IOT Security and MiCO System

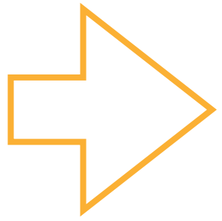
MiCO系统针对IOT领域的安全风险的应对方案

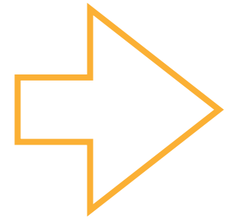
本演示文档所引用图片仅用于表达技术问题，并非代表其本身

Agenda

- IOT领域的安全现状和需求
- MXCHIP的物联网模组和MiCO系统
- 针对IOT安全的软硬件设计
  - 应用：固件验证和加密（保护设备厂商）
  - 应用：物联网设备的身份认证（保护云服务厂商）
  - 应用：端到端的信息安全传输（保护用户隐私）

IOT领域的安全现状和需求





- 非法复制，拷贝
- 破坏产品功能
- 给黑客以可乘之机
- 低成本，低销量

- 未授权的设备访问
- 无法形成各方认可收费体系
- 云端数据泄漏
- 无法建立良好生态和口碑

- 用户隐私泄漏
- 威胁用户安全
- 降低对物联网的信任

IOT领域的安全现状和需求

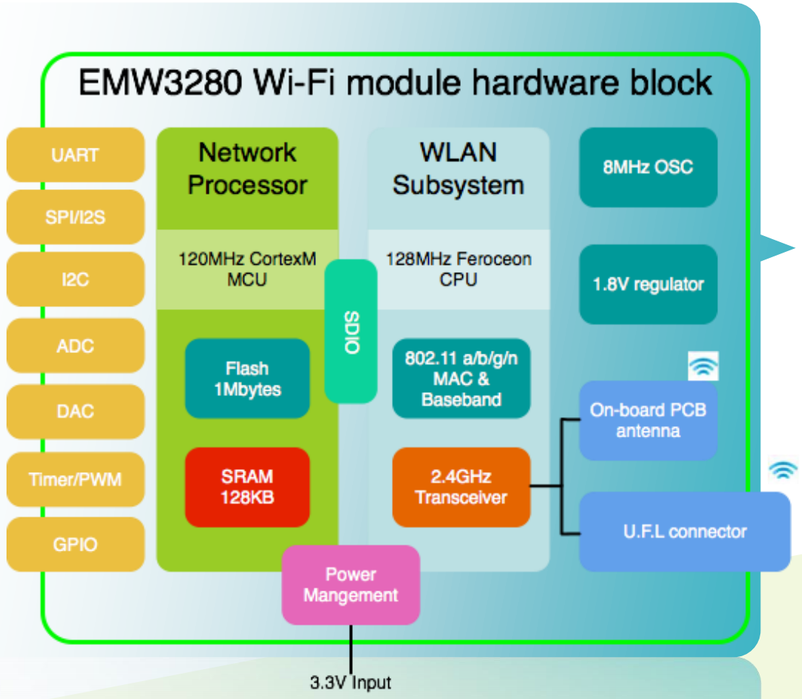
良好的物联网生态圈

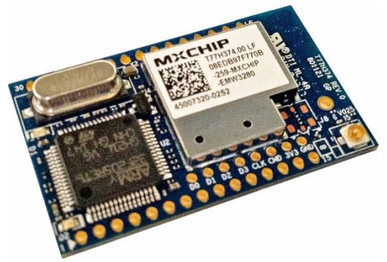
需要安全可信的物联网终端设备

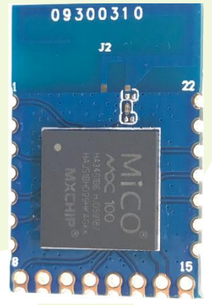
MXCHIP®

为嵌入式设备接入物联网提供安全的软硬件解决方案

MXCHIP的物联网模组







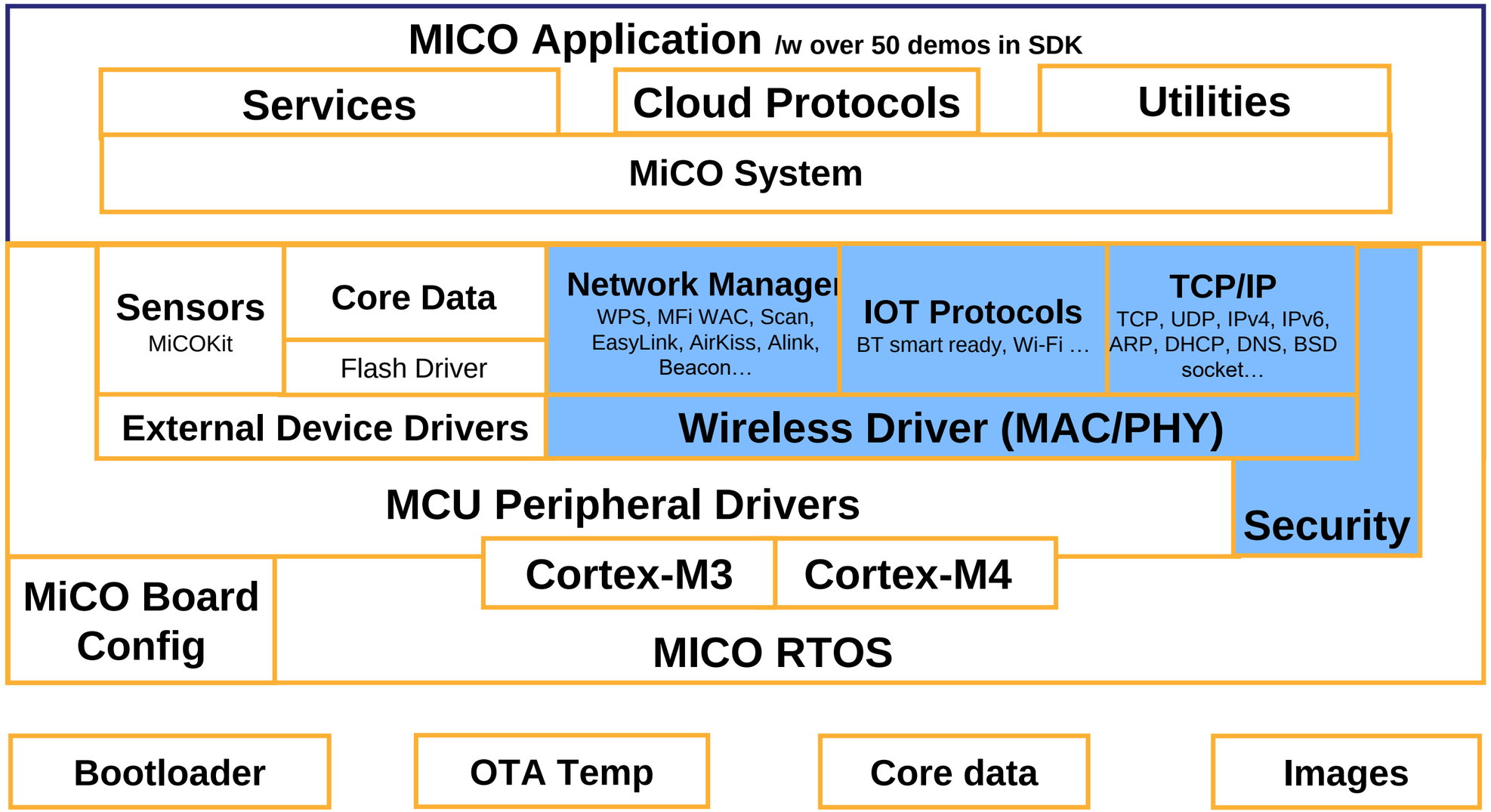
What is MICO™

Micro-controller based Internet Connectivity Operation system

基于微控制器的互联网接入操作系统

操作系统：让用户更容易地操控计算机硬件

MiCO SDK v3.0


MICO Application /w over 50 demos in SDK
Services
Cloud Protocols
Utilities
MiCO System
Sensors
MiCOKit
Core Data
Flash Driver
Network Manager
WPS, MFi WAC, Scan, EasyLink, AirKiss, Alink, Beacon…
IOT Protocols
BT smart ready, Wi-Fi …
TCP/IP
TCP, UDP, IPv4, IPv6, ARP, DHCP, DNS, BSD socket…
External Device Drivers
Wireless Driver (MAC/PHY)
MCU Peripheral Drivers
Security
Cortex-M3
Cortex-M4
MiCO Board Config
MICO RTOS
Bootloader
OTA Temp
Core data
Images

MiCoder开发工具

MiCoder ToolChain
组件化开发方式

MiCoder IDE
集成开发环境

源代码 → 用于生产的安全固件

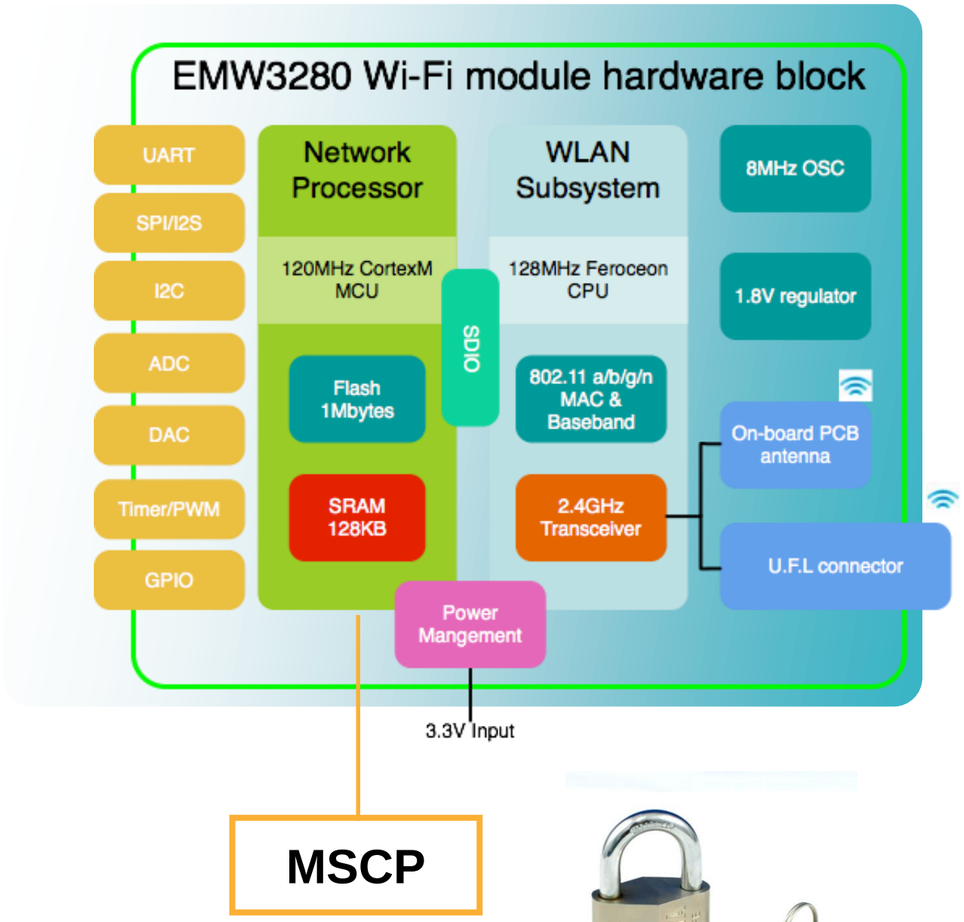
嵌入式设备安全的标准化开发方法

在实践中不断完善

上海庆科信息技术有限公司

MXCHIP
INSPIRE YOUR IDEAS

MXCHIP Security Coprocessor



MSCP的主要功能
存储+签名+密钥协商

- 私钥安全存储（不可读出）
- 安全数据，密钥的存储（可以被安全地读出）
- 自动生成公私钥对
- SHA256散列算法，HKDF密钥衍生算法
- EdDSA：椭圆曲线签名算法（如ED25519）

Security Stack on MiCO

散列算法

- SHA1，SHA224，SHA256，SHA384， SHA512
- MD5

密钥衍生算法KDF

- HKDF

公钥加密

- RSA，ECC

强密钥交互协议

- SRP6a

数据加密算法

- AES，DES，chacha20

消息验证码函数

- HMAC, Poly1305

密钥交换

- RSA，DH，ECDH(curve25519…)

数字签名算法

- RSA，DSA，ECDSA ，EdDSA（ED25519）

随机数生成器

Security Stack on MiCO

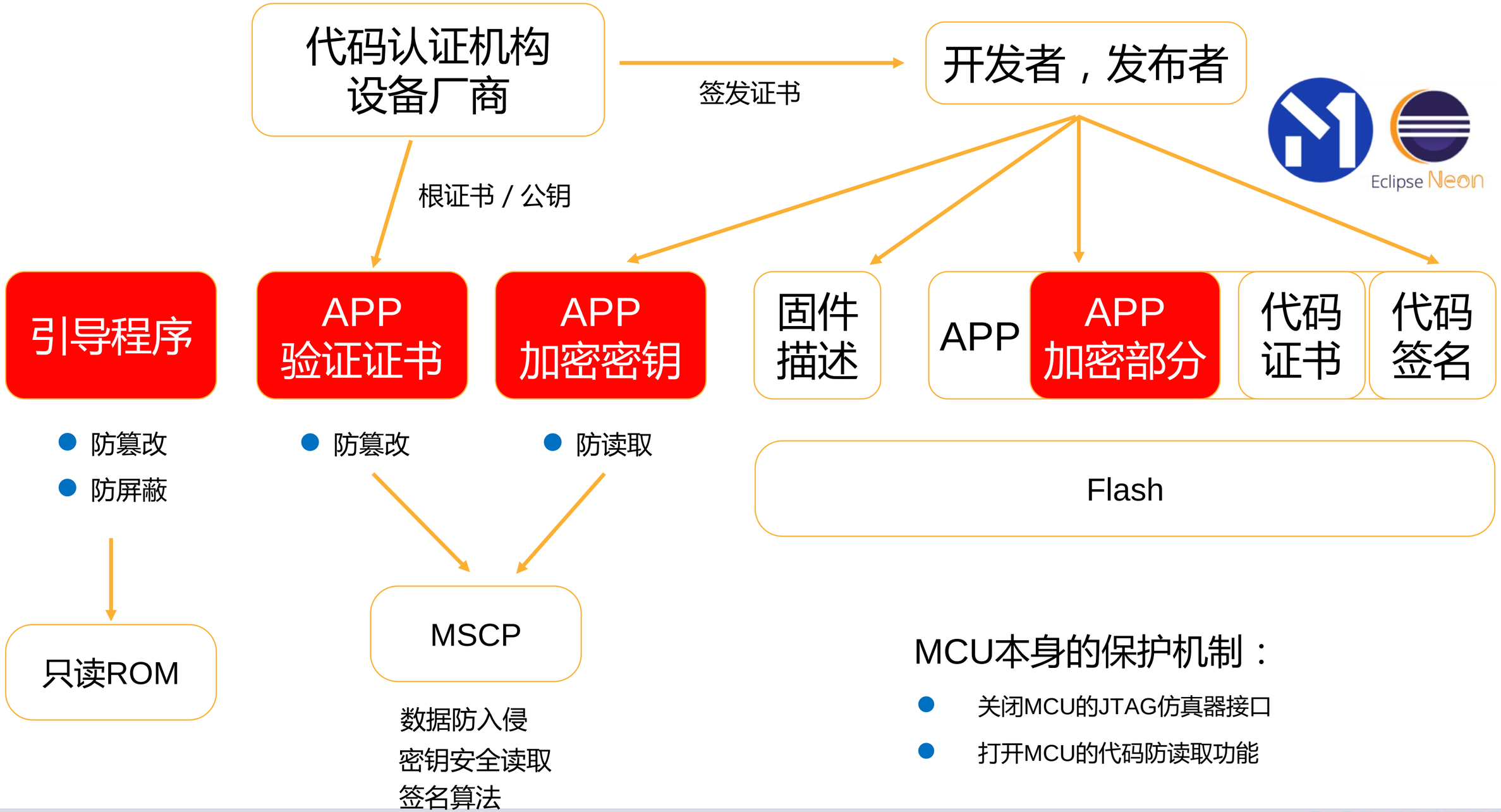
专为嵌入式系统设计的商用TLS库

- 已经广泛的部署和验证
- 支持TLS 1.2 和DTLS 1.2
- 支持MiCO RTOS接口环境
- 代码大小仅为OpenSSL的5%
- 已付费的商业授权，用户无二次费用

FIPS 140-2认证

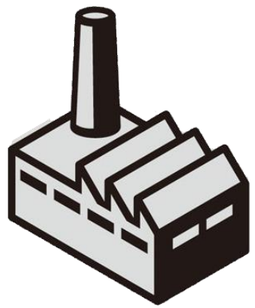
Federal Information Processing Standards

应用1：固件验证和加密

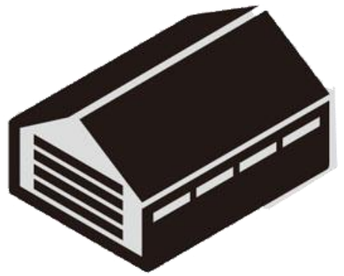

代码认证机构
设备厂商
签发证书
开发者，发布者
根证书 / 公钥
Eclipse Neon
引导程序
APP
验证证书
APP
加密密钥
固件
描述
APP
APP
加密部分
代码
证书
代码
签名
防篡改
防屏蔽
防篡改
防读取
只读ROM
MSCP
数据防入侵
密钥安全读取
签名算法
Flash
MCU本身的保护机制：
关闭MCU的JTAG仿真器接口
打开MCU的代码防读取功能

应用2：物联网设备的身份认证

云服务商如何赋予设备一个可信的身份



工厂环境



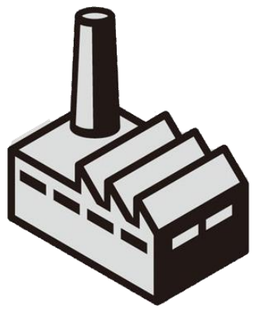
使用环境

授权

读取

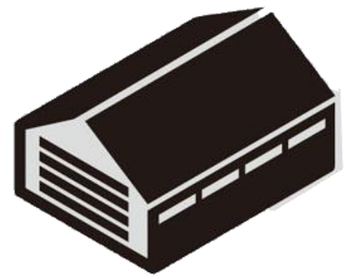
验证

应用2：物联网设备的身份认证：传统方法



工厂环境

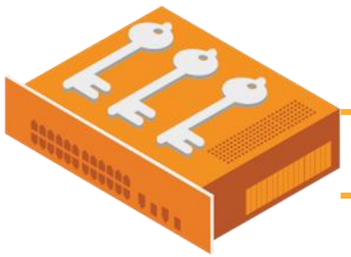
云平台的密钥数据库



使用环境

保存Key

3:验证签名中的Key



HSM
硬件加密模块

证书

Key

处理器

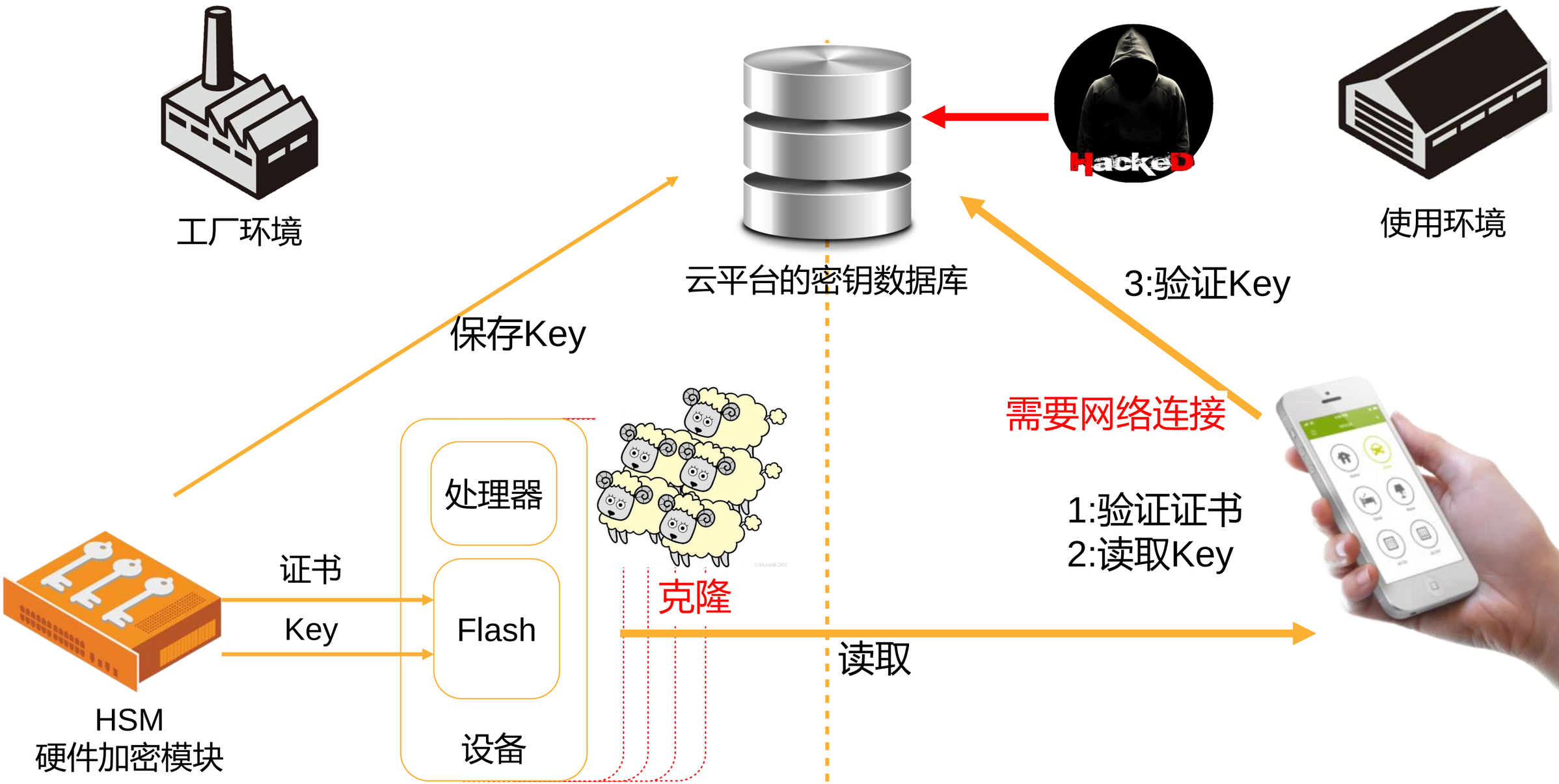
Flash

设备

1:验证证书
2:读取Key

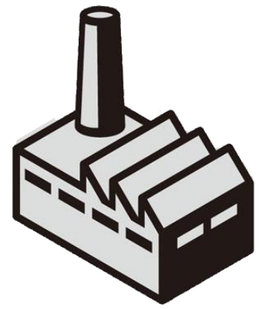
读取

应用2：物联网设备的身份认证：传统方法的安全问题



应用2：物联网设备的身份认证：使用MSCP

使用MSCP的设备身份授权使得每个设备拥有安全的私钥



工厂环境

签发证书

数字签名设备

产测PC

签发证书

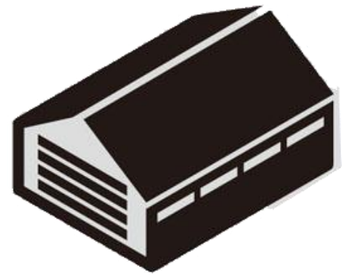
处理器

MSCP

产生私 / 公钥

Flash

设备



使用环境

CA根证书

Digest

证书，Sign(Digest)

1:验证证书
2:验证数字签名

应用3：端到端的信息安全传输

用户与设备之间直接建立安全通讯，独立于任何第三方

大部分云服务会保存客户的用户数据
在某些应用上会有安全风险

非安全网络环境

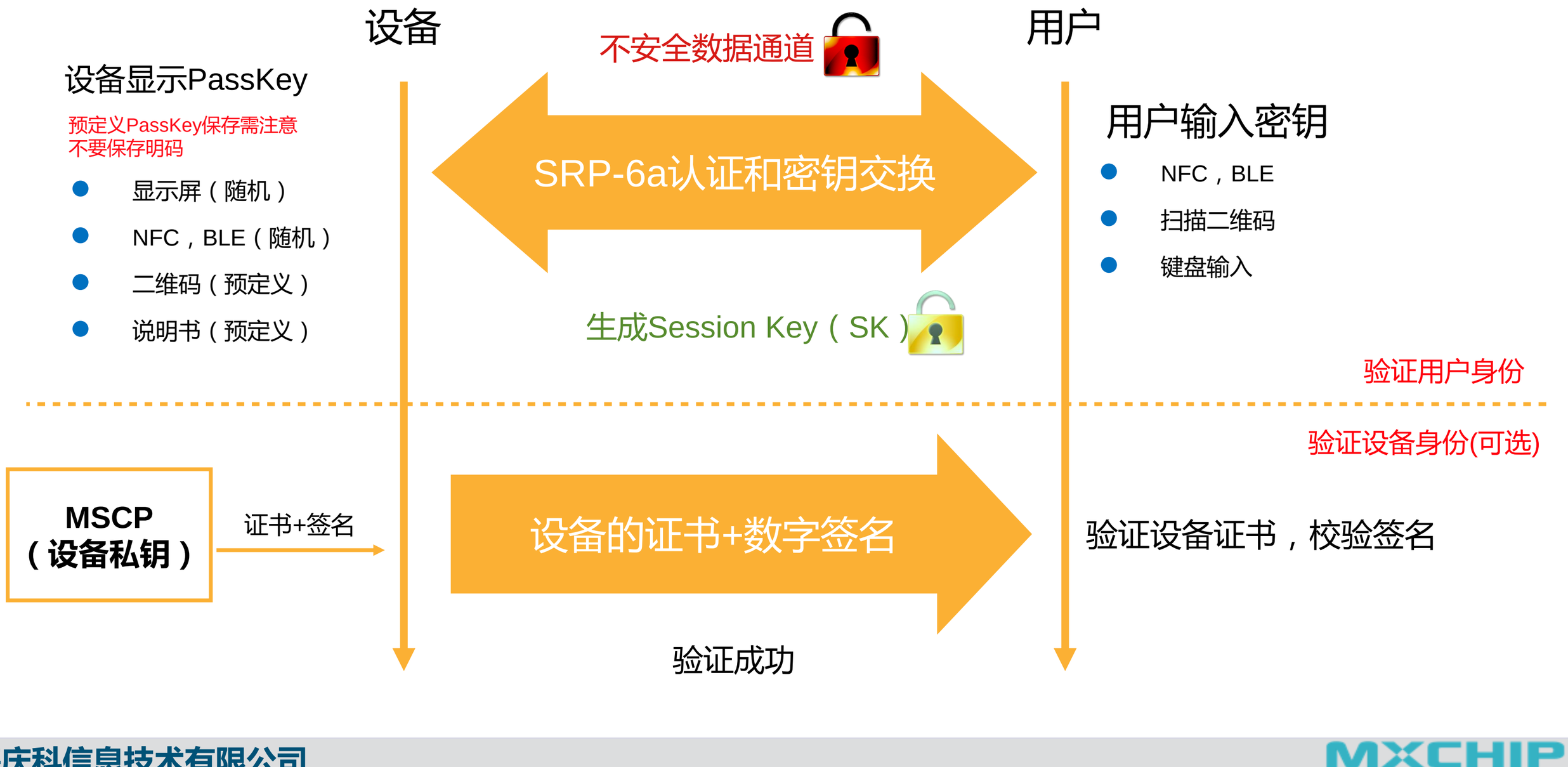
安全网络环境

安全通道独立于整个传输环节

在安全网络环境构建与设备的信任关系

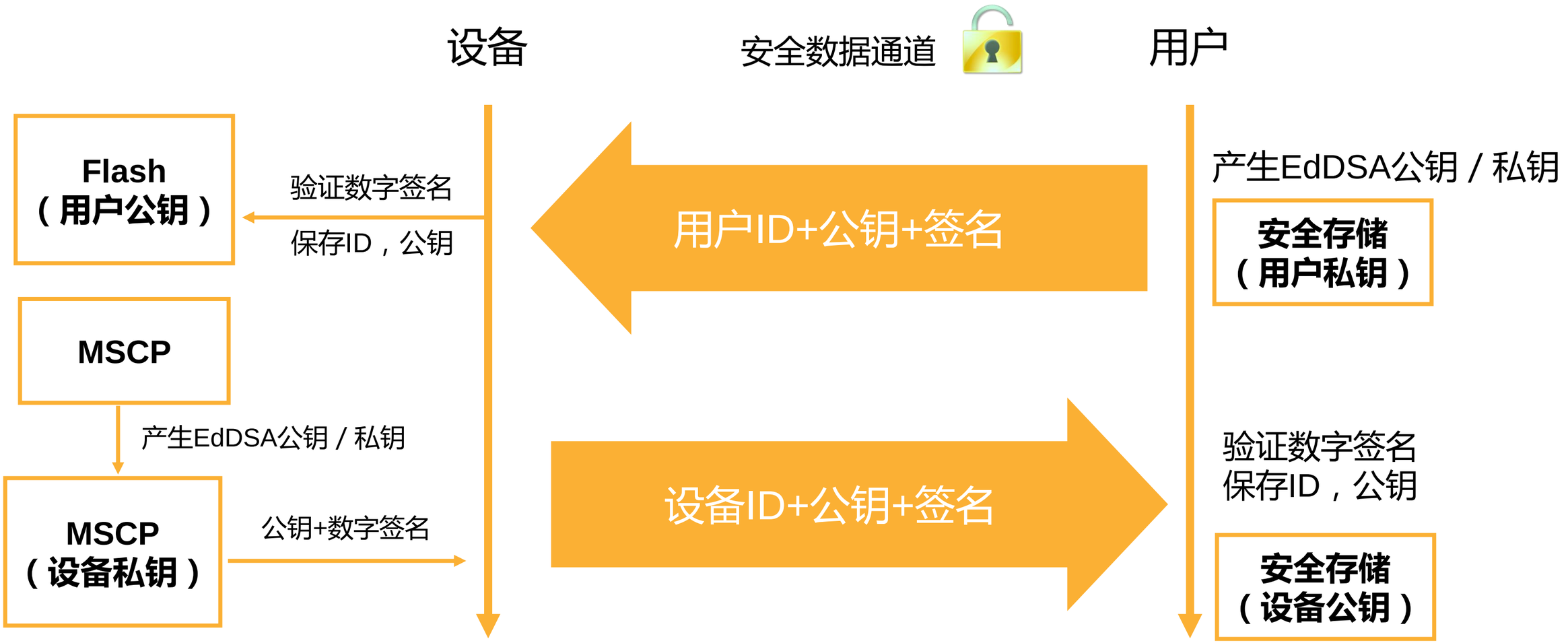
应用3：端到端的信息安全传输

在可信的环境下通讯双方进行身份验证：基于SRP-6a强密钥交互协议+设备证书



应用3：端到端的信息安全传输

在可信的环境下通讯双方进行身份验证：建立数字签名需要的私 / 公钥

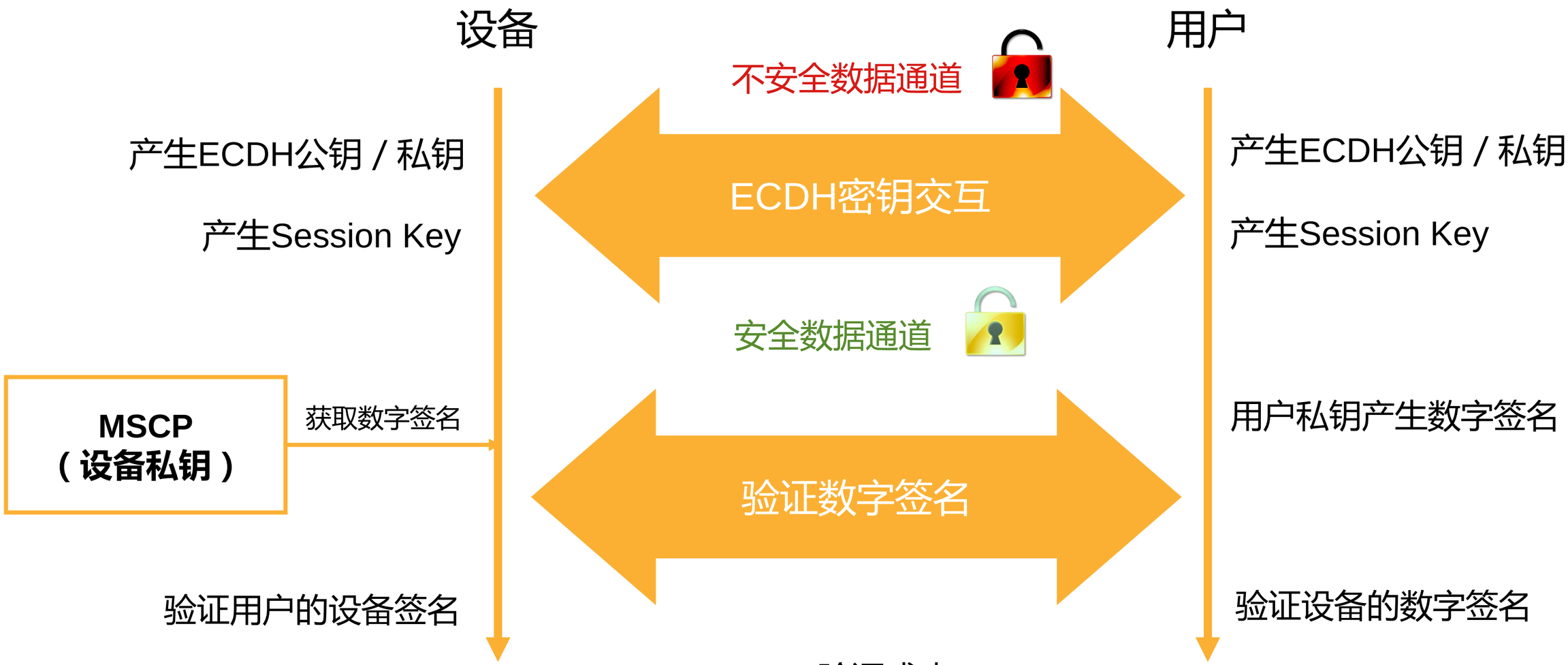


完成，关闭通讯链路

下次设备通讯时可以通过保存的对方的公钥验证对方的数字签名，确保设备的合法性

应用3：端到端的信息安全传输

在通讯前建立安全通道并验证双方的身份



验证成功

接下来可以用Session Key加密接下来的数据通讯了

应用3：端到端的信息安全传输

设备和云建立信任关系是云服务进行数据转发的基础

谢谢大家