计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330874
61(9): 2145−2155, 2024

基于 Uptane 的汽车软件在线升级优化框架

谢 勇[1] 胡秋燕[2] 李仁发[3] 谢国琪[3] 肖 甫[1]
[1](南京邮电大学计算机学院 南京 210003)
[2](万集科技股份有限公司 北京 100193)
[3](湖南大学信息科学与工程学院 长沙 410082)
(yongxie@njupt.edu.cn)

An Optimized Over-the-Air Software Update Framework Based on Uptane for Automobiles

Xie Yong[1], Hu Qiuyan[2], Li Renfa[3], Xie Guoqi[3], and Xiao Fu[1]
[1](*School of Computer Science*, *Nanjing University of Posts and Telecommunications*, *Nanjing* 210003)
[2](*VanJee Technology*, *Beijing* 100193)
[3](*College of Computer Science and Electronic Engineering*, *Hunan University*, *Changsha* 410082)

Abstract Autonomous driving, connected vehicles, electrification of the powertrain, and shared mobility lead to the rapid increasing of the complexity of automotive electronic system, and the functional safety and cyber-security problems of the automotive electronic system cause a serial of recalls frequently, which is resulting in a huge economic loss and user experience decline of the original equipment manufacturer (OEM). The over-the-air (OTA) update technology uses wireless network to achieve remote update of software and firmware in scenarios such as automatic driving function update, on-board software update and on-board safety system upgrade, thus avoiding the adverse impact of recall, but how to guarantee the cyber-security and efficient implementation of the OTA processes is a key problem needing to be resolved by auto industry. The opensource Uptane framework is an industry reference specification for OTA, but it has some cyber-security vulnerabilities and brings too large system overhead for its current reference implementation. By choosing efficient Hash and signature algorithms and introducing a new verification mechanism based on alliance chain, an optimized Uptane framework is proposed with reduced resource and time overhead and increased cyber-security. The prototype implementation verifies the cyber-security of the optimized Uptane framework, and by comparing with the original Uptane framework, the memory consumption and delay overhead of the optimized Uptane framework are reduced by 6.9% and 28.6%, respectively.

Key words intelligent connected vehicles; over-the-air software update; system resource optimization; blockchain; encryption algorithm

摘 要 “新四化”使得车内电子系统的复杂性骤增,因电子系统的功能安全问题和网络安全问题导致的汽车召回事件频发,这给整车厂商造成巨大的经济损失和用户体验下降.在线升级技术借助于无线网络实现自动驾驶功能更新、车载软件更新和车载安全系统升级等场景下的系统固件和软件的远程升级,可避免汽车召回造成的影响,但是如何保障在线升级的安全和高效实现是汽车行业亟待解决的关键问题.

收稿日期: 2023-10-31; **修回日期**: 2024-05-20
基金项目: 国家自然科学基金项目(62172234); 江苏省自然科学基金项目(BK20211272); 南京邮电大学“1311”人才计划项目
This work was supported by the National Natural Science Foundation of China (62172234), the Natural Science Foundation of Jiangsu Province (BK20211272), and the Research Foundation of NJUPT for 1311 Talents Training Project.
通信作者: 肖甫(xiaof@njupt.edu.cn)

Uptane 开源框架是汽车软件在线升级的行业参考规范，但该框架仍存在安全性和系统资源开销过大等不足. 分别从加密算法选择和引入基于联盟链的验证机制 2 个方面对 Uptane 框架进行优化，以降低实现开销和提升安全性. 通过原型实现和测试验证了所提出 Uptane 优化框架的安全性，并通过与原 Uptane 框架的对比分析可知，所提出优化框架的内存开销和时延开销分别降低了 6.9% 和 28.6%.

关键词 智能网联汽车；软件在线升级；系统资源优化；区块链；加密算法

中图法分类号 TP393

以“新四化”为主要特点的智能网联汽车成为汽车工业发展的主流趋势，但是“新四化”使得车内电子系统的复杂性骤增，一些高端乘用车中的代码量超过了 1 亿行，汽车从封闭的物理系统演化成开放的智能移动终端，这在功能安全和网络安全等方面对智能网联汽车的设计、实现和维护等提出了严峻的挑战[1-2]. 例如 2015 年菲亚特克莱斯勒就曾因白帽黑客发现的网络安全问题而召回了 140 万辆车，2022 年特斯拉因为软件缺陷而带来的潜在安全缺陷而召回了 2.6 万辆车. 传统的汽车召回和软件更新需整车厂商(original equipment manufacturer, OEM)通知经销商和车主，经销商收到汽车以及来自 OEM 的更新包后对汽车进行升级. 这种方式将消耗大量的时间和成本，与此同时也降低了用户体验和满意度. 为此，以特斯拉为代表的 OEM 采用在线(over-the-air, OTA)升级技术，通过无线网络对智能联网汽车中的固件和软件实现远程管理和升级，可有效降低成本和提升用户满意度[3]. 根据分析机构评估，2022 年 OTA 技术将节省超过 350 亿美元的汽车维护成本[4].

OTA 在提升智能网联汽车的安全性和降低 OEM 成本等方面将发挥重要作用，但是 OTA 采用的无线网络传输方式可能引入潜在的网络安全问题. 一个完整的 OTA 系统包括云端服务系统、车端中央控制器(如 T-BOX)等，软件升级包在服务端、转发节点、传输链路和车端都可能面临网络安全攻击，比如拒绝服务攻击、中间人攻击和冻结攻击等[5]. 为此，文献 [6] 提出了面向汽车行业的开源在线软件升级框架 Uptane，对 OTA 相关流程和机制进行了定义. 2018 年 IEEE/ISTO 成立了非营利组织 Uptane Alliance 以推动 Uptane 的发展和应用，2019 年 Uptane 成为 IEEE/ISTO6100 标准. 因此，Uptane 框架已成为汽车行业 OTA 的参考规范.

更新框架(the update framework, TUF)[7] 是第 1 个面向台式机和服务器的软件升级框架，Uptane 在 TUF 的基础之上针对汽车行业的需求进行了定制和优化. 文献 [8] 在 TUF 和 Uptane 的基础上提出了适用于物联网场景的 ASSURED 安全框架，解决了 OEM 和设备交互面临的端到端安全问题和模型中未考虑客户端而可能导致的密钥泄露等问题. 文献 [9] 提出在 OTA 过程中采用多种角色对元数据文件进行签名和验证，以保障 OTA 安全. 文献 [10−11] 采用基于可信执行环境的系统隔离技术来提高 OTA 安全，但是可信执行环境如 TrustZone 自身可能存在安全漏洞，因此无法保证 OTA 的安全. 文献 [5] 基于云平台和蜂窝网络设计实现了一个汽车 OTA 框架，该框架采用基于密文策略的属性加密方法保证 OTA 内容安全，并集成了动态消息流调度策略来优化 OTA 过程的吞吐量和时延. 文献 [12] 将 Uptane 框架应用于无人机场景，以实现单机和多机配置下的远程软件升级. 文献 [13] 对系统功能模块更新进行形式化描述和矩阵建模，在此基础之上对系统功能模块更新进行增量验证，以支持系统功能更新的开发、实施和管理.

区块链技术在分布式系统的安全性保障方面发挥了重要作用，一些研究将其引入 OTA 场景中[14]. 如文献 [15] 对 OTA 过程中面临的安全威胁进行了分析，并提出了基于区块链的安全框架. 文献 [16] 提出了一种基于区块链和智能合约技术的 OTA 方案，以验证固件身份和保障固件的完整性. 文献 [17] 提出了一种基于区块链的汽车 OTA 架构实现方案，以保障 OTA 过程中数据传输的完整性和保密性，以及 OTA 参与方的隐私.

综上所述，现有研究未充分考虑智能网联汽车场景下软件升级的可定制性问题，不能实现在同一时间为不同车辆下发不同的更新，大多数安全升级方案的安全保障建立在信任第三方安全中心的基础上. 并且车端的系统资源有限，软件升级方案的具体实现需在安全性和系统资源开销之间进行权衡. Uptane 框架仅提供 OTA 的流程和规则，缺乏具体实现参考，并且在“采用多重签名机制无法直接适配硬件资源有限的汽车场景”和“无法抵御密钥泄露的中间人攻击和内存资源泄露的拒绝服务攻击”等方面存在不足. 因此，拟基于 Uptane 框架提出一种优化的

OTA 技术框架，既保留 Uptane 可定制化的优势，又不依赖于第三方平台，在提高 Uptane 优化框架安全性的同时降低其资源开销，从而进一步提升其实用性.

1 Uptane 框架

1.1 Uptane 主要组成

如图 1 所示，Uptane 框架包括镜像库（image repository）和管理库（director repository）两个库. 其中，镜像库用于保存和记录 OEM 当前部署的软件镜像及证明其身份的元文件（Targets 文件）；管理库用于提供可定制性，并对更新过程进行管理. 如对于车载软件更新管理而言，管理库可根据当前库状态，向电子控制单元（electronic control unit, ECU）分发适当的软件镜像. 当需要进行软件更新时，车辆首先向管理库提供最新的软件版本清单或者已安装的镜像文件信息，管理库根据车辆版本清单运行库进程，并执行依赖解析和选择下一步需要安装的软件文件. 管理库还维护了一个时钟服务器（time server），为无可靠时钟源的 ECU 提供时间，从而避免冻结攻击.

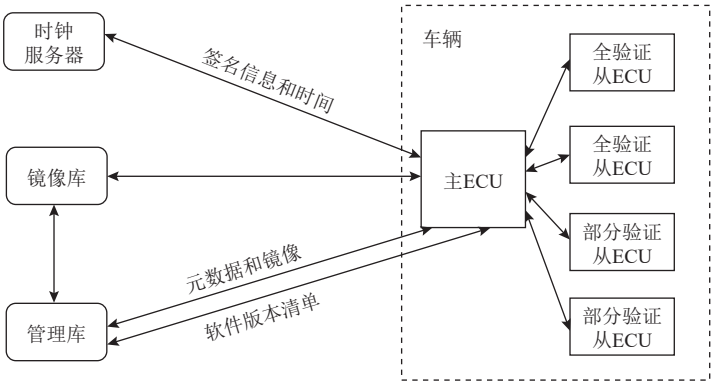

Fig. 1 Composition of the Uptane framework

图 1 Uptane 框架的组成

Uptane 框架采用多重签名机制，管理库和指令库包含了 Root, Targets, Snapshot, Timestamp 共 4 个角色，分别用于对不同的元数据进行签名. 每个角色持有 1 对密钥，不同角色的职责范围不同，某个角色的密钥泄露只影响该角色的职责范围. Root 角色的级别最高，它是整个框架的信任中心，Root 文件中包含授权给所有其他角色的密钥以及自身的密钥. 在使用过程中，需离线保存 Root 角色的密钥. Targets 角色主要提供镜像文件相关的关键元数据，包括哈希值和文件长度等，并对这些内容进行数据签名. Snapshot 角色主要标明库中所有元数据的最新版本，并对 Root 文件和 Targets 文件进行签名，防止 ECU 安装过期的镜像. Timestamp 角色对 Snapshot 文件进行签名，用于标注镜像文件或者元数据是否有改动.

Uptane 包含主 ECU 和从 ECU，其中从 ECU 的算力较弱、存储容量较小，不与外部无线网络进行直接连接；主 ECU 的算力更强、存储容量更大，可连接外部无线网络. 主 ECU 直接与镜像库和管理库交互，交互过程包括签名验证、元数据验证及镜像下载等. 从 ECU 与主 ECU 交互，主 ECU 帮助从 ECU 进行元数据验证，从 ECU 仅针对部分元数据进行验证. 为了降低成本，Uptane 采取分级验证机制，对高安全性 ECU 进行全验证，如核对管理库中 Targets 元数据的哈希值和文件大小是否与镜像库中的一致；对低安全性 ECU 仅进行部分验证，如只验证 Targets 元数据的签名.

1.2 基于 Uptane 的软件在线升级过程

图 2 对基于 Uptane 框架的 OTA 过程进行了描述，该过程主要包括 8 个步骤：

1）主 ECU 收集所有 ECU 和自身的版本清单；

2）主 ECU 从时钟服务器获取最新的时间信息；

3）主 ECU 向管理库上报版本清单，并从管理库中下载元数据，然后进行验证；

4）主 ECU 从镜像库中下载软件升级包，并进行验证；

5）主 ECU 发送元数据给其他 ECU；

6）主 ECU 发送升级包给其他 ECU；

7）ECU 验证升级包与元数据是否匹配，如果匹配则进行软件升级；

8）ECU 软件升级后，将最新的软件版本信息上报给主 ECU，并由主 ECU 上报给管理库.

2 Uptane 框架的优化方案

车用 OTA 框架需同时满足高安全性和资源受限

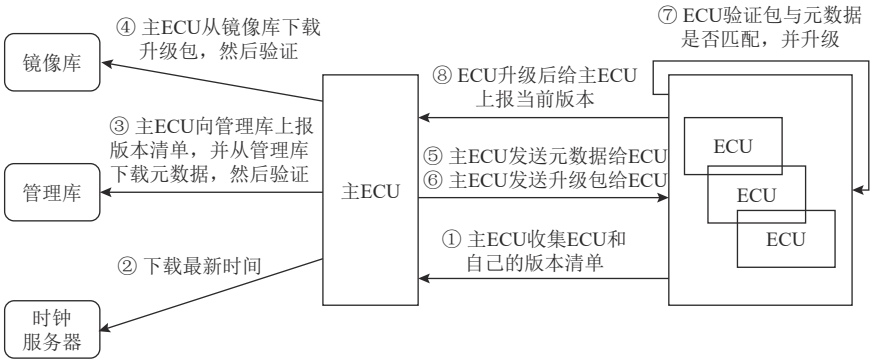

Fig. 2 Uptane-based over-the-air software update process

图 2 基于 Uptane 的软件在线升级过程

要求，但是现有 Uptane 框架采用的多重签名机制不仅消耗大量系统资源，还可能导致过大的时延开销，如主 ECU 进行 1 次更新就需要进行多次签名验证和哈希运算. 另外，Uptane 采用的元数据验证方式虽然可在密钥可靠的前提下防止车辆安装恶意文件，但在建立 TCP 链接前无法进行身份识别，这使得车辆容易遭受拒绝服务攻击. Uptane 未采取密钥保护措施，即使密钥存储在硬件资源中，攻击者还是可以通过侧信道攻击获取密钥或者攻击运行程序破坏密钥、签署恶意文件等. 针对上述问题，本文分别尝试从性能和安全性 2 个方面对现有 Uptane 框架进行优化，以提升其实用性.

2.1 Uptane 框架的性能优化

Uptane 框架中多次采用哈希算法和签名算法来进行安全保障，上述算法决定了其内存开销和实时性能. 其中，哈希算法被用于计算密钥 ID 和传输文件摘要、校验文件完整性和正确性等，签名算法被用于主 ECU 在更新和注册时向管理库提交带有签名的最新车辆版本信息. 因此本文将通过轻量型密码学算法的对比和选择，以及汽车客户端验证算法简化来优化 Uptane 的性能.

在笔记本电脑上搭建 OTA 模拟环境来对主流哈希算法和签名算法的时间开销和内存开销的大小进行对比分析，并基于此来选择最优的加密算法. 虽然该实验环境与车载软硬件环境不同，但是该实验主要关注相同软硬件环境下算法的性能对比情况，因此其结果可用于哈希算法和签名算法的选择. 笔记本电脑的软硬件配置如下：CPU 为 Intel® Core™ i5-4210M，8 GB 内存，Ubuntu 操作系统，编程语言为 Python3.8.5，IDE 为 pycharm. 根据企业工程实践经验，以及考虑到 Uptane 框架的适配性和通用性需求，对比分析的签名算法包括 SM2，ED25519，RSA；哈希算法包括 SHA256，SHA512，SM3. 在保证输入信息一致的前提下，取 100 次实验的平均值来进行对比分析，不同哈希算法和签名算法的运行时间如表 1 和表 2 所示. 通过分析可知，哈希算法中 SHA512 和 SHA256 的时延相近，SM3 的时延稍大；签名算法中 ED25519 和 SM2 的时延大小在同一个数量级，RSA 的时延比前 2 个算法的时延大一个数量级.

Table 1 Comparison of Different Hash Algorithms

表 1 不同哈希算法的对比 ms

哈希算法	时延
SHA256	10.34
SHA512	10.07
SM3	17.90

Table 2 Comparison of Different Signature Algorithms

表 2 不同签名算法的对比 ms

签名算法	时延
SM2	41.22
ED25519	10.45
RSA	420.30

通过设置性能监视器，笔记本除运行 pycharm，Python 之外没有开启其他应用，哈希算法和签名算法的输入为 1 KB 大小的文件，2 类算法运行 100 次后对应的内存消耗情况如图 3 和图 4 所示. 通过对比分析可知，哈希算法中 SM3 的平均内存开销最小，SHA512 和 SHA256 的平均内存开销相近；签名算法中 SM2 的内存开销最小，ED25519 的平均内存开销稍小于 RSA.

哈希算法中 SM3 是对 SHA256 的改进，其安全性更高；SM3 的平均内存开销最小，时延开销与另外 2 个算法在同一个数量级，因此在 Uptane 优化框架中将选用 SM3 来生成哈希值. 签名算法方面，SM2 和 ED25519 均采用了椭圆曲线密码理论，但是 SM2 的第 1 次杂凑算法输入量加入了签名者的可辨别标识，

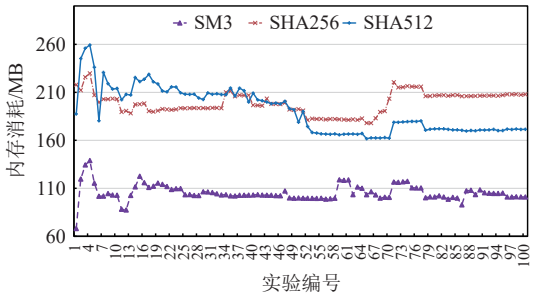

Fig. 3 Memory consumption of different Hash algorithms

图 3 不同哈希算法的内存消耗

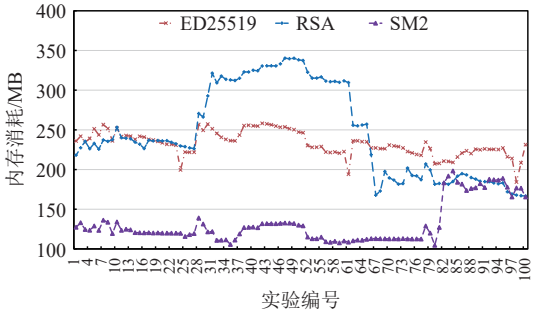

Fig. 4 Memory consumption of different signature algorithms

图 4 不同签名算法的内存消耗

其安全性更高；SM2 内存开销最小，其时延开销与 ED25519 在同一个数量级，因此在 Uptane 优化框架中将选用 SM2 进行签名验证.

2.2 Uptane 框架的安全性优化

联盟链既有公有链的去中心化和可信度，又有私有链的高效性和可控制性，鉴于车端资源受限的情况，在 Uptane 框架中引入联盟链来提高其安全性. 为进一步提升 Uptane 优化框架的性能，提出了基于领域划分的委员会共识机制，采用多节点共同广播的方法来减轻链路负载压力. 在智能合约部分设计了车端分区验证机制，在降低时延和资源开销的同时提高了安全性. 目前，智能网联汽车的电子电气架构正由基于网关的分布式架构向基于域控制器的集中式架构演进，整个汽车将由多个域控制器通过以太网技术互联组成一个分层架构. Uptane 优化框架采取了基于领域划分的分层结构，因此该框架适合于基于域控制器的电子电气架构场景.

联盟链主要包括委员会共识机制和智能合约机制 2 个部分，接下来将对这 2 个机制进行详细说明.

2.2.1 委员会共识机制

本文提出基于 Uptane 框架划分领域的委员会共识机制，图 5 给出了委员会共识机制的架构，其设计包括 3 个关键点：

1）委员会成员生成. 将链中的节点按照所处位置划分为多个领域，每个领域再将所属节点分为 1 级节点（管理库）和 2 级节点（主 ECU）. 每个领域从 1 级节点中选举产生 1 个委员，不同领域的委员组成委员会.

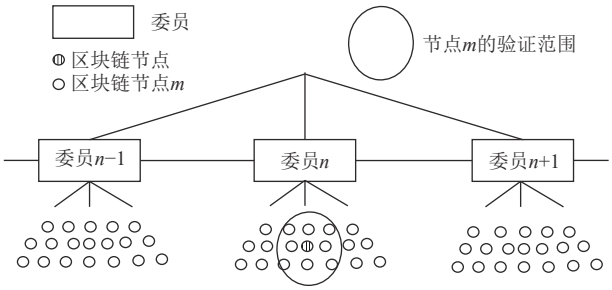

Fig. 5 Committee and node membership structure of the alliance chain

图 5 联盟链委员及节点成员结构

2）委员会代替主 ECU 进行安全验证. 管理库下发更新指令时将向联盟链发布交易信息和智能合约触发信号，委员会自动执行智能合约. 交易信息包括目标文件哈希值、目标文件长度和时间戳等. 委员会将代替主 ECU 进行交易信息真实性验证，验证通过后委员会成员才能生成新区块，并将区块信息向其领域下的节点广播.

3）主 ECU 分区域验证. 当镜像文件验证通过后，管理库下发更新至车辆. 主 ECU 首先从库中下载最新区块进行验证，然后下载目标文件元数据，查看更新文件下载信息. 为降低开销，主 ECU 采取分区域的验证方式. 在车端设备资源约束范围内，尽可能扩大分区以避免某一委员或者某一领域内所有节点被攻击而接收恶意文件，从而保障安全性.

2.2.2 智能合约机制

智能合约是嵌入在联盟链中的一段自动化执行脚本，本文将其嵌入在各委员会成员中，当达到合约执行条件时，智能合约自动执行. 智能合约信息包括激励机制、目标文件真实性验证、新区块计算方式和广播区块等. 图 6 给出了智能合约下的联盟链系统，其设计的关键点有 3 个：

1）激励机制. 为维护区块链稳定和保证各委员会成员的广播质量，本文在智能合约中加入激励机制. 管理库向区块链发送交易信息时附上激励值，委员会成员收到激励值后触发智能合约自动执行. 管理库需要支付激励值来兑换委员会成员执行智能合约，管理库也参与委员会成员选举，成为委员后通过验证和广播交易信息来赚取激励值.

2）目标文件真实性验证. 委员会成员收到激励

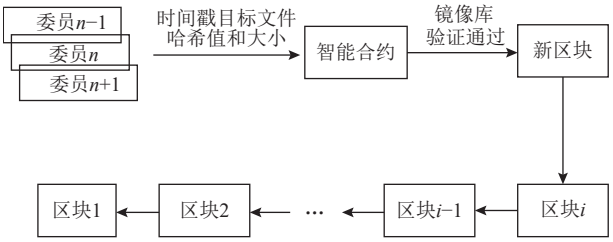

Fig. 6 Alliance chain system under smart contract

图 6 智能合约下的联盟链系统

值之后，首先根据交易信息向镜像库验证交易信息. 当所有委员会成员都验证通过后，执行智能合约区块生成算法生成新的区块，并将其在所属领域内进行广播. 若有委员会成员验证不通过，则不生成新区块.

3)生成新区块. 区块信息包括哈希值和区块索引，每个区块按照时间顺序生成索引，并将其附加在区块哈希值前成为区块头. 智能合约执行和区块存储需要大量算力和存储资源，所以区块链中的 2 级节点不参与智能合约，而只存储一个最新的区块信息，以便其他节点更新时进行验证. 1 级节点中委员会存储所有区块信息，即每个委员都单独记账，拥有自己的账本. 当委员会中有成员新增或变更时，新成员需向现有成员复制账本信息，用于生成新区块. 委员会生成新区块后，向所属领域内的下属节点进行广播.

Uptane 优化框架中联盟链相关的核心代码如算法 1 所示：

算法 1. 服务端委员创建算法.

① *block_start*();

②*create_block_directory*(); /*创建区块链目录结构*/

③ *block_host*(); /*启动区块链服务器*/

④ *listen*(). /*开启监听线程*/

客户端的核心算法如算法 2 所示：

算法 2. 客户端区块创建算法.

① *creat_block*(*vin*, *im_data*, *this_timestamp*);

② *previous_block_index* =*previous_block*[‘index’]; /*获得前一区块索引*/

③ *previous_block_hash* =*previous_block*[‘hash’]; /*获得前一区块哈希值*/

④ *block_to_add* =*next_block*(*previous_block_index*, *previous_block_hash*, *vin*, *im_data*, *this_timestamp*); /*创建新区块*/

⑤ *blockchain. append*(*block_to_add*). /*将新区块加入区块链*/

主 ECU 下载新区块并解码如算法 3 所示：

算法 3. 主 ECU 新区块下载和解码算法.

① *download_block*(*url*);

② *request* =*six. moves. urllib. request. Request*(*url*);

③ *res* =*six. moves. urllib. request. urlopen*(*request*);

④ *data* = *res. read*(). *decode*();

⑤ *data* = *json. loads*(*data*);

⑥ return *data*.

主 ECU 进行区块验证时首先根据本地存储的区块索引和哈希值计算出最新区块的索引和哈希值，然后与从委员下载的区块比对，该过程的核心代码如算法 4 所示：

算法 4. 主 ECU 区块验证算法.

① *verify_block*(*url*);

② *my_block* = *loadmyblock*();

③ *my_block_index* = *my_block*[‘index’];

④ *my_block_hash* = *my_block*[‘hash’];

⑤ *trusted_block* = *get_trusted_block*(*url*);

⑥ if *my_block_index* == *trusted_block*[‘index’]and *my_block_hash* == *trusted_block*[‘hash’]

⑦ return true;

⑧ else

⑨ return false;

⑩ end if

3 Uptane 优化框架的实现

本文将 Uptane 优化框架划分为服务端、客户端、时钟服务器和区块链合约中心 4 个组成部分，并分别实现各个模块的安全机制和功能，以及各模块间的通信. 其中服务端包括指令服务端(管理库)、镜像服务端(镜像库)两个模块，客户端包括主客户端(主 ECU)和从客户端(从 ECU)两个模块. Uptane 优化框架的具体执行流程如图 7 所示.

3.1 服务端实现

Uptane 优化框架服务端的运行流程如图 8 所示，该服务端包括指令服务端、时钟服务器和镜像服务端，在实际应用中镜像服务端和指令服务端一般由不同企业来实现，时钟服务器由指令服务端维护.

镜像服务端主要负责监管所有镜像文件，在初始化完成之后，它将开启一个监听进程，以便其他节点查询和验证镜像文件的真实性. 指令服务端主要负责向更新节点下发指令，在密钥准备完成后，将为管辖内的车辆创建一个元数据文件库，并开启监听

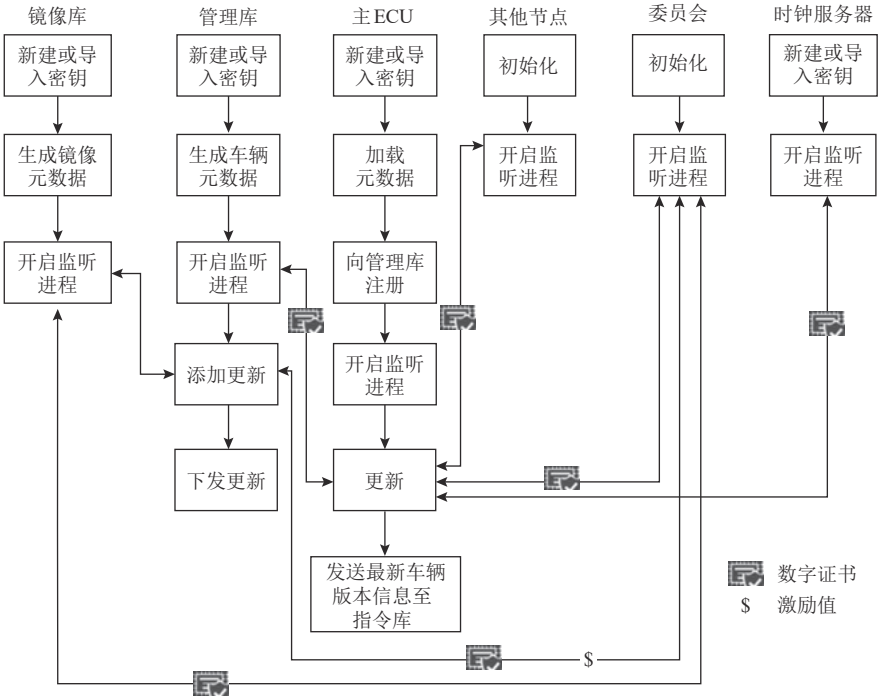

Fig. 7 Execution process of the optimized Uptane framework

图 7 Uptane 优化框架的执行流程

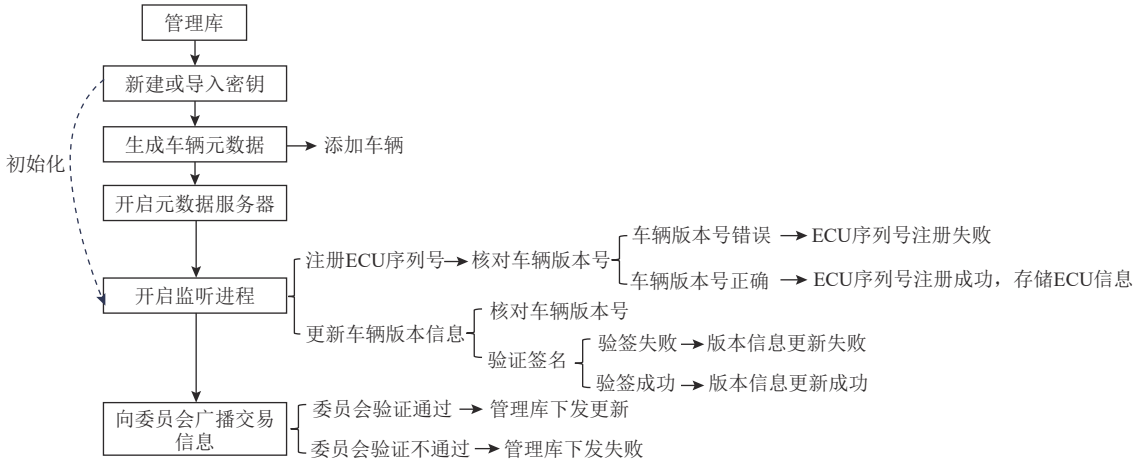

Fig. 8 Detailed server-side process of the optimized Uptane framework

图 8 Uptane 优化框架的服务端详细流程

进程，以便其他节点注册和车辆版本信息的更新. 优化框架对指令服务端中的元数据文件进行简化，指令服务端只保留 Targets 角色，Targets 元数据中只记录当前版本号车辆的更新文件信息和签名信息. 在更新下发之前，向区块链中每个委员节点发放一个激励值，并附上生成新区块所需的时间戳和更新文件信息等.

委员会运行流程如图 9 所示. 委员会成员收到激励值后根据更新文件信息进行镜像库验证，验证通过后自动执行智能合约，并向所属域中的节点广播新区块.

3.2 客户端实现

Uptane 优化框架的客户端包括主客户端（主 ECU）和从客户端（从 ECU），主 ECU 负责文件的更新和元数据的验证以及其他一些对算力要求较高的计算任务. 主 ECU 在完成密钥准备工作之后，加载指令服务端和镜像服务端的 Root 元数据并生成本地库. 本地库中存放了指令库、镜像库的元数据文件，以及更新时下载的镜像文件. 主 ECU 将开启一个进程来监听从 ECU 的 ECU 序列号注册、更新最新的版本信息，并实现 ECU 注册和 ECU 版本信息的注册接口. 从 ECU 注册时只需向主 ECU 申请该功能的调用，并

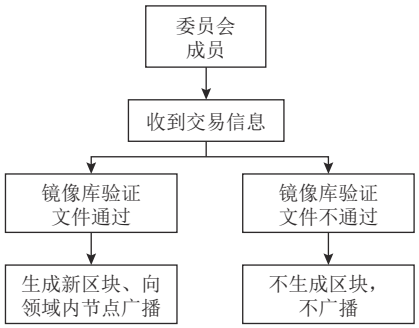

Fig. 9 Execution process of the committee

图 9 委员会执行流程

从主 ECU 获取调用结果. 主 ECU 在更新镜像文件前向时钟服务器获得当前时间，该时间由时钟服务器对其签名后下发至主 ECU，以用于验证元数据文件的有效期，从而保证时间数据的真实性.

Uptane 优化框架在客户端主要针对主 ECU 的流程进行了优化，其实现的详细情况如图 10 所示. 主 ECU 更新不需对镜像库元数据和管理库元数据进行多次签名和算法验签，它从指令库下载简化的 Targets 文件后，若有更新，只需向时钟服务器获取当前时间，然后进行多节点的区块验证，验证成功后即可下载文件. 从 ECU 只需向主 ECU 下载简化后的指令库元数据，在镜像库文件验证成功后下载更新的文件.

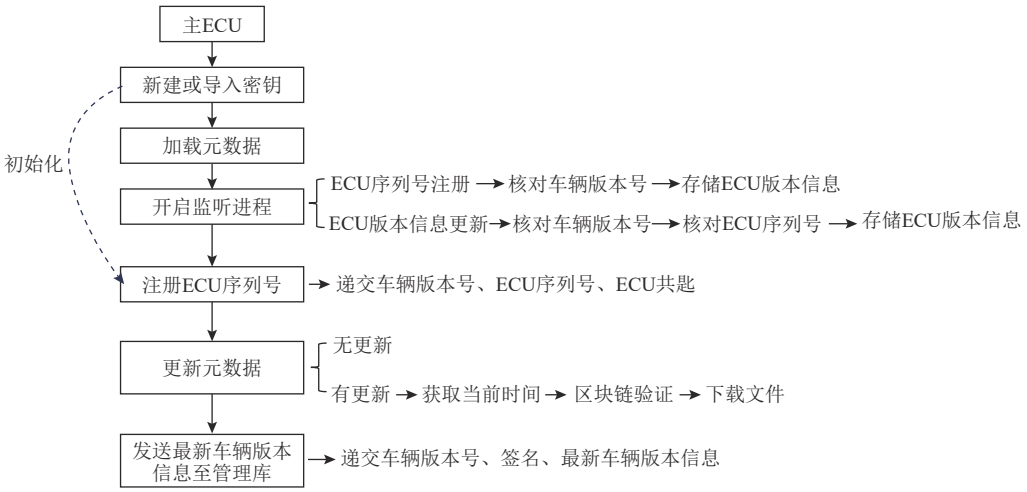

Fig. 10 Detailed client-side process of the optimized Uptane framework

图 10 Uptane 优化框架的主客户端详细流程

3.3 远程安全通信及认证中心结构实现

服务端与主客户端以及其他节点采用远程安全通信技术，通过 HTTPS 双向认证保障通信链路中的数据安全，以防止欺骗和减少拒绝服务攻击的发生. 网络中所有节点的证书都有同一个根认证中心(certificate authority, CA)，利用 Opensll 在 Linux 系统下构建 CA 机构，CA 机构可实现多级证书生成和颁发，根证书采用自签名方式，证书生成遵循 X.509 规范.

4 Uptane 优化框架的安全性测试和性能分析

4.1 测试环境

如图 11 所示，本文搭建了 Uptane 优化框架的性能测试平台. 该平台由 1 台笔记本电脑、2 块树莓派开发板(Raspberry Pi 4 model B)、1 块显示屏和 1 部智能手机组成. 笔记本电脑中运行 64 位 Windows10 操作系统，其硬件配置为：CPU 为 Intel® Core™ i5-4210M，8 GB 内存. 树莓派的内存配置为 2 GB LPDDR4 SDRAM，手机支持第 4 代移动通信网络和 WLAN 热点. Uptane 优化框架的服务端和区块链合约中心部

Fig. 11 Performance testing platform of the optimized Uptane framework

图 11 Uptane 优化框架的性能测试平台

署在笔记本电脑中，客户端部署在树莓派中. 本实验使用 Wireshark 工具来跟踪端到端会话数据流，并对抓取到的数据包进行分析.

4.2 安全性测试

Uptane 优化框架实现了不同节点之间通信的身份认证，因此本实验重点对身份认证机制、通信数据的完整性和机密性、御防拒绝服务攻击等方面的安全性进行测试. 测试通过 Wireshark 工具在同一局域网下进行数据包抓取和通信跟踪来实现，测试用例如表 3 所示.

参照表 3 给出的测试用例，并借助于 Wireshark 工具可以得到如图 12 所示的会话数据流，通过分析表明该远程通信中存在双向证书交换. 对通信报文进一步分析可得到如图 13 所示的结果，该结果表明远程通信中的数据通过加密方式进行传输. 通信过程中的双向身份认证可以保障数据传输的机密性和真实性，数字证书认证可以降低设备遭受拒绝服务攻击的风险.

Table 3 Test Case for Communication Security

表 3 通信安全测试用例

测试项	测试用例	预期测试结果
双向身份认证测试	利用 Wireshark 工具进行通信状态跟踪，查看是否存在证书交换	存在双向数字证书交换报文
数据加密传输	利用 Wireshark 工具记录通信协议状态，查看是否使用 TLS 协议、传输数据是否加密	存在 TLS 协议使用，数据传输为密文传输
欺骗 IP 的拒绝服务攻击	在笔记本电脑上对主 ECU 建立大量数据访问请求链接，查看主 ECU 的响应状态	攻击方因没有证书或证书错误无法完成认证，从而拒绝与其建立 TCP 链接

本文将联盟链技术引入 Uptane 优化框架中，从而可预防因密钥泄露导致的安全问题. 针对该方面的安全测试用例如表 4 所示. 主 ECU 收到更新指令后首先进行区块验证，由于攻击方篡改了文件，导致哈希值验证不通过，因此恶意文件被成功阻止下载.

```
TLSv1   158 Client Hello
TCP      54 30501 → 35806 [ACK] Seq=1 Ack=105 Win=131072 Len=0
TLSv1  1514 Server Hello
TLSv1   433 Certificate, Server Key Exchange, Certificate Request, Server Hello Done
TCP      54 35806 → 30501 [ACK] Seq=105 Ack=1461 Win=64128 Len=0
TCP      54 35806 → 30501 [ACK] Seq=105 Ack=1840 Win=64128 Len=0
TLSv1  1514 Certificate
TLSv1   404 Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
TCP      54 30501 → 35806 [ACK] Seq=1840 Ack=1915 Win=131328 Len=0
TLSv1  1012 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
TCP      54 35806 → 30501 [ACK] Seq=1915 Ack=2798 Win=64128 Len=0
TLSv1   280 Application Data, Application Data
TLSv1   936 Application Data, Application Data
```

Fig. 12 Tracking analysis of the session data flow

图 12 会话数据流跟踪分析

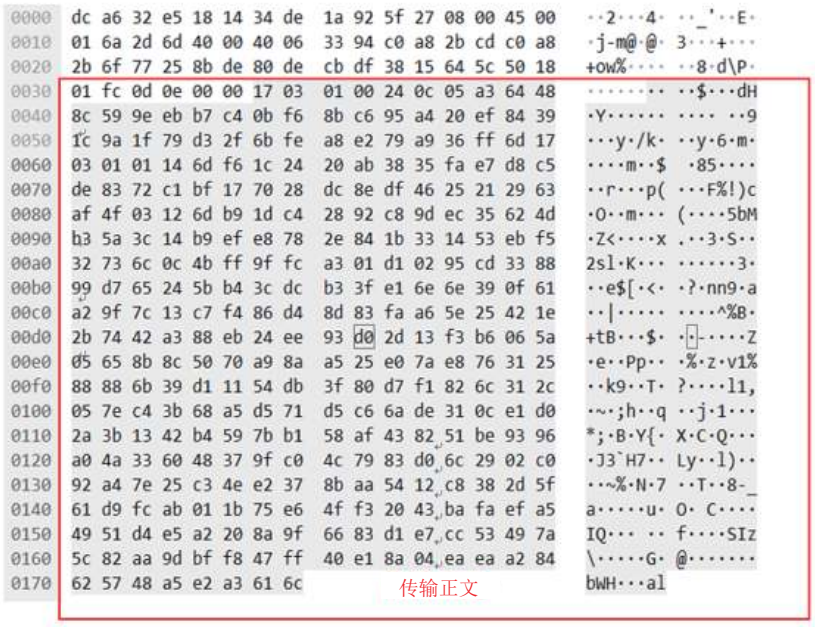

Fig. 13 Analysis of the Wireshark communication message

图 13 Wireshark 通信报文分析

Table 4 Test Cases for Key Leakage

表 4 密钥泄露测试用例

测试项	测试用例	预期测试结果
密钥泄露的中间人攻击	使用镜像库和管理库密钥签署篡改过的文件并下发，查看主 ECU 是否进行下载	区块链验证不通过，主 ECU 未下载恶意文件

4.3 性能分析

本实验采用 Linux 系统监测功能来对比分析 Uptane 优化框架与原 Uptane 框架的内存开销和时延大小. 在树莓派中运行 100 次软件更新任务, 内存开销如图 14 所示. 本文采用设备本地时钟同步检测的方式记录每次框架更新所需的时间, 并对 100 次记录取平均值. 为进一步分析 Uptane 优化框架中节点数量与时延大小的关系, 实验记录了优化框架在更新过程中分别验证 10～100 个节点的更新时间大小. 在上述各种配置下时延对比分析结果如图 15 所示.

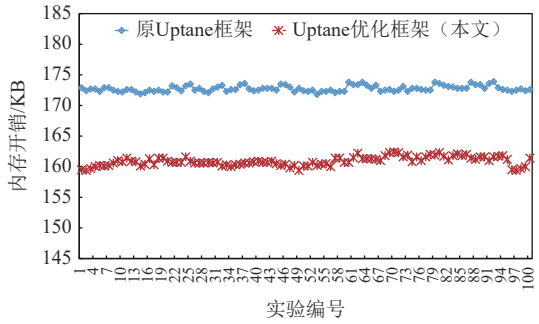

Fig. 14 Comparative analysis of memory consumption

图 14 内存开销的对比分析

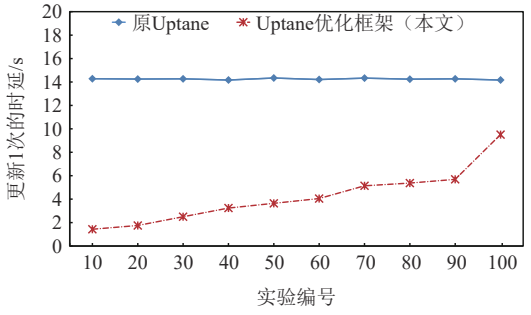

Fig. 15 Comparative analysis of delay

图 15 时延的对比分析

由图 14 可知, Uptane 优化框架和原 Uptane 框架对应的平均内存大小分别为 161 KB 和 173 KB 左右. 因此, 相比原 Uptane 框架, Uptane 优化框架的内存开销降低了 6.9%. 由图 15 可知, Uptane 优化框架和原 Uptane 框架对应的平均时延分别为 10 s 和 14 s 左右. 因此, 相比原 Uptane 框架, Uptane 优化框架的时延开销降低了 28.6%. 原 Uptane 框架采用的多重签名机制带来了较高的内存开销和时延开销, 本文提出采用轻量级的签名算法和哈希算法, 以及基于领域划分的委员会共识机制和车端分区验证的智能合约实现的联盟链技术, 有效降低了 Uptane 优化框架的时延开销和内存开销.

5 结 论

软件在线升级技术在降低成本和提升用户体验等方面的优势使得其在智能网联汽车中将发挥重要作用, 但是如何保障在线升级的安全和高效实现是亟待解决的关键问题. 本文分别从轻量级加密算法选择和引入基于联盟链的验证机制 2 个方面对现有的 Uptane 开源框架进行了优化, 并设计和实现了优化框架的原型系统. 通过测试验证了所提出 Uptane 优化框架的安全性, 并通过与原 Uptane 框架的对比分析可知, 所提出优化框架的内存开销和时延开销分别降低了 6.9% 和 28.6%.

作者贡献声明: 谢勇负责完成系统设计、论文主要内容的撰写; 胡秋燕负责系统实现; 李仁发对论文结构和内容提出指导意见; 谢国琪和肖甫参与讨论并提出了修改意见和实验思路.

参 考 文 献

[1] Kuang Boyu, Li Yuze, Gu Fangming, et al. A review of Internet of vehicle security research: Threats, countermeasures, and future prospects[J]. Journal of Computer Research and Development, 2023, 60(10): 2304−2321(in Chinese)
(况博裕, 李雨泽, 顾芳铭, 等. 车联网安全研究综述: 威胁、对策与未来展望[J]. 计算机研究与发展, 2023, 60(10): 2304−2321)

[2] Xie Yong, Zhou Yu, Xu Jing, et al. Cybersecurity protection on in-vehicle networks for distributed automotive cyber-physical systems: State of the art and future challenges[J]. Software: Practice and Experience, 2021, 51(11): 2108−2127

[3] Halder S, Ghosal A, Conti M. Secure over-the-air software updates in connected vehicles: A survey[J]. Computer Networks, 2020, 178: 107343

[4] Frank S. Where are OTA and in-car data management heading in your car[EB/OL]. 2017[2022-03-15].https://www.iot-now.com/2017/07/05/63657-ota-car-data-management-heading-car/

[5] Ghosal A, Halder S, Conti M. Secure over-the-air software update for connected vehicles[J]. Computer Networks, 2022, 218: 109394

[6] Kuppysamy T K, Delong L A, Cappos J. Securing software updates for automotives using Uptane[J]. Security, 2017, 42(2): 63−67

[7] Linux Foundation. TUF[EB/OL]. 2017[2022-04-12].https://gitee.

com/mirrors/TUF

[8] Asokan N, Nyman T, Rattanavipanon N, et al. Assured: Architecture for secure software update of realistic embedded devices[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2018, 37(11): 2290−2300

[9] Mbakoyiannis D, Tomoutzoglou O, Kornaros G. Secure over-the-air firmware updating for automotive electronic control units[C]//Proc of the 34th ACM/SIGAPP Symp on Applied Computing. New York: ACM, 2019: 174−181

[10] Kornaros G, Tomoutzoglou O, Mbakoyiannis D, et al. Towards holistic secure networking in connected vehicles through securing CAN-bus Communication and firmware-over-the-air updating[J]. Journal of Systems Architecture, 2020, 109: 101761

[11] Plappert C, Fuchs A. Secure and lightweight ECU attestations for resilient over-the-air updates in connected vehicles[C]//Proc of the Annual Computer Security Applications Conf. New York: ACM, 2023: 283−297

[12] Blooshi S A, Han K. A study on employing Uptane for secure software update OTA in drone environment[C/OL]//Proc of the IEEE Int Conf on Omni-Layer Intelligent Systems. Piscataway, NJ: IEEE, 2022[2023-02-13].https://ieeexplore.ieee.org/document/9854983

[13] Guissouma H, Hohl C P, Lesniakl F, et al. Lifecycle management of automotive safety-critical over the air updates: A systems approach[J]. IEEE Access, 2022, 10: 57696−57717

[14] Zhu Guangyu, Zhao Fuquan, Hao Han, et al. Blockchain technology and its application in automotive field[J]. Automotive Engineering, 2021, 43(9): 1278−1284(in Chinese)
(朱光钰, 赵福全, 郝瀚, 等. 区块链及其在汽车领域的应用[J]. 汽车工程, 2021, 43(9): 1278−1284)

[15] Dorri A, Steger M, Kanhere S S, et al. Blockchain: A distributed solution to automotive security and privacy[J]. IEEE Communication Magazine, 2017, 55(12): 119−125

[16] Baza M, Nabil M, Lasla N, et al. Blockchain-based firmware update scheme tailored for autonomous vehicles[C/OL]//Proc of the IEEE Wireless Communications and Networking Conf. Piscataway, NJ: IEEE, 2019[2022-04-19].https://ieeexplore.ieee.org/document/8885769

[17] Stger M, Dorri A, Kanhere S S, et al. Secure wireless automotive software updates using blockchains: A proof of concept[G]//LNMOB: Proc of the Advanced Microsystems for Automotive Applications 2017. Berlin: Springer, 2017: 137−149

Xie Yong, born in 1985. PhD, professor. Senior member of CCF. His main research interests include automotive cyber-physical systems and Internet-of-things.

谢　勇, 1985 年生. 博士, 教授. CCF 高级会员. 主要研究方向为汽车信息物理系统和物联网.

Hu Qiuyan, born in 1996. Master, engineer. Her main research interest includes automotive cyber-physical systems.

胡秋燕, 1996 年生. 硕士, 工程师. 主要研究方向为汽车信息物理系统.

Li Renfa, born in 1956. PhD, professor. His main research interests include automotive cyber-physical systems and computer architecture.

李仁发, 1956 年生. 博士, 教授. 主要研究方向为汽车信息物理系统、计算机体系结构.

Xie Guoqi, born in 1983. PhD, professor. His main research interests include automotive cyber-physical systems and embedded real-time systems.

谢国琪, 1983 年生. 博士, 教授. 主要研究方向为汽车信息物理系统、嵌入式实时系统.

Xiao Fu, born in 1980. PhD, professor. His main research interests include computer network and Internet-of-things.

肖　甫, 1980 年生. 博士, 教授. 主要研究方向为计算机网络、物联网.