信息，以防止劫持者将其页面附着在客户端之上。

- 进程栈保护

应用宜针对进程栈进行相应的保护，不应允许其他进程放置于客户端之上。

- **Content Provider 访问**

应用 ContentProvider 组件的文件访问接口被外部调用时，应对传入的目标文件 URL 参数对应的文件绝对路径进行过滤判断，仅执行允许访问路径的文件访问请求，以防止攻击者通过传入包含父级目录的 URL 未经授权读取任意目录可读文件，造成数据泄露。

过滤方法：
获取最终要访问文件的绝对路径(通过File.getCanonicalPath)方法，判断该路径是否是允许访问的目录路径开头的，如果不是，那么访问的文件不合法，不提供后续的业务功能。

- **Intent**

- 在使用 PendingIntent 时不应使用空 Intent 或隐式 Intent

应用不应使用空 Intent 构造 PendingIntent，并且构造 PendingIntent 的 Intent 一定要设置 ComponentName 或者 action，以防止 Intent 被劫持导致信息泄露。

- 应用接收 Intent 传入的 URL 时应对 Intent 对象及 URL 做严格过滤

应用接收 Intent 传入的 URL 应做严格过滤，并对 Intent 对象设置相应过滤规则，以避免应用对 Intent Scheme URL 处理不当导致的基于 Intent 的攻击，同时也能避免外部传入的 Intent 直接启动系统组件，导致攻击者可以绕过部分签名直接启动组件或启动未导出的组件。

● WebView 证书认证

当应用 WebView 组件加载网页发生证书认证错误时，应停止加载问题页面，不应忽略证书错误[8]，以防止受到中间人攻击导致隐私泄露。

```
方法：如果重载了onReceivedSslError方法，则应在onReceivedSslError中使用
handler.cancel()方法停止加载问题页面
注释：不在重载的onReceivedSslError方法中调用handler.proceed()忽略证书
校验，继续加载网页
```

● WebView 跨域访问

应用应关闭 WebViewfile 跨域访问本地文件，以防止应用克隆攻击或本地数据远程泄漏。

```
关闭方法：
setAllowFileAccess(false)，setAllowFileAccessFromFileURLs(false)，
setAllowUniversalAccessFromFileURLs(false)，并对Intent传入的最终
访问的文件URL进行过滤判断（详见9.10 Content Provider访问 ）
```

- **WebView jsbridge 调用**

应用应严格限定 schema 为 https，并使用白名单限制 Webview 可访问的域名，同时对最终访问文件的绝对路径进行过滤判断（详见 9.10 ContentProvider 访问），以防止 WebView 的 JSAPI 不加鉴别地允许任意网页执行，导致用户隐私泄露或远程代码执行攻击。

- **WebView 密码存储**

应用在使用 WebView 的过程中应禁止保存密码，以防止用户在 WebView 中输入的用户名和密码被明文保存到应用数据目录的 databases/webview.db 中，以防止用户个人敏感信息泄露。

```
方法：使用WebView.setSavePassword(false)方法
```

9.11 防屏幕录像

应用宜使用隐藏字符遮挡用户隐私信息，或监控进程列表判断 screencap（屏幕录像）进程是否运行，如果在运行则提示用户有风险，以防止恶意程序窃听用户输入的敏感信息。

9.12 内存数据保护

应用宜对内存数据进行加密处理，或通过检测 root 权限限制攻击者进行注入、hook、反调试， 以防止攻击者访问应用内存空间导致内存中的帐号、密码等敏感信息泄露。

10

通用编码安全

10.1 第三方软件 / 库

设备使用到的第三方软件 / 库应使用最新版本，如特殊需求不能使用最新版本，应经过安全评估。在明确安全漏洞的情况下，应及时更新到最新修复版本。

注：软件 / 库漏洞可前往：https://www.cvedetails.com/product-list.php 查询。

10.2 随机数生成函数

设备系统及应用应使用真随机算法产生认证或加密所需的随机数。

类型	生成方式	建议
真 / 强伪随机	/dev/urandom 芯片支持的熵随机能力	推荐
伪随机	srand()、rand() 并使用 time(0) 为种子生成	不推荐

10.3 字符串或内存操作函数

系统及应用应使用安全的字符串或内存操作函数，以防止使用不执行边界检查的字符串或内存函数被攻击者用来进行缓冲区溢出攻击。

```
安全建议：
不使用strcpy，使用strncpy；
不使用strcat，使用strncat；
不使用vsprint，使用vsnprintf；
不使用sprintf/vsprintf，使用snprintf
```

注：使用安全的函数，用法不安全也可能导致安全问题

如 strncpy(dest, src, size)，每次拷贝前，需要根据 dest 的缓冲区的大小指定 size，size 必须小于 dest 的大小，一定不要使用 "dest 固定长度且 sizelen(src)" 的做法，这样的话 dest 获得的数据长度等同于 src 的长度，此时 strncpy 函数等同于 strcpy 函数。

代码示例：

```
【错误代码示例】
dest = malloc(20);
src = *param1;
size = len(src);
strncpy(dest, src, size);
```

上面的代码示例中，dest 申请 20 的大小，src 来自外部传入的参数，也就是说对攻击者可控，size 等于 src 的长度，所以此时 strncpy 等价于 strcpy

```
【正确代码示例1】
dest = malloc(50);
strncpy(dest, src, 40)  // 此时dest的缓冲区大小一定大于size的大小
【正确代码示例2】
size = len(src);
dest = malloc(size + 1);
strncpy(dest, src, size);  // 这样能保证dest的容量一定大于src
```

10.4 格式化字符串函数参数

系统及应用使用格式化字符串函数不应使用外部可控变量作为参数，以防止格式化字符串漏洞[9]造成严重危害。

```
举例：
sprintf(char *string, char *format [,argument,...])中，format参数不允许使用外部可控变量。
```

10.5 代码库管理

设备相关代码库不应在未经允许的情况下上传至 Github、Gitee 等公用代码仓库或百度网盘等公开、半公开服务，防止源代码泄露。

11

Linux 应用编码安全

11.1 栈 Cookie 防溢出

在编译 Linux 程序代码时，应开启 Linux 程序的 CANARY 栈溢出保护选项。

```
开启方法：添加gcc编译参数 -fstack-protector-all
```

11.2 栈不可执行保护

在编译 Linux 程序代码时，应开启 Linux 程序的 NX 栈不可执行保护选项。

```
开启方法：添加gcc 编译参数 -z noexecstack
```

11.3 基址随机加载保护

应用应开启 PIE 应用基址随机加载保护选项。

```
方法：gcc 编译参数-pie -fPIE（注意大小写），同时需要系统 ASLR 支持，参见：8.1
```

11.4 Linux 程序代码编译

在编译 Linux 程序代码时，需使用 Strip 函数删除调试符号表，以提升逆向分析难度并减少程序体积。

11.5 系统调用函数参数

编码中使用系统调用函数时（如 system、popen、exec 等）应对外部可控参数进行过滤，防止命令注入。

```
方法：
对系统调用函数参数进行严格的字符过滤，如$/()/`/;/|/&/<>/" 等具有shell特殊含义的字符；
将参数限制在一对单引号中，然后过滤或转义后续传入的所有单引号；
采用 execl 类函数对指定进程进行参数化执行，execl 第一个参数不建议为 shell 程序
```

12

业务逻辑安全

12.1 设备可绑定状态

未绑定设备上电 30 分钟后仍未绑定应关闭待绑定状态，待重新上电后重新进入绑定状态，避免设备始终允许接受绑定请求而被恶意绑定的风险。

12.2 绑定确认

设备绑定时宜在产品能力允许的情况下，要求用户在设备端进行绑定确认（OOB），交通工具和高安全级别设备（如门铃、门锁、户外监控）应强制要求 OOB，以防止设备被恶意绑定。

12.3 防重复绑定

对于具备重置能力的设备，应被重置后才可再次接受绑定请求，以防止被其他用户重复绑定或恶意绑定控制。

12.4 强绑定关系

室外设备、高安全级别设备应在云端将设备和账户建立强绑定关系，即云端记录设备 ID 与用户 ID 的绑定关系。设备绑定时应先验证云端记录，仅允许在云端没有绑定记录的设备的绑定请求，并且设备端重置不应清除云端绑定记录，仅当设备所有者（在云端有绑定记录的用户 ID）在控制端应用上主动解绑时才清除绑定关系，以防止设备重置后被非法绑定控制。

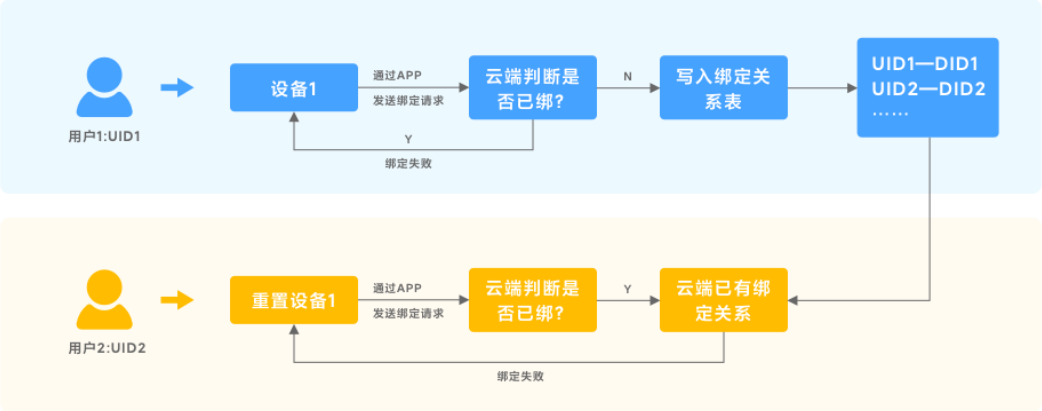

图 8 - 强绑定

12.5 恢复出厂设置

设备恢复出厂设置后应完全清除设备中所有用户数据、设置（如用户使用记录、设置、NFC 门卡、eSIM 卡记录等）。

12.6 Wi-Fi 接入点用途

设备建立的 Wi-Fi 接入点应只允许与设备本身通信，不应访问设备外部网络（如家庭网或互联网），以防止攻击者通过设备 Wi-Fi 向家庭网渗透。

12.7 身份认证逻辑

设备进行身份认证的代码逻辑应进行正确的判断，应验证凭证完全正确才能通过认证，以防止通过空认证凭据跳过认证判断，直接进入控制逻辑。

```
逻辑漏洞示例：
将认证检测逻辑放在认证凭证是否为空的判断内，且没有实现认证凭据为空的异常处理，如：
if ( token !== ''){
    if ( checkauth( token ) == FALSE ) {
        exit("Auth fail");
    }
}
```

13

数据安全

13.1 加密算法

设备应使用安全的加密算法且密钥长度符合对应算法的最低要求（见下表），不应使用 DES、TDES、RC4 等不安全加密算法。

算法	密钥长度（bit 位）
AES	128 / 192 / 256
ECDSA / ECDH	256/384/512
RSA	2048 /3072/ 4096
DSA	Prime P: 2048 /3072/4096
	Prime Q: 256/384/512
SHA	256/384/512
SM4	128

注：

1. 使用 RSA 算法加密时不使用 NoPadding

2.IvParameterSpec 初始化时，不使用常量 vector

3. 在选择加密模式时避免使用 ECB 模式

13.2 哈希算法

设备宜使用 256 bit 以上的安全 hash 函数，如 sha256 ，sha512 等，不宜使用已有安全风险的 hash 函数，如 md5 或 sha1。

13.3 多重密钥

部分设备具有多重密钥协商逻辑，即设备通过内置信息导出交换密钥，用于在设备绑定时与服务端加密通信获取控制指令密钥，后续的设备控制指令使用获取到的控制指令密钥加密。此交换密钥应仅用于绑定过程中获取控制指令密钥，不应用于加密后续控制指令。

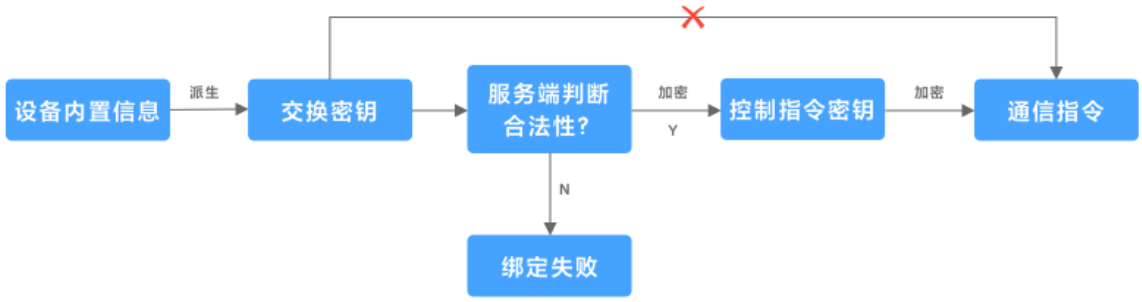

图 9 – 多重密钥

13.4 日志上报

设备、系统及应用打印的日志不应上传任何明文或加密的敏感信息（如 Wi-Fi SSID、密码、手机 IMEI、地理位置等敏感信息）。如经评估需要上报使用，应在隐私声明中明示作用与存储方式。

13.5 跨境网络请求

设备或控制应用应识别当前用户所属国家或地区，根据不同国家或地区的隐私合规要求发送网络请求，以防止数据跨境传输带来的隐私风险。

术语和定义

下列术语和定义适用于本文件

1 消费级物联网设备 Consumer IoT Device

网络连接（和网络连接）设备（本文简称“设备”），该规范内消费级物联网终端设备指具有无线（Wi-Fi、BLE/Mesh、Bluetooth classic、Zigbee、NFC、RF 射频）、有线（RJ45）联网与组网能力，或具有固件逻辑更新能力的任意智能终端产品。

注 1：消费级物联网设备也通常用于商业环境。这些设备仍然被归类为消费级物联网设备。

注 2：消费级物联网设备通常可供消费者在零售环境中购买。消费级物联网设备也可以委托和 / 或专业安装。

2 用户 User

自然人或组织。

3 关键安全参数 critical security parameter

与安全相关的秘密信息，这些信息被泄露或被修改后会危及智能家居安全性。例如后台系统管理员认证信息、操作系统登录认证信息，网络设备认证信息等。

4 公共安全参数 public security parameter

与安全相关的公共信息，其修改会损害安全模块的安全性。

示例 1：验证软件更新的真实性 / 完整性的公钥。

示例 2：证书的公共组件。

5 敏感安全参数 sensitive security parameters

公共安全参数和关键安全参数。

6 个人信息 personal data

与已识别或可识别的自然人有关的任何资料。

7 敏感信息 Sensitive information

敏感信息为敏感安全参数与个人信息的统称。

8 调试接口 debug interface

制造商在开发过程中用于与设备通信的物理接口，或用于对设备的问题进行分类 . 例如：测试点，UART，SWD，JTAG。

9 逻辑接口 logical interface

利用网络接口通过信道或端口在网络上通信的软件实现。

10 网络接口 network interface

可用于通过网络访问消费者物联网功能的物理接口。

11 物理接口 physical interface

物理端口或空气接口（如无线电、音频或光学接口），用于与物理层的设备通信，如：收音机、以太网端口、USB 等串行接口和用于调试的接口。

12 安全更新 security update

解决制造商发现或报告给制造商的安全漏洞的软件更新。

注：如果漏洞的严重程度需要更高的优先级修复，软件更新可以是纯粹的安全更新。

13 日志 telemetry

来自设备的数据，可以提供信息，帮助制造商识别与设备使用有关的问题或信息。例如：消费者物联网设备向制造商报告软件故障，使他们能够识别和补救原因。

14 鉴权 authentication mechanism

用于证明实体真实性的方法：“实体”既可以是用户，也可以是机器。例如：认证机制可以是请求密码、扫描二维码或使用生物特征指纹扫描仪。

15 闪存 Flash

Flash 是存储芯片的一种，通过特定的程序可以修改里面的数据。FLASH 在电子以及半导体领域内往往表示 Flash Memory 的意思，即平时所说的“闪存”，全名叫 Flash EEPROM Memory。

16 Bootloader

在嵌入式操作系统中，BootLoader 是在操作系统内核运行之前运行，可以初始化硬件设备、建立内存空间映射图，从而将系统的软硬件环境带到一个合适状态，以便为最终调用操作系统内核准备好正确的环境。

17 U-boot

U-Boot 是一个主要用于嵌入式系统的引导加载程序，可以支持多种不同的计算机系统结构，包括 PPC、ARM、AVR32、MIPS、x86、68k、Nios 与 MicroBlaze。

18 Kernel

嵌入式内核是在嵌入式硬件和软件之间的抽象层，在 Linux 的术语中被称为“内核”，也可以称为“核心”。Linux 内核的主要模块（或组件）分以下几个部分：存储管理、CPU 和进程管理、文件系统、设备管理和驱动、网络通信，以及系统的初始化（引导）、系统调用等。

19 蓝牙信标 Beacon

蓝牙信标（Beacon）是建立在低功耗蓝牙协议基础上的一种广播协议。

20 地址随机化 ASLR

地址随机化（ASLR）是一种针对缓冲区溢出的安全保护技术，通过对堆、栈、共享库映射等线性区布局的随机化，通过增加攻击者预测目的地址的难度，以防止攻击者直接定位攻击代码位置，达到阻止溢出攻击的目的。

21 ID 卡 Identification Card

ID 卡全称为身份识别卡（Identification Card），是一种不可写入的感应卡，含固定的编号，主要有台湾 SYRIS 的 EM 格式、美国 HIDMOTOROLA 等各类 ID 卡。ID 卡与磁卡一样，都仅仅使用了“卡的号码”而已，卡内除了卡号外，无任何保密功能，其“卡号”是公开、裸露的。所以说 ID 卡就是“感应式磁卡”。

22 M1 卡 M1 card

M1 芯片卡，是指飞利浦下属子公司恩智浦出品的芯片缩写，全称为 NXP Mifare1 系列，常用的有 S50 及 S70 两种型号。常见的有卡式和钥匙扣式。

23 真插芯

智能门锁的真插芯是指锁芯直接固定在再锁体结构上，利用锁芯上的拨片直接驱动锁体，使其实现开关效果的一种组合方式，直观表现为：使用钥匙时，钥匙与门面为垂直关系。

24　假插芯

智能门锁的假插芯是将锁芯与锁体位置分离，锁芯转动时，利用齿轮箱和连接杆，将动力传送给锁体，从而实现锁体开关效果的一种新型组合方式，直观表现为：使用钥匙时，钥匙与门面为平行关系。

25　例程 Routine

例程是某个系统对外提供的功能接口或服务的集合。比如操作系统的 API、服务等就是例程。

缩略语

下列缩略语适用于本文件。

API	应用编程接口（Application Programming Interface）
IoT	物联网（Internet of Things）
IP	网络协议（Internet Protocol）
JTAG	联合测试工作组（Joint Test Action Group）
OTA	空中下载（Over The Air）
UART	通用异步收发传输器（Universal Asynchronous Receiver-Transmitter）
USB	通用串行总线（Universal Serial Bus）
SSH	安全外壳协议（Secure Shell）
MAC	介质访问控制地址（Media Access Control）
TLS	安全传输层协议（Transport Layer Security）
TCP	传输控制协议（Transmission Control Protocol）
UDP	用户数据报协议（User Datagram Protocol）
FTP	文件传输协议（File Transfer Protocol）
UID	用户名（User ID）
DID	设备标识码（Device ID）
SWD	串行线调试（Serial Wire Debug）
NAND	与非门逻辑电路（Not And）
EMMC	Embedded Multi Media Card
MQTT	消息队列遥测传输（Message Queuing Telemetry Transport）
TCLK	信任中心链接密钥（Trust Center Link Key）
ASLR	地址空间布局随机化（ASLR）
NX Stack	栈不可执行（No Execute Stack）
PIE	位置独立的可执行文件（Position Independent Executables）
OOB	带外数据（Out of Band）
XML	可扩展标记语言（eXtensible Markup Language）
AES	高级加密标准（Advanced Encryption Standard）
DES	数据加密标准（Data Encryption Standard）
RSA	一种非对称加密算法，以此算法的三位共同提出者姓氏开头字母拼在一起而命名
TX	发送（transmit）
RX	接收（Receive）
PSK	预共享密钥（Pre-Shared Key）
ADB	安卓调试桥（Android Debug Bridge）
JSAPI	JavaScript 应用编程接口（JavaScript Application Programming Interface）

附录

个人信息

个人信息是指以电子或其他方式记录的能够单独或者与其他信息结合识别特定自然人身份或者反映自然人活动情况的各种信息。

表A.1　个人信息举例

个人基本资料	个人姓名、生日、性别、民族、国籍、家庭关系、住址、个人电话号码、电子邮件地址等
个人身份信息	身份证、军官证、护照、驾驶证、工作证、出入证、社保卡、居住证等
个人生物识别信息	个人基因、指纹、声纹、掌纹、耳廓、虹膜、面部识别特征等
网络身份标识信息	个人信息主体账号、IP 地址、个人数字证书等
个人健康生理信息	个人因生病医治等产生的相关记录，如病症、住院志、医嘱单、检验报告、手术及麻醉记录、护理记录、用药记录、药物食物过敏信息、生育信息、以往病史、诊治情况、家族病史、现病史、传染病史等，以及与个人身体健康状况相关的信息，如体重、身高、肺活量等
个人教育工作信息	个人职业、职位、工作单位、学历、学位、教育经历、工作经历、培训记录、成绩单等
个人财产信息	银行账户、鉴别信息(口令)、存款信息（包括资金数量、支付收款记录等）、房产信息、信贷记录、征信信息、交易和消费记录、流水记录等，以及虚拟货币、虚拟交易、游戏类兑换码等虚拟财产信息
个人通信信息	通信记录和内容、短信、彩信、电子邮件，以及描述个人通信的数据（通常称为元数据）等
联系人信息	通讯录、好友列表、群列表、电子邮件地址列表等
个人上网记录	指通过日志储存的个人信息主体操作记录，包括网站浏览记录、软件使用记录、点击记录、收藏列表等
个人常用设备信息	指包括硬件序列号、设备 MAC 地址、软件列表、唯一设备识别码（如 IMEI/Android ID/IDFA/OpenUDID/GUID/SIM 卡 IMSI 信息等）等在内的描述个人常用设备基本情况的信息
个人位置信息	包括行踪轨迹、精准定位信息、住宿信息、经纬度等
其他信息	婚史、宗教信仰、性取向、未公开的违法犯罪记录等

[来源：GB/T35273-2020《个人信息安全规范》附录 A]

个人敏感信息

个人敏感信息是指一旦泄露、非法提供或滥用可能危害人身和财产安全，极易导致个人名誉、身心健康受到损害或歧视性待遇等的个人信息。

表B.1　个人敏感信息举例

个人财产信息	银行账户、鉴别信息（口令）、存款信息（包括资金数量、支付收款记录等）、房产信息、信贷记录、征信信息、交易和消费记录、流水记录等，以及虚拟货币、虚拟交易、游戏类兑换码等虚拟财产信息
个人健康生理信息	个人因生病医治等产生的相关记录，如病症、住院志、医嘱单、检验报告、手术及麻醉记录、护理记录、用药记录、药物食物过敏信息、生育信息、以往病史、诊治情况、家族病史、现病史、传染病史等
个人生物识别信息	个人基因、指纹、声纹、掌纹、耳廓、虹膜、面部识别特征等
个人身份信息	身份证、军官证、护照、驾驶证、工作证、社保卡、居住证等
其他信息	性取向、婚史、宗教信仰、未公开的违法犯罪记录、通信记录和内容、通讯录、好友列表、群组列表、行踪轨迹、网页浏览记录、住宿信息、精准定位信息等

[来源：GB/T35273-2020《个人信息安全规范》附录B]

参考资料

【1】 SweynTooth 漏洞：

- Nordic 芯片作为 GATT 客户端受影响的协议栈版本为 S110/S120/S130/S132 v2.0.0

https://infocenter.nordicsemi.com/pdf/in_119_v1.0.pdf?cp=3_1_3_1

- Dialog，Telink（链接层加密），NXP，Cypress，Microchip，Texas Instruments，STMicroelectronics 具体型号版本参考：

https://asset-group.github.io/disclosures/sweyntooth/

SweynTooth 影响产品版本列表

SoC 供应商	Soc 型号	SDK（<= 存在漏洞）
BLE Version 5.0/5.1		
Cypress (PSoC 6)	CYBLE-416045	2.1
Texas Instruments	CC2640R2	3.30.00.20
Telink	TLSR8258	3.4.0
STMicroelectronics	WB55	1.3.0
STMicroelectroncis	BlueNRG-2	3.1.0
Dialog	DA1469X*	10.0.6
Dialog	DA14585/6*	6.0.12.1020
BLE Version 4.2		
Cypress (PSoC 4)	CYBL11573	3.6
NXP	KW41Z	2.2.1
Dialog	DA14680	1.0.14.X
BLE Version 4.1		
Texas Instruments	CC2540	1.5.0
Dialog	DA14580	5.0.4
Microchip	ATSAMB11	6.2

- SweynTooth 漏洞相关资料：

https://asset-group.github.io/disclosures/sweyntooth/#commento-login-box-container

【2】 安卓 4.3 引入 BLE 时的开发者文档：

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

【3】 Zigbee 3.0 官方 Install code 方案：

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

【4】 Rejoin 攻击：

https://research.kudelskisecurity.com/2017/11/21/zigbee-security-basics-part-3/

https://www.nxp.com/docs/en/supporting-information/MAXSECZBNETART.pdf

Re-using link key: ZigBee allows link keys to be re-used for rejoining the network. Hence, it would make it possible for an attacker to copy a device's addressing credentials and spoof a network layer insecure rejoin using a separate device. This would result in the Trust Center passing the network key encrypted with the previously used link key to the cloned device. As a consequence, an attacker could gain complete access to the network key and hence the entire network.

【5】 滚动码

滚动码被用于设备通信身份验证，防止重放攻击。

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

- 业界成熟方案：

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

【6】 Nordic OTA 例程

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

【7】 安卓 permisssion 级别

https://developer.android.com/guide/topics/permissions/overview?hl=en-us

【8】 Android Webview onReceivedSslError 方法：

https://developer.android.com/reference/android/webkit/WebViewClient#onReceivedSslError(android.webkit.WebView, %20android.webkit.SslErrorHandler, %20android.net.http.SslError)

【9】 格式化字符串漏洞

以 snprintf 为例，它会将可变个参数 (...) 按照 format 格式化成字符串，然后将其复制到 str 中。

- 如果格式化后的字符串长度 < size，则将此字符串全部复制到 str 中，并给其后添加一个字符串结束符 ('\0')；
- 如果格式化后的字符串长度 >= size，则只将其中的 (size-1) 个字符复制到 str 中，并给其后添加一个字符串结束符 ('\0')，返回值为欲写入的字符串长度。

当字符串操作函数接收的字符串参数长度过大时会造成整数溢出和栈溢出，如果攻击者在字符串中嵌套了恶意代码，可能导致信息泄露和数据被篡改等问题。因此对于使用 snprintf 等函数返回值做大小判断的场景，需要严格校验其值是否超过范围。