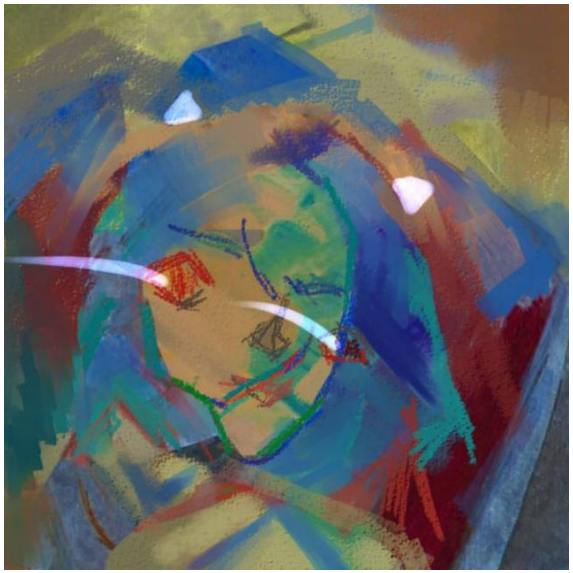
CanoKey 应用、代码与实现

郑鉉壬
i@zenithal.me

2022-01-12

自我介绍



(a) **Z**enithal**H**ourly**R**ate

(b) VX: 蓝天白云**红人**

- 郑鉱（hóng）壬（rén）（LaTeX 默认中文字体不能显示中间这个字）
- 清华大学交叉信息研究院（姚班）大四
- 目前在 PLCT Chisel 小组实习
- CanoKey 用户（贡献过代码）

目录

- 讲 CanoKey 有几个目的
  - 用户：为什么要用 Key，怎么用 Key
  - 软件开发者：怎么用 Key 增加网站/应用安全性
  - 硬件密钥开发者：需要怎样的硬件条件，软件是怎么组织的
- 目录
  - 应用与实例
  - 协议栈与代码组织
  - 讨论：用 RISC-V 实现，需要什么

1

为什么需要硬件密钥

密钥管理

- 我们有很多密钥
  - 密码（网站密码，开机密码等）
  - SSH 密钥（例如 `~/.ssh/id_rsa`）
  - X.509 证书（签名、解密、认证）
  - OpenPGP 密钥（签名、解密、认证）

密钥管理

- 我们有很多密钥
  - 密码（网站密码，开机密码等）
  - SSH 密钥（例如 `~/.ssh/id_rsa`）
  - X.509 证书（签名、解密、认证）
  - OpenPGP 密钥（签名、解密、认证）
- 难以管理

密钥管理

- 我们有很多密钥
  - 密码（网站密码，开机密码等）
  - SSH 密钥（例如 `~/.ssh/id_rsa`）
  - X.509 证书（签名、解密、认证）
  - OpenPGP 密钥（签名、解密、认证）
- 难以管理
  - 难以记忆：密码，4096 位的 SSH 密钥

密钥管理

- 我们有很多密钥
  - 密码（网站密码，开机密码等）
  - SSH 密钥（例如 `~/.ssh/id_rsa`）
  - X.509 证书（签名、解密、认证）
  - OpenPGP 密钥（签名、解密、认证）
- 难以管理
  - 难以记忆：密码，4096 位的 SSH 密钥
  - 难以隔离：你运行的所有程序（包括你不信任的程序）都能够读取到，操作系统不阻拦
    - 你运行的联网软件能够读取并上传你的 SSH 私钥！

密钥管理

- 我们有很多密钥
  - 密码（网站密码，开机密码等）
  - SSH 密钥（例如 `~/.ssh/id_rsa`）
  - X.509 证书（签名、解密、认证）
  - OpenPGP 密钥（签名、解密、认证）
- 难以管理
  - 难以记忆：密码，4096 位的 SSH 密钥
  - 难以隔离：你运行的所有程序（包括你不信任的程序）都能够读取到，操作系统不阻拦
    - 你运行的联网软件能够读取并上传你的 SSH 私钥！
  - 难以迁移
    - 一台机器的密钥如何安全拷贝到另一台机器上（聊天软件分发?）
    - 一台电脑遗失了需要吊销所有电脑的密钥

密钥管理

- 我们有很多密钥
  - 密码（网站密码，开机密码等）
  - SSH 密钥（例如 `~/.ssh/id_rsa`）
  - X.509 证书（签名、解密、认证）
  - OpenPGP 密钥（签名、解密、认证）
- 难以管理
  - 难以记忆：密码，4096 位的 SSH 密钥
  - 难以隔离：你运行的所有程序（包括你不信任的程序）都能够读取到，操作系统不阻拦
    - 你运行的联网软件能够读取并上传你的 SSH 私钥！
  - 难以迁移
    - 一台机器的密钥如何安全拷贝到另一台机器上（聊天软件分发?）
    - 一台电脑遗失了需要吊销所有电脑的密钥
- CanoKey 提供隔离（不可导出密钥）且便携的硬件密钥

多因子验证

- 多因子验证（MFA）是指，除了帐号密码，还需要通过其他的因子验证登录者即帐号所有者

多因子验证

- 多因子验证（MFA）是指，除了帐号密码，还需要通过其他的因子验证登录者即帐号所有者
- 多因子验证已经被实践了很久
  - 手机号 SMS 短信验证码（OTP）
  - 生物因素（人脸识别、指纹）（Windows Hello）
  - 软件密钥（手机，Microsoft Authenticator）

多因子验证

- 多因子验证（MFA）是指，除了帐号密码，还需要通过其他的因子验证登录者即帐号所有者
- 多因子验证已经被实践了很久
  - 手机号 SMS 短信验证码（OTP）
  - 生物因素（人脸识别、指纹）（Windows Hello）
  - 软件密钥（手机，Microsoft Authenticator）
- 硬件密钥也能提供一重因子
  - 密码学保证其难以伪造（对比手机号、人脸等）
  - 安全芯片保证其破解成本高
  - 国内支持硬件密钥的帐号系统较少

多因子验证

- 多因子验证（MFA）是指，除了帐号密码，还需要通过其他的因子验证登录者即帐号所有者
- 多因子验证已经被实践了很久
  - 手机号 SMS 短信验证码（OTP）
  - 生物因素（人脸识别、指纹）（Windows Hello）
  - 软件密钥（手机，Microsoft Authenticator）
- 硬件密钥也能提供一重因子
  - 密码学保证其难以伪造（对比手机号、人脸等）
  - 安全芯片保证其破解成本高
  - 国内支持硬件密钥的帐号系统较少
- MFA 根据不同的场景（安全需求）可以有不同的变种
  - 要求所有的的因子都存在，例如密码，生物信息，硬件密钥缺一不可
  - 要求只存在两重因子，除了密码外，生物信息、硬件密钥、验证码任一均可
  - 去除密码，只要求一重因子（一键登录、Passwordless，SSH 不允许密钥登录）

2

CanoKey 总览

CanoKey 功能：密钥管理

- OpenPGP: RFC4880，实现了 RSA4096，Ed25519 等算法
  - 常被开发者所使用（利用信任网络（WoT））
  - 可用于签署代码、邮件等
  - 可用于加密邮件、文件等
  - 可用于登录系统（SSH）

CanoKey 功能：密钥管理

- OpenPGP: RFC4880，实现了 RSA4096，Ed25519 等算法
  - 常被开发者所使用（利用信任网络（WoT））
  - 可用于签署代码、邮件等
  - 可用于加密邮件、文件等
  - 可用于登录系统（SSH）
- PIV: 用于储存 X.509 证书与密钥
  - 常被机构和企业使用（利用了公钥体系（PKI））
  - 可用于签署文档、发票、代码、邮件等
  - 也可用于加密邮件，文件等
  - 也可用于登录系统

CanoKey 功能：密钥管理

- OpenPGP: RFC4880，实现了 RSA4096，Ed25519 等算法
  - 常被开发者所使用（利用信任网络（WoT））
  - 可用于签署代码、邮件等
  - 可用于加密邮件、文件等
  - 可用于登录系统（SSH）
- PIV: 用于储存 X.509 证书与密钥
  - 常被机构和企业使用（利用了公钥体系（PKI））
  - 可用于签署文档、发票、代码、邮件等
  - 也可用于加密邮件，文件等
  - 也可用于登录系统
- 闲话：WoT 与 PKI 的联系与区别
  - 如何将密钥与对应的身份对应起来，是个困难的问题
  - 公钥体系（PKI）：自顶向下一层层颁发证书，类似身份证
  - 信任网络（WoT）：互相之间的信任与验证，类似博客友链

CanoKey 功能：多因子验证

- OATH：开放认证组织，提供 OTP 的标准
  - 一次性密码（OTP），类似于手机短信验证码
  - 简易标准，更易实现和检查
  - 一般适用于卡离线的场景

CanoKey 功能：多因子验证

- OATH：开放认证组织，提供 OTP 的标准
  - 一次性密码（OTP），类似于手机短信验证码
  - 简易标准，更易实现和检查
  - 一般适用于卡离线的场景
- FIDO2/U2F：FIDO 联盟提供的标准套件
  - FIDO2 是一套协议，有不同的组织方式，与 CanoKey 有关的有 WebAuthn 与 CTAP
  - WebAuthn：W3C 的标准，用于网站服务器、浏览器验证用户（验证用户方式之一是 CTAP）
  - CTAP: Client-To-Authenticator Protocol，用于浏览器与用户密钥之间的通信
  - 需要客户端与卡之间有连接（USB，NFC，蓝牙等）
  - U2F: Universal 2nd Factor，正在被 FIDO2 替代

CanoKey 历史与现有产品

- 上古时期 YutriKey: STM32 + Javacard，上层应用使用现有 Javacard 代码
- CanoKey STM32：供开发，在 STM32（并非安全芯片）上运行，完全开源（包括硬件设计与固件），无密码学加速
- CanoKey Pigeon：最近发售，在安全芯片上运行，有密码学加速，核心代码开源

3

友商与类似产品

友商与类似产品

- Yubikey
  - 行业头部
  - 很多标准都有其参与、制订，很多库都有其开发
  - 其实现不开源，CanoKey 的核心实现开源
  - 其一些应用是私有协议
- Nitrokey
  - 开源安全密钥
  - 实现协议较少
- HSM（Hardware Security Module）
  - 一般用于服务器场景，不同应用有不同的 HSM 需求
  - 功能上类似，应用场景不同，实现标准也不同（可能有私有协议）
- TPM（Trusted Platform Module）
  - 与设备绑定在一起的安全组件
  - 功能上类似，应用场景不同，实现标准也不同

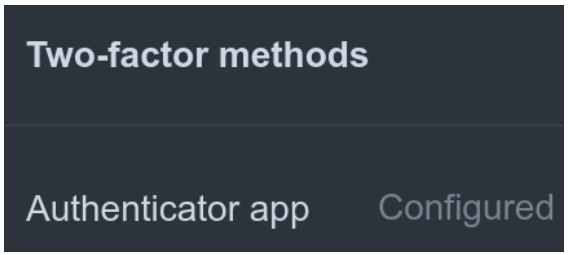
4

具体展示

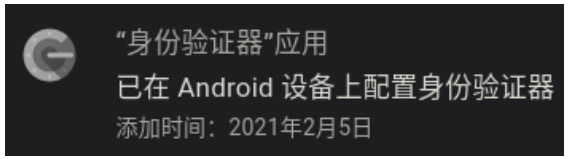
OATH：TOTP 和 HOTP

- 根据场景不同，OATH 有 TOTP 与 HOTP 两种标准
- TOTP 根据时间生成不同的 OTP（需要向卡内传入时间）
- HOTP 根据计算次数不同生成不同的 OTP（卡内维护计数器）
- TOTP 计算一次的结果类似于下图

- GitHub，Google 等均支持该验证方式



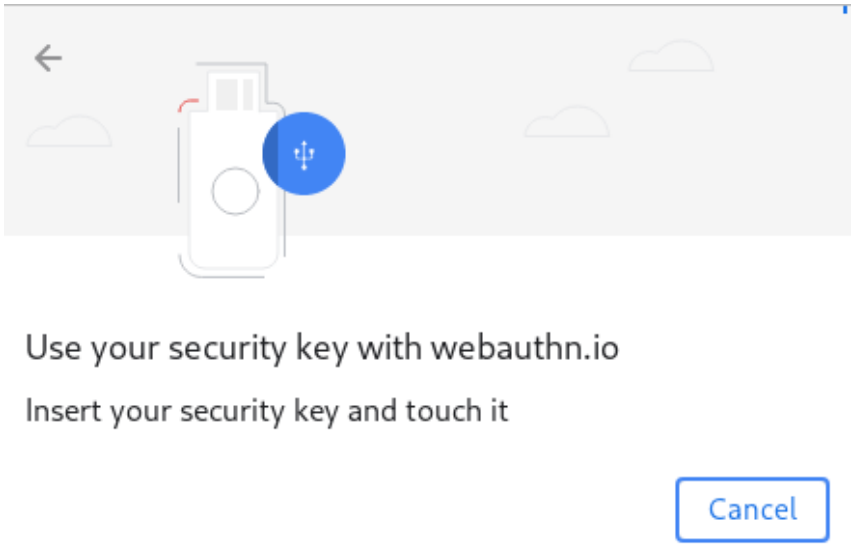
(a) GitHub

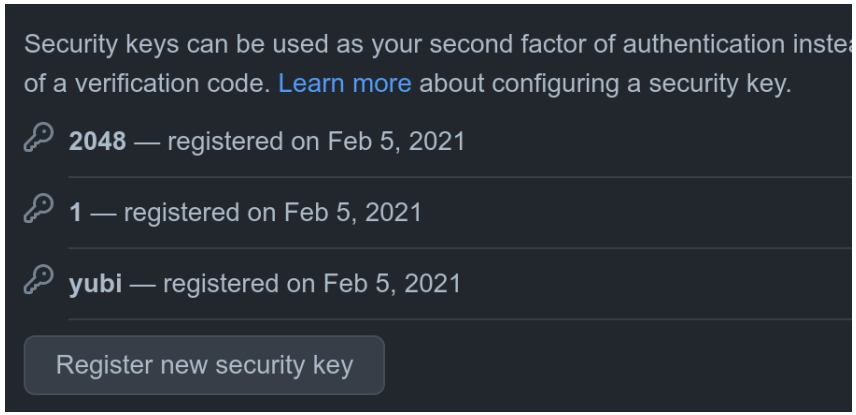


(b) Google

(c) Microsoft

- 登录网页时，二步验证要求插入智能卡并触摸
- 一些网站也支持该验证方式（右图为 GitHub）





OpenPGP

- OpenPGP 密钥有四种角色
  - C（Certificate）：对密钥签名（WoT 特色）
  - S（Signing）：文档签名、邮件签名
  - E（Encryption）：加密数据
  - A（Authentication）：认证（例如 SSH 登录）
- 最佳实践是各个用途单独一个密钥
  - 虽然可以将所有功能集中在一个密钥上
  - 即使交出了用于加密的密钥，签名的不可否认性并不会被影响
- GPG 是 OpenPGP 的一种实现，下图为 GPG 的截图

```
$ gpg --list-keys 1127F188280AE3123619332987E17EEF9B18B6C9
pub   rsa4096 2020-11-16 [SC]
      1127F188280AE3123619332987E17EEF9B18B6C9
uid           [ultimate] Zenithal <i@zenithal.me>
uid           [ultimate] Hongren Zheng (Tsinghua University
uid           [ultimate] Zenithal <z18ham@gmail.com>
uid           [ultimate] ZenithalHourlyRate (GitHub) <i@zen
uid           [ultimate] Hongren Zheng (TUNA) <hongren.zhen
uid           [ultimate] Hongren Zheng (Tsinghua University
uid           [ unknown] Hongren Zheng (Tsinghua University
uid           [ unknown] Zenithal (MirrorZ) <zenithal@mirro
uid           [ unknown] Zenithal (Canokeys) <zenithal@canok
uid           [ unknown] Zenithal (SDF) <zenithal@sdf.org>
sub   ed25519 2021-01-11 [S]
sub   ed25519 2021-01-11 [A]
sub   cv25519 2021-01-11 [E]
```

PIV

- PIV 提供的功能类似于 OpenPGP
  - Digital Signature：类似 OpenPGP 的 S（邮件签名标准为 S/MIME）
  - Encryption：类似 OpenPGP 的 E
  - PIV Authenticate：验证你是你，类似 OpenPGP 的 A
  - Card Authenticate：验证卡是卡（例如门禁）
- PKI 特色，除了要向卡内导入密钥，还要导入机构颁发的证书

5

协议栈

协议栈

应用 | OpenPGP | PIV | OATH | CTAP | Admin | NDEF | Meta

- 应用层均是之前提到的应用
- OATH 是 HOTP/TOTP 的合称
- CTAP 是 FIDO2 的一部分，涵盖 Client-To-Authenticator 这一部分
- Admin 用于管理卡中的一些参数
- NDEF 用于储存用户自定义的信息
- Meta 用于上报一些元数据，为了与 ykman 兼容

协议栈

应用 | OpenPGP | PIV | OATH | CTAP | Admin | NDEF | Meta

指令 | APDU

- 这些应用均由 APDU（Application Protocol Data Unit）来驱动
- 分为 C-APDU 与 R-APDU 两种（Command, Response）[1]

Table 3: Format of C-APDU

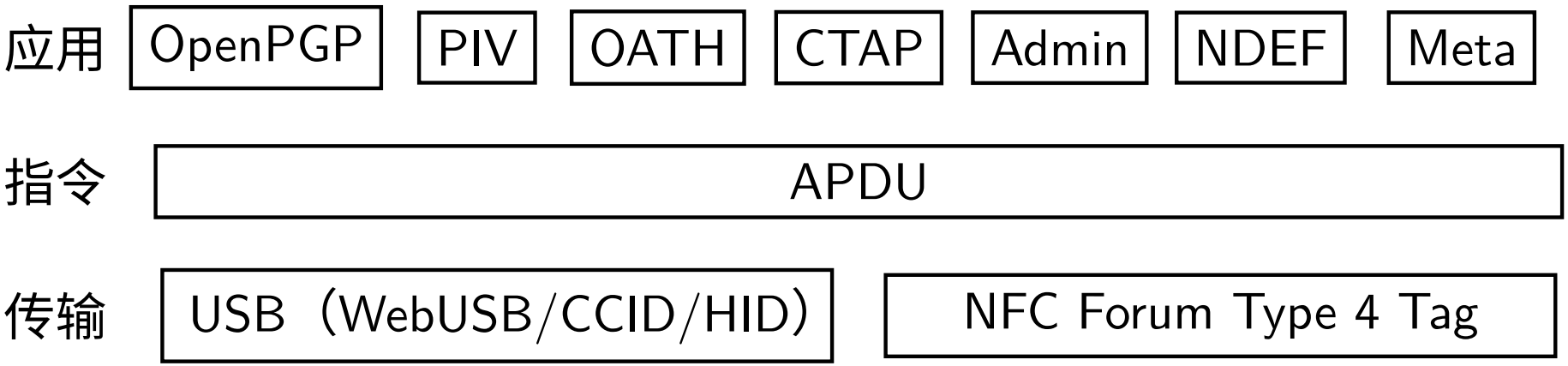
CLA	INS	P1	P2	Lc (optional)	Data (optional)	Le (optional)
Class byte	Instr. byte	Param. byte 1	Param. byte 2	Lc field	Data bytes (Lc bytes)	Le field

Table 4: Format of R-APDU

Response Body (optional)	SW1	SW2
Data bytes	Status Word 1	Status Word 2

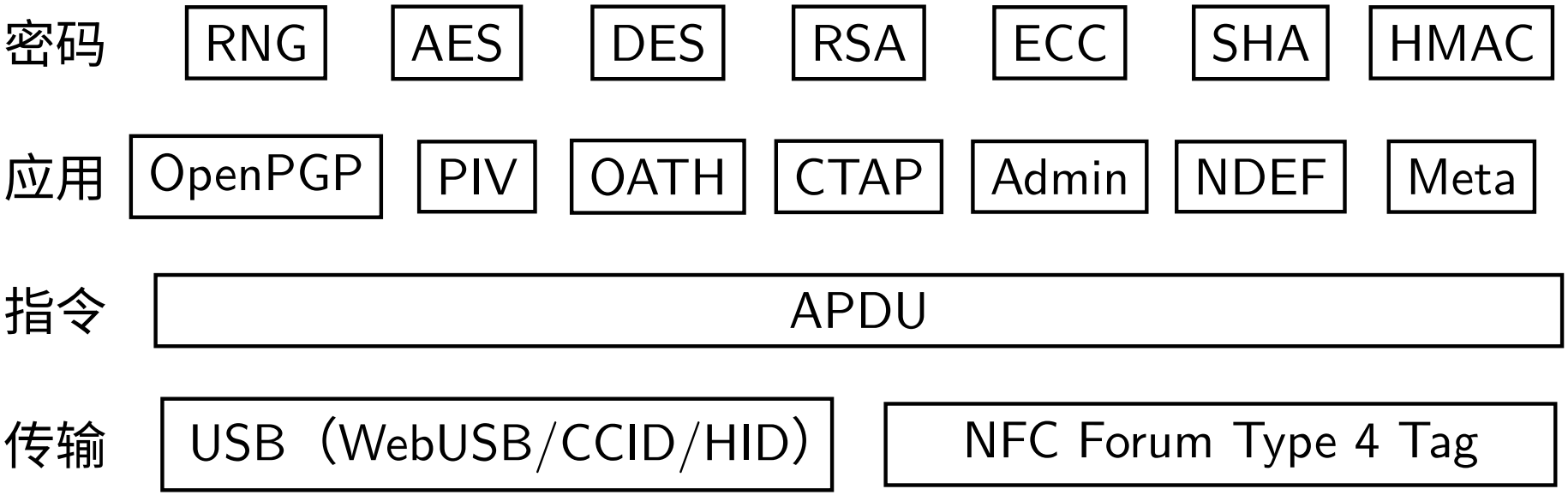
[1]图源 NFC Forum Type 4 Tag Operation Specification

协议栈



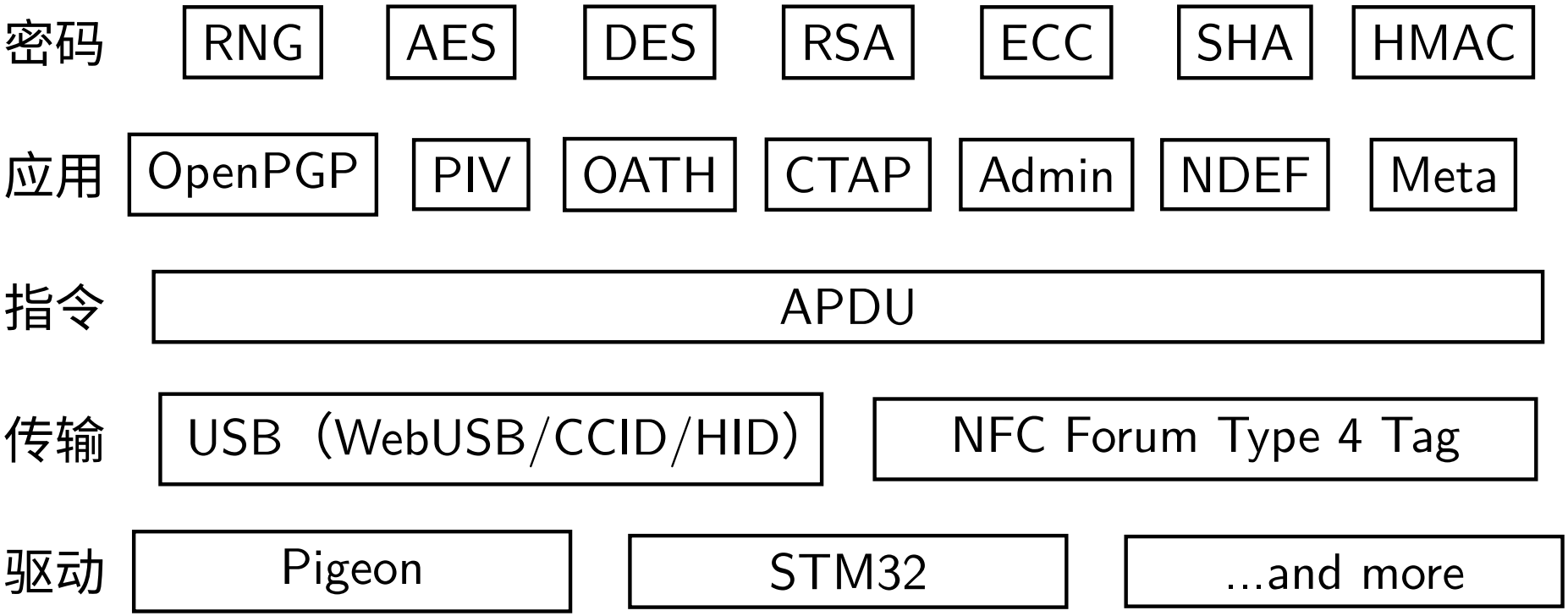
- 指令通过 USB 或 NFC 来传输
- 对于 USB 这边，根据应用的不同，有不同的传输方式
- CCID（chip card interface device）：用于 OpenPGP，PIV, OATH
- HID（human interface device）：用于 CTAP，OATH-HTOP
- WebUSB：用于 Admin

协议栈



- 应用之间可以共享一些密码学操作，故单独拆成一层
- RNG：随机数发生器，用于生成密钥、nonce 等
- AES，DES：对称加密，PIV 和 CTAP 使用
- RSA，ECC：非对称加密/签名，OpenPGP，PIV 和 CTAP 使用
- SHA，HMAC：哈希与认证码，被其他密码学操作使用，以及 OATH 会直接使用

协议栈

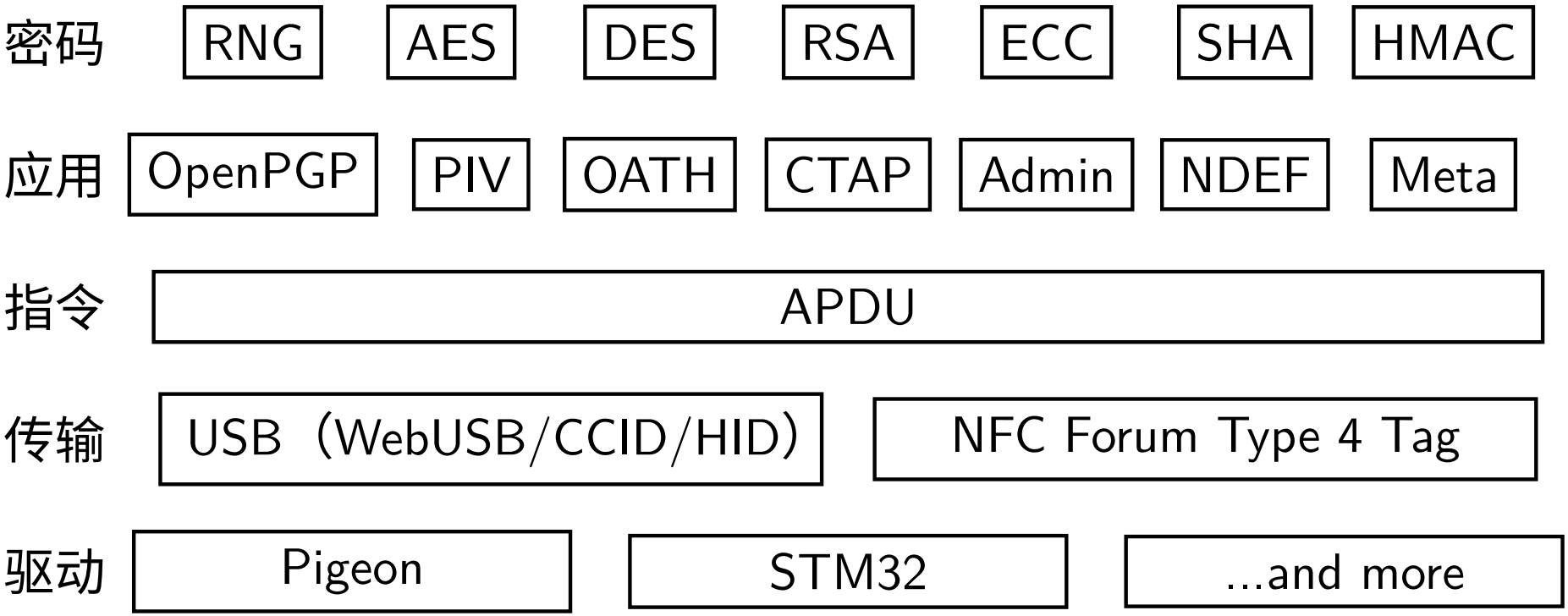


- 针对不同的硬件条件，会有不同的驱动存在，这也单独拆成一层

6

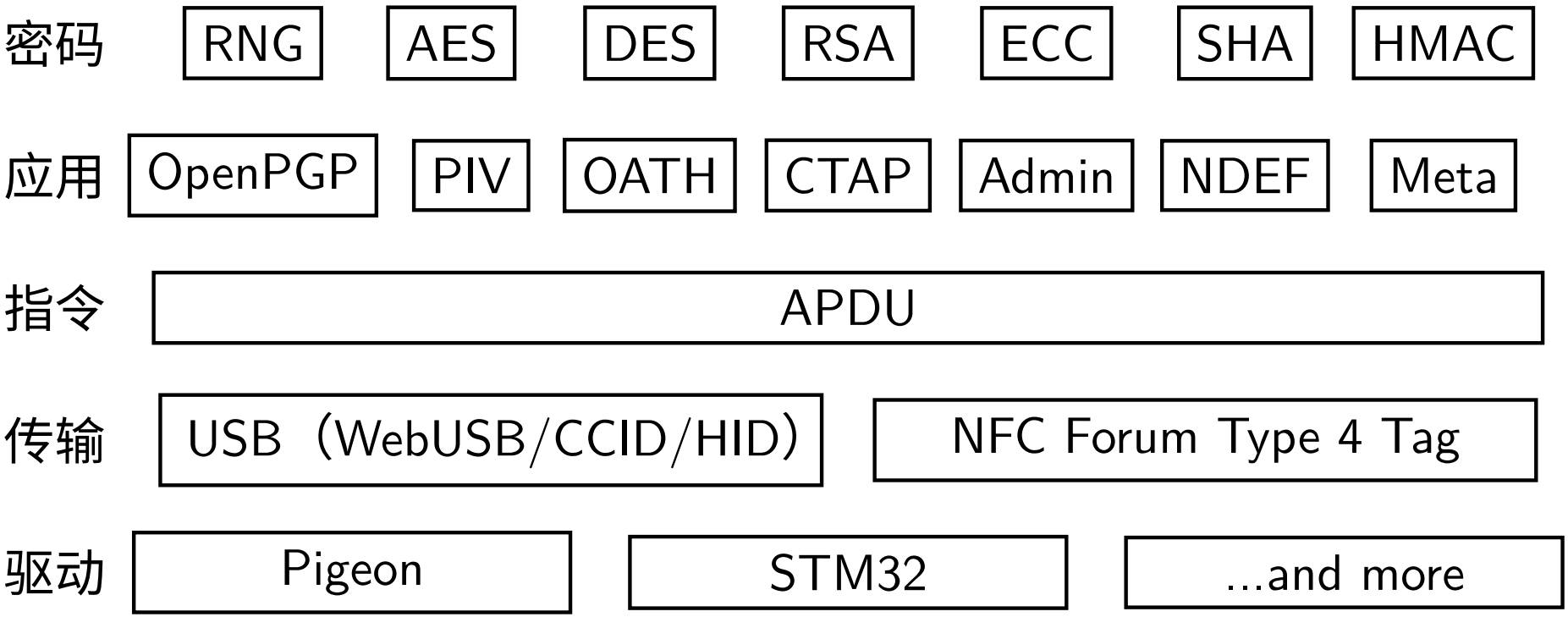
代码组织

代码组织



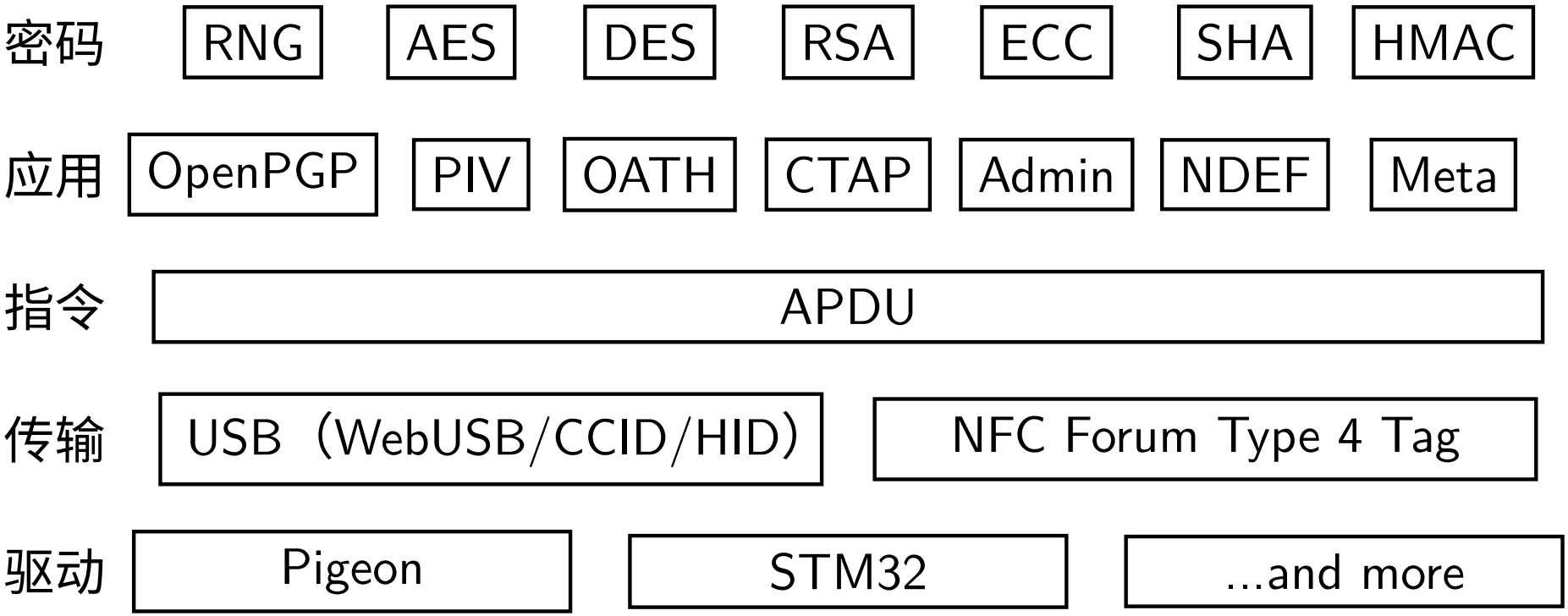
- 核心代码包括应用、指令、传输三层
  - 核心代码即 `https://github.com/canokeys/canokey-core`
  - 应用代码在 `canokey-core/applets` 下
  - 传输代码在 `canokey-core/interfaces` 下
  - 指令代码在 `canokey-core/src` 下（包括其他代码，例如文件系统，触摸）

代码组织



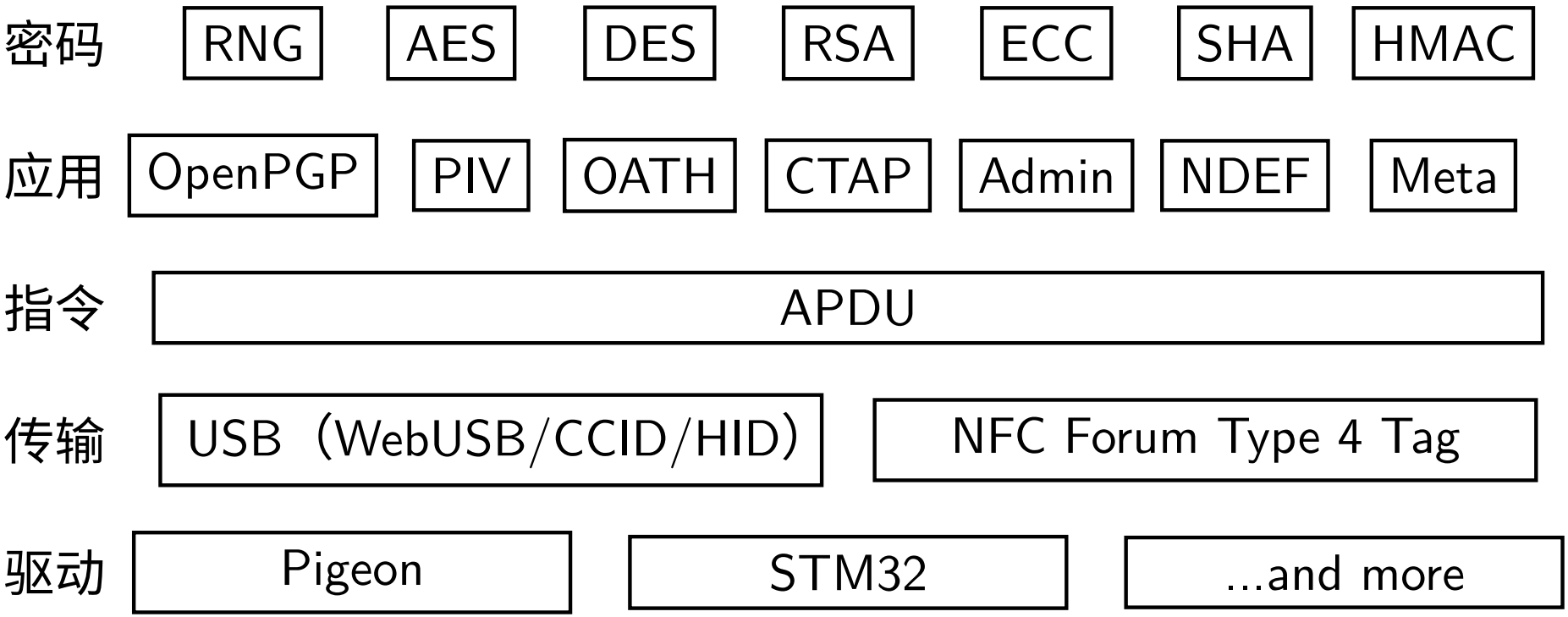
- 硬件相关在单独的项目中
  - STM32 的驱动：`https://github.com/canokeys/canokey-stm32`
  - STM32 版的硬件设计：
    `https://github.com/canokeys/canokey-hardware`

代码组织



- 密码学相关在单独的项目中：canokey-crypto

代码组织

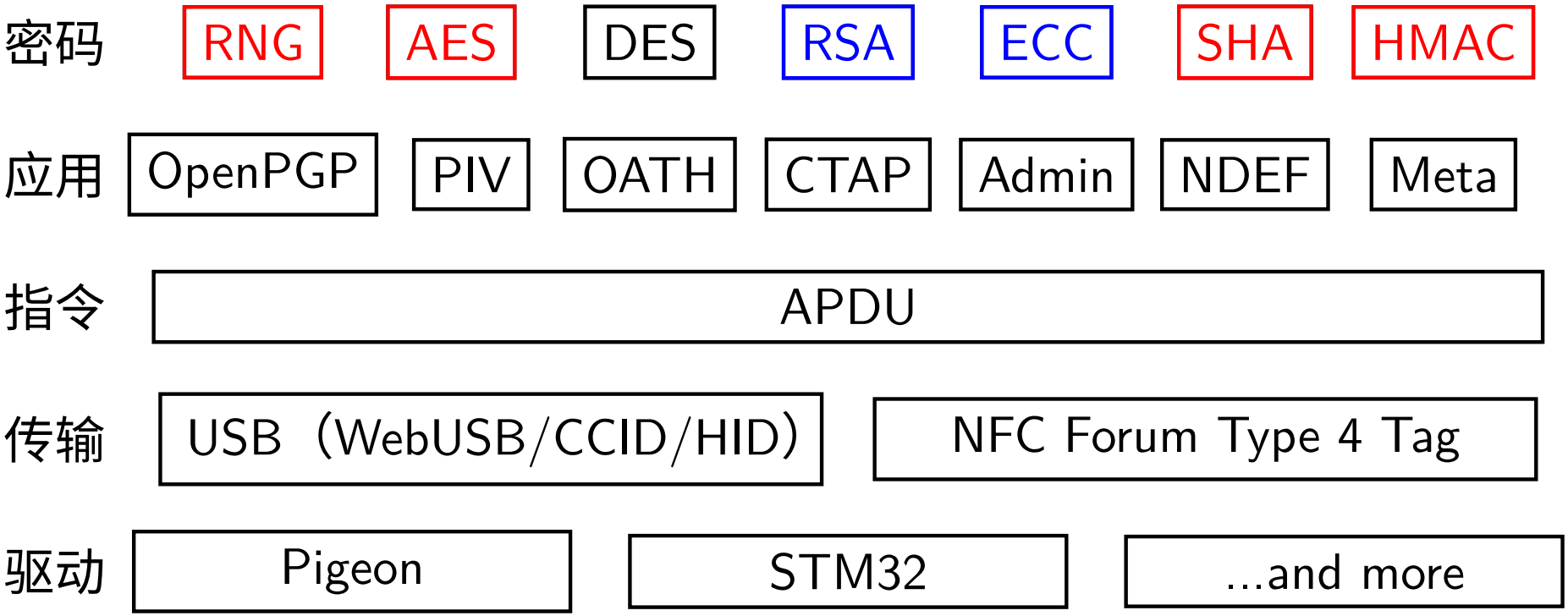


- 代码组织
  - 硬件：可使用 canokey-hardware 的板子，或使用开发板
  - 固件：驱动 canokey-stm32，核心 canokey-core，密码学 canokey-crypto （submodule 关系）
  - 软件：canokey-web-console, canokey-management-tool 用于管理

7

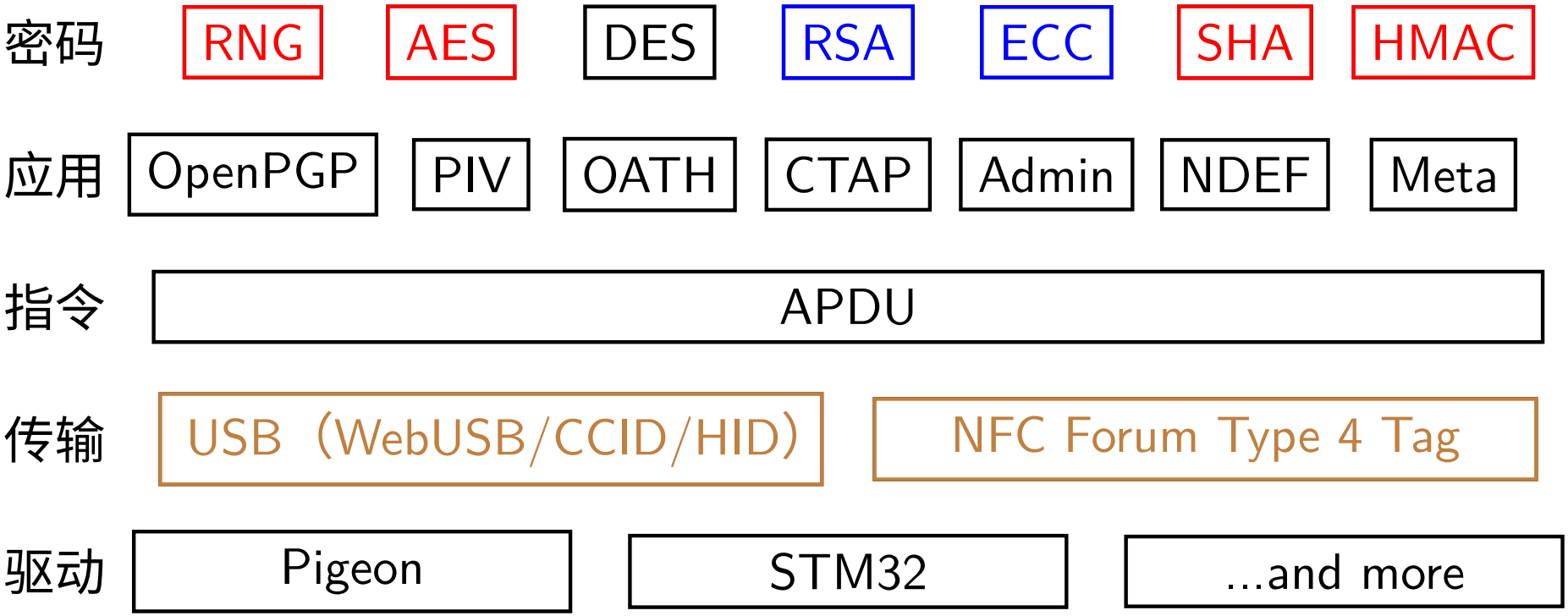
讨论：RISC-V 实现

讨论：RISC-V 实现



- 密码学部分
  - 红色部分可以用 RV K 扩展实现
  - 蓝色部分主要使用 MMM（Montgomery Modular Multiplication）
  - MMM 中主要是大整数加法与位移

讨论：RISC-V 实现



- 驱动与硬件部分
  - USB 设备控制器
  - NFC 部分