计算机科学
COMPUTER SCIENCE

面向云存储的机载软件持有性证明

岳猛, 文程, 洪雪婷, 严思敏

引用本文

岳猛, 文程, 洪雪婷, 严思敏. 面向云存储的机载软件持有性证明[J]. 计算机科学, 2024, 51(11A): 240400040-10.

YUE Meng, WEN Cheng, HONG Xueting, YAN Simin. Airborne Software Provable Data Possession for Cloud Storage [J]. Computer Science, 2024, 51(11A): 240400040-10.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.240400040

面向云存储的机载软件持有性证明

岳 猛[1] 文 程[1] 洪雪婷[2] 严思敏[1]
1 中国民航大学安全科学与工程学院 天津 300300
2 中国移动通信集团安徽有限公司芜湖分公司 安徽 芜湖 241100

摘 要 随着民用航空机载软件数量的不断增长,传统的软件分发方式面临效率低、成本高、安全性差的问题。为了提高机载软件的分发效率,将云存储与机载软件相结合,提出了一种基于 Cloud-P2P 的机载软件存储架构,实现了机载软件的分布式云存储以及机载软件共享。在此基础上,提出了一种数据持有性证明协议,通过将标识与公钥绑定,降低了共谋风险,并通过抽样审计完成对云上机载软件的完整性验证,减小了验证成本。安全性分析表明,所提方案具有不可伪造性和抗重放攻击的能力,并且证明了数据持有性证明协议的正确性。与现有的数据完整性审计方案相比,计算开销减少了 10%,通信开销减少了 20%。该研究对保证机载软件的高效安全分发具有实际意义。

关键词:机载软件;云存储;P2P;数据持有性证明;抽样审计
中图分类号 V328.3

Airborne Software Provable Data Possession for Cloud Storage

YUE Meng[1], WEN Cheng[1], HONG Xueting[2] and YAN Simin[1]
1 School of Safety Science and Engineering, Civil Aviation University of China, Tianjin 300300, China
2 China Mobile Group Anhui Company Limited Wuhu Branch, Wuhu, Anhui 241100, China

Abstract With the increasing number of civil aviation airborne software, the traditional software distribution methods face the problems of low efficiency, high cost and poor security. In order to improve the distribution efficiency of airborne software, we combine cloud storage with airborne software and propose an airborne software storage architecture based on Cloud-P2P, which realizes distributed cloud storage of airborne software and airborne software sharing. On this basis, a provable data possession is proposed, which reduces the risk of complicity by binding the logo to the public key, and completes the integrity verification of the airborne software through sampling audit, reducing the verification cost. Security analysis shows that this scheme is unforgeable and resistant to replay attacks, and proves the correctness of the data-holding proof protocol. Compared with existing data integrity auditing schemes, the computational overhead is reduced by 10% and the communication overhead is reduced by 20%. This research has practical implications for ensuring efficient and secure distribution of airborne software.

Keywords Airborne software, Cloud storage, P2P, Provable data possession, Sampling audit

机载软件作为航空电子系统的核心部件可以控制飞机系统,实现不同的飞行功能[1]。随着新机型的不断推出,机载软件数量正在呈指数型增长,波音 737 和空客 A320 装载了大约 30 个机载软件,波音 777 软件数量已达到 120 多个,波音 787 的软件数量不仅跃升到 500 个,而且这 500 个软件需要加载到 800~900 个位置上[2]。随着机载软件功能和数量的增加,软件的存储与分发变得具有挑战性。第一,机载软件规模的膨胀使得机载硬件系统也随之变得更加复杂和庞大,但机载硬件设备的规模不能与软件数量同比增长。这是因为物理设备的增长会导致飞机重量、燃油消耗、数据实时性、经济成本等问题[3]。第二,机载软件从开发到应用再到维护更新,涉及软件开发商、飞机制造商、飞机运行商等诸多不同的参与方,机载软件经过链条式分发,存在软件不完整,甚至恶意篡改的安全隐患。

近年来,航空电子系统的分布式存储和机载软件的完整性验证技术成为许多研究学者关心的问题之一[4]。Fan 等[4]设计了一种适合航空电子系统数据访问特征的云存储方案,用于读写任务分解,但如何保障云中数据安全并未提及。将大量的机载软件上传至云端虽然缓解了航电系统的存储压力,但软件所有者失去了对机载软件的直接控制,导致机载软件的安全无法得到保障,这也是制约云存储技术在航空领域发展的关键因素。空客公司[5]提出基于安全头文件(Secured

基金项目:国家自然科学基金(621724183);天津市应用基础研究多元投入重点项目(21JCZDJC00830)
This work was supported by the National Natural Science Foundation of China (621724183) and Tianjin Natural Science Foundation (21JCZDJC00830).
通信作者:岳猛(myue_23@163.com)

Header File,SHF)的验证方案,并使用光盘实现对机载软件的存储、传输以及装载。传统的以软盘、U 盘、CD/DVD 为传输媒介的分发方式,不仅需要花费数小时将软件部件加载到多架飞机上的多个可在线更换单元(Line Replaceable Unit,LRU),而且一些传输媒介会因存储不当在未使用时就已经损坏,造成不必要的资源浪费。波音公司[5]同样使用基于安全文件的验证方案,并开发了一种机载软件电子发布与无线传输系统。

针对可能会发生的机载软件丢失、损坏以及被篡改等安全隐患,采用数据持有性证明机制(Provable Data Possession,PDP)对云上机载软件进行完整性验证可以保障软件安全。文献[6-14]提出的 PDP 方案大多采取公钥基础设施(Public Key Infrastructure,PKI),并涉及复杂的密钥管理问题,在验证数据完整性时会大大增加计算开销和通信开销。Gudeme 等[15]提出云存储中共享数据的基于属性的公共审计,简化了密钥管理,但该方案增加了团队经理以及可信机构,在一定程度上增加了通信成本。Yu 等[16]描述了基于属性的云数据完整性审计协议的具体构造,需要用户将属性发送到密钥生成中心生成密钥,存在用户隐私易泄露的问题。Zhang 等[17]提出基于身份的审计方案,减少了复杂的证书管理问题,但身份可能会被伪造。Rehman 等[18]构造了一个新的基于身份的远程数据完整性检查(Remote Data Integrity Checking,RDIC)协议,引入了一种新的高效双重检查策略,但提出的模型使用了更多的模幂运算,存在时间开销大的问题。Yoosuf 等[19]提出用于验证云存储服务器中数据完整性的轻量级双重审计协议,结合公共和私人审计方案以解决审计信任问题,该方案使用 Cramer-Shoup 密码系统加密使得密文大小略高,降低了云存储的效率。

目前机载航空电子系统正向分布式、综合化方向发展,分布式云存储架构为航空电子系统的发展提供了新方向[20-21]。因此,本文提出了一种基于 Cloud-P2P 的分布式机载软件存储架构,将云存储与 P2P 技术相结合,实现机载软件的去中心化云存储以及机载软件共享。为了确保云上机载软件的完整性,解决密钥管理问题以及验证过程中的成本消耗问题,本文提出了一种基于 IPK(Identity Public Key)和抽样审计的数据持有性证明协议。

1 民航机载软件

1.1 机载软件分类

根据 ARINC 规范,目前民航机载软件可以分为两类:硬件控制软件(Hardware Controlled Software,HCS)和飞机控制软件(Aircraft Controlled Software,ACS)。如图 1 所示[22],HCS 和 ACS 的主要区别在于软件是否受硬件控制。HCS 是受硬件控制的软件。HCS 又根据软件是否可加载分为硬件控制可加载软件(Hardware Controlled Loadable Software Part,HCLSP)和常驻软件(Resident Software,RS)。由于 HCS 是与硬件绑定的,因此对于 HCS 的分发,可以遵循硬件分发标准规范。

ACS 是独立的飞机部件,与硬件分开管理,但目前对于 ACS 高效和安全分发的标准和技术尚不完善。此外,ACS 都需要具有可加载属性,因此 ACS 也可以被视为飞机控制可加载软件(Aircraft Controlled Loadable Software Part,ACLSP),ACLSP 安全性会直接影响飞机的运行。本文主要针对 ACLSP 开展研究。

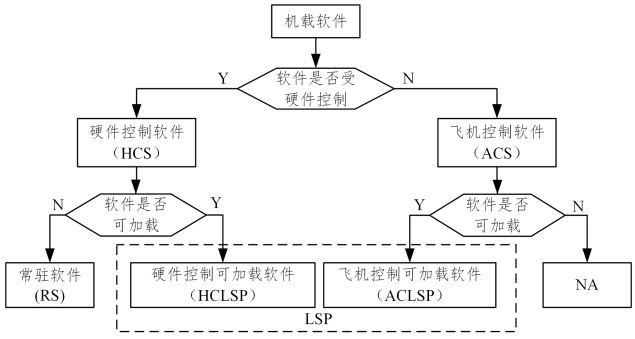


图 1 机载软件分类

Fig. 1 Classification of airborne software

1.2 ACLSP 结构

ARINC665-3 文件[23]定义了 ACLSP 的相关标准,描述了应用于数据加载系统部件的通用原则,包括 ACLSP 的零件编号(Part Number,PN)、内容、标签和格式等。ACLSP 由一个头文件和一个或多个数据文件组成,此外还可能包含支持文件。头文件的扩展名为 LUH,其主要功能是描述 ACLSP 所包含的主要内容,包含 ACLSP 的 PN 号、目标硬件号、校验值、签名算法等基本信息。每一个 ACLSP 拥有唯一的 PN 号,任何时候修改 ACLSP 都应重新分配一个新的唯一的 PN。ACLSP 的 PN 格式应为 MMMCC-SSSS-SSSS,其中 MMM 是分配给每个软件供应商的唯一标识符,CC 是从 PN 中的其他字符生成的两个“检查字符”,SSSS-SSSS 是软件供应商定义的唯一产品标识符。数据文件的扩展名为 LUP,其主要功能是用于存储数据,主要内容由软件供应商决定,可以选择压缩数据文件以节省存储空间以及传输时间,包含基本信息的头文件则不应压缩。支持文件包括后缀名为 LUM 的十六进制媒体集文件以及后缀名为 LUR 的十六进制加载请求文件等。LUM 的主要功能是描述软件加载到 LRU 的方式,如电子加载或利用物理媒介加载。LUR 的主要功能是记录 ACLSP 有关加载请求的信息。

将 ACLSP 加载到相应的 LRU 之前,需要检查 ACLSP 头文件中的配置信息,其中〈KEY〉〈CODE〉〈PART_NUMBER〉等作为 ACLSP 的专属字段,可以唯一地代表该 ACLSP 的相关属性。利用这些字段进行配置检查,以确保软件和目标硬件是兼容的。根据 ARINC 665 标准生成 ACLSP,其头文件的范例如图 2 所示。

头文件中主要配置的信息如下:

〈KEY〉C823CD5851DB〈/KEY〉:软件签名值。

〈CODE〉XYZ〈/CODE〉:供应商标识符。

〈THW_ID〉 THWID1〈/THW_ID〉：目标硬件号。

〈PART_NUMBER〉ABCDEFGH〈/PART_NUMBER〉：软件标识符。

〈LOAD integrity_check＝“MD5”〉：完整性验证算法。

〈USER_DATA_FILE〉path\user_data_fil-ename. extension〈/USER_DATA_FILE〉：数据文件路径以及名称。

〈SUPPORT_FILE〉path\user_data_filena-me. extension〈/SUPPORT_FILE〉：支持文件路径以及名称。

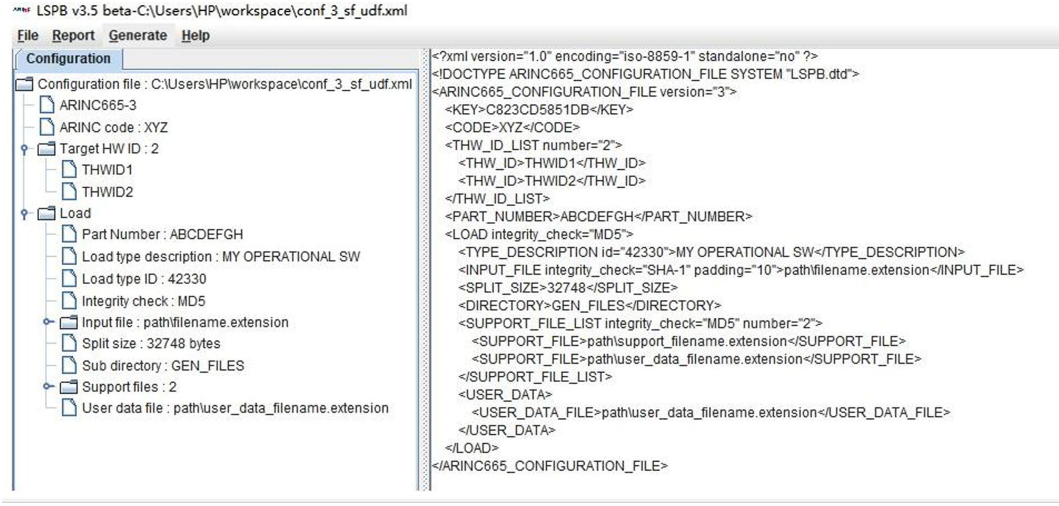


图 2　ACLSP 头文件

Fig. 2　Headerfile of ACLSP

1.3　ACLSP 分发流程

ACLSP 的分发涉及到多个实体，如软件供应商、航空公司以及 LRU 等。ACLSP 的传统分发流程如图 3 所示。

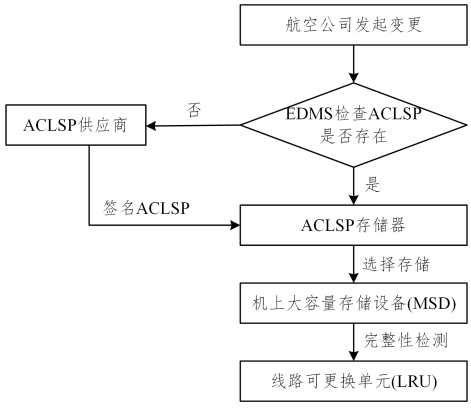


图 3　ACLSP 分发过程

Fig. 3　Distribution process of ACLSP

具体分发步骤如下：

1）确定电子数据管理设备（Electronic Data Management System，EDMS）中授权了新的更改后，将进行检查，以确定航空公司是否已经拥有这些部件。如果没有，则启动新 ACLSP 订单，并发送给相应的供应商。

2）订单由授权的供应商执行，制造满足需求的 ACLSP，并对其进行数字签名后发送给航空公司。

3）航空公司接收带有数字签名的 ACLSP 并检查，将通过检查的 ACLSP 与其数字签名一起存储到安全的 ACLSP 电子库中。

4）根据飞行需要选择相应的 ACLSP 存储在机上的大容量存储设备（Onboard Mass Storage device，MSD）。

5）根据配置文件，在 MSD 中为 LRU 选择合适的 ACLSP 并对其进行完整性验证。

6）最后将通过完整性检测的 ACLSP 加载到相应的 LRU 上。

2　机载软件持有性证明

2.1　基于 Cloud-P2P 的机载软件存储架构

本文根据机载软件的分发流程，以及对存储、共享等方面的需求提出了一种基于 Cloud-P2P 的存储架构，如图 4 所示。首先，由于机载软件数量日益庞大，机上以及地面存储设备面临着巨大压力。运用云存储技术将机载软件上传至云服务器，可以减轻存储压力，减少机上用于存储的机柜数量，从而使得飞机轻量化。其次，不同的机队可能会使用同一机载软件，例如不同的飞机在同一地域内飞行时，都需要该地域的天气报告。运用 P2P 技术形成去中心化的机载软件共享网络，避免了因软件数量庞大而造成中心服务器性能瓶颈的问题。随着更多飞机的加入，整体资源得到了补充，具有一定的可扩展性。每一架飞机作为云中的一个节点，既是用户可以使用机载软件，同时也是存储节点用来存储机载软件。这种存储模式不仅可以减少存储开销而且可以更好地实现机载软件共享，意味着访问者无需花费额外的开销，只需拥有者的授权即可访问机载软件，提高了机载软件的使用效率。

图 4 中，机载软件分发的参与者可以根据所属地域或所属机构设置专属中心云，形成分布式云存储网络。软件发布者将带有数字签名的 ACLSP 分发到中心云进行存储。中心云下层由各地域或机构管理的飞机构成用户扩展云，同一扩展云中的飞机构成 P2P 网络，不同中心云之间也构成 P2P 网络。飞机之间可以实现域内软件共享，也可通过中心云实现跨域共享。该架构中，中心云在 ACLSP 分发流程中承担存储 ACLSP 的责任，即由中心云代替 ACLSP 电子库。用户扩展云中的每一个用户节点相当于 ACLSP 分发流程中的 MSD，用来存储该飞机飞行时需要的 ACLSP。

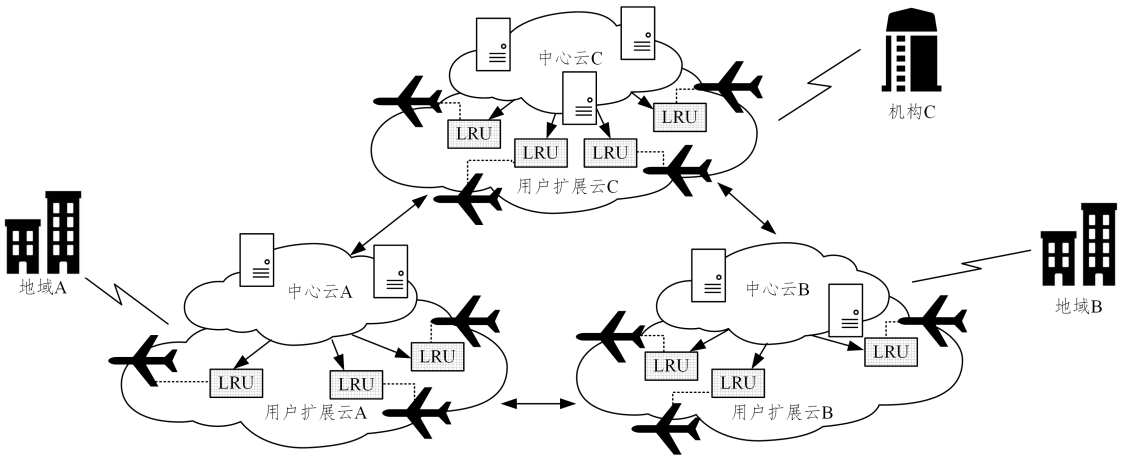


图 4 Cloud-P2P 存储架构

Fig. 4 Cloud-P2P storage architecture for airborne software

Cloud-P2P 存储架构涉及到云与用户、用户与用户以及云与云之间数据的传输，因此需要定义数据访问协议使端与端之间建立数据传输。如图 5 所示，用户首先向中心云发送接入请求，中心云返回唯一的 ID 值，用户加入用户扩展云之后，再向中心云发送数据请求，若中心云存在该数据则返回该数据，若该中心云没有存入相应数据，则返回数据地址，用户再向拥有该数据的其他用户发送数据请求，拥有数据的用户为其返回数据。若该地域中心云和用户都不存在该数据，则可以向其他地域的中心云以及用户发送数据请求。用户接收到所需数据后需进行持有性证明，检查数据是否完整。

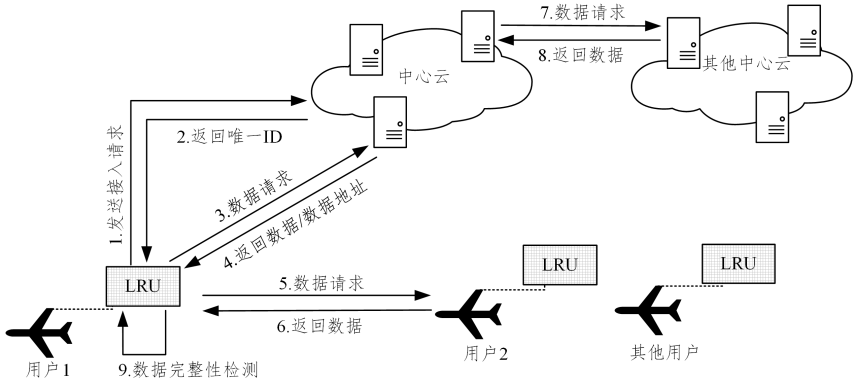


图 5 数据访问协议

Fig. 5 Data access protocol

2.2 基于 IPK 的机载软件持有性证明

持有性证明是一种技术机制，用于确保存储在云上的数据不仅完整无缺，未遭篡改，而且可以随时被数据拥有者访问和检索。这种证明通常涉及加密技术和挑战-响应协议，允许用户或第三方审计者验证数据的完整性，而无需实际访问或下载数据本身。通过这种方式，持有性证明增强了对云存储解决方案的信任，并有助于保护存储在云端数据的安全和隐私。

ACLSP 的分发涉及软件供应商、飞机制造商、飞机运营商、机务维修服务商等多个环节，存在完整性破坏的安全隐患。本文提出一种基于 IPK 的机载软件持有性证明方法，只有通过完整性验证的 ACLSP，才能加载到相应的 LRU 上。

首先根据签名者标识 φ 和 SM2 签名算法计算出公钥 IPK，IPK 的生成过程如图 6 所示。签名者标识 φ 由 ACLSP 头文件中〈PART_NUMBER〉字段、〈CODE〉字段和校验值 CC 构成。其中校验值 CC 由〈PART_NUMBER〉字段和〈CODE〉字段计算得出。

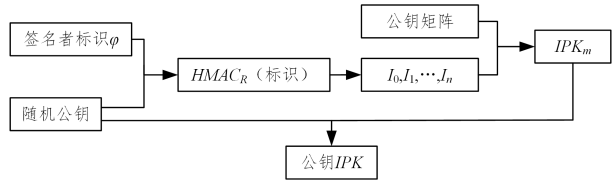


图 6 IPK 的生成

Fig. 6 Generation of the IPK

该方法运用基于标识的 SM2 密钥对生成方法（IPK）生成公钥，与传统的 PKI 证书公钥体系相比，无需证书颁发机构（Certificate Authority，CA）的参与，对标识映射方法进行了彻底的改进，解决了组合公钥（Combined Public Key，CPK）存在的线性共谋风险问题。同时，本文方法还实现了对随机公钥的真实性证明。本文方法的技术实现是将随机公钥（终端自定义公钥＋密钥中心自定义公钥）参与标识映射，实现随机公钥与标识的绑定，从而以标识替代公钥，简化了海量公钥的管理与分发。同时，让终端与密钥中心分别定义随机密钥对，以参与最终的密钥复合。有效解决了标识体系中密钥仅

能在密钥中心生成的问题，实现了私钥只有终端自己所有，密钥中心无法知道终端私钥。

然后对机载软件进行预处理，在机载软件持有性证明过程中需要公共验证参数以及数据标记等变量。基于此，本文采用了一种通用的文件结构，以生成验证时所需要的变量。如图 7 所示，将文件分成 s 块，每个文件块被分为 k 个扇区。文件块和签名标记组成标记对 (m_i,σ_i) 存储在云端。审计过程中，审计者随机生成 (i,v_i) 查询对发送给云端，云端产生响应 $(\mu_1,\mu_2,\cdots,\sigma')$。按照此结构片段将机载软件划分成 $s\times k$ 个扇区，每个块都对应一个标记，该标记由一组随机秘密和代表机载软件属性的 PN 号共同决定，且标记不会随着扇区数 k 的变化而变化，可以有效减少标记的额外存储，提高审计的性能。

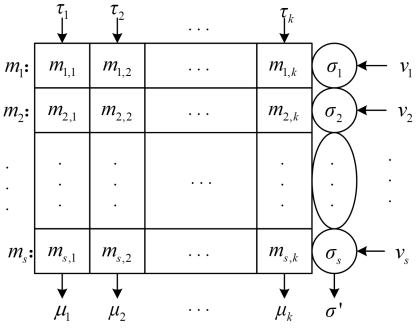


图 7　文件结构

Fig. 7　File architecture

为了判断云中机载软件的完整性，本文提出了一种审计架构，引入了第三方审计者对云中的机载软件进行完整性判断。如图 8 所示，架构主要涉及 3 个主体：软件用户（Software User，SU）、云服务提供商（Cloud Service Provider，CSP）、第三方审计者（Third Party Auditor，TPA）。SU 需要将大量机载软件传输至 CSP 进行存储，CSP 拥有足够的存储空间和计算资源以便提供机载软件存储服务，TPA 在 DO 的授权下管理或监控云中的机载软件，判断云中的机载软件是否完整。

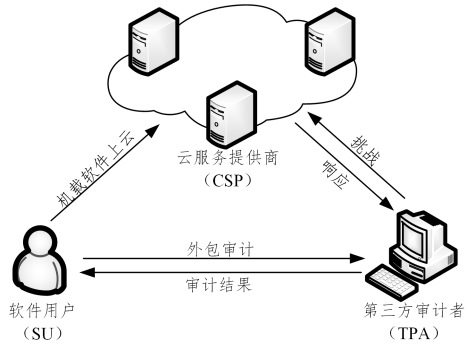


图 8　审计架构

Fig. 8　Audit structure

假设 TPA 可以独立完成审计功能且是可信可靠的。TPA 应在适当的时间间隔内完成对云上机载软件完整性的周期性检测且能够真实地记录各类数据操作，审计过程中无需 SU 的参与，由 TPA 完成对机载软件的管理、监控以及检测。为了确保机载软件的安全，TPA 需要具备抵御恶意攻击的能力，并且需要严格管理控制 TPA，防止未经授权的访问，访问者需要授权才可以查看审计结果并使用云上的机载软件。该审计机制不仅提高了检测机载软件完整性的效率，且减轻了 SU 的存储负担以及审计过程中的计算负担。

假设将所有机载软件交由 CSP 进行存储，并且希望 TPA 维护机载软件安全，TPA 对云上机载软件进行抽样审计，与全样本审计相比大大减少了工作量，基于此本文介绍了一种数据持有性证明协议。协议流程如图 9 所示，首先 SU 计算出公钥/私钥对，然后将公钥 pk 发送给 TPA。然后 SU 随机选取一组秘密 τ，生成公共验证信息 u 和签名标记 σ。最后 SU 将 u 发送给 TPA，将 σ 和机载软件发送给 CSP。在任何时候，TPA 都可以验证云端机载软件的完整性。TPA 和 CSP 之间形成一个交互协议，主要有 3 个部分：承诺、挑战、响应。协议结束之后，TPA 对机载软件的完整性进行检测并返回结果。

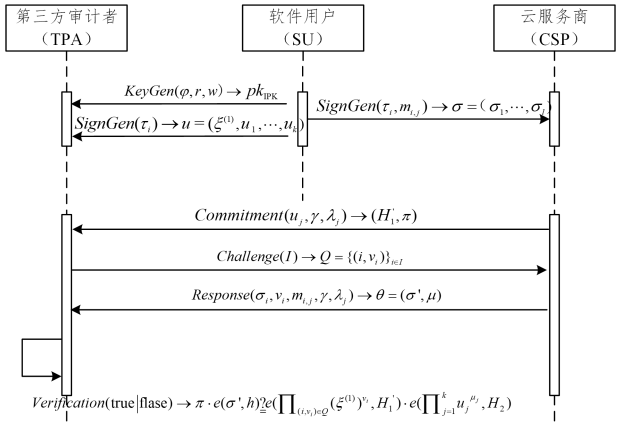


图 9　数据持有性证明协议

Fig. 9　Protocol of provable data possession

协议具体算法的描述如下。

$KeyGen\rightarrow(sk,pk)$：给定一个双线性映射组 $\mathbb{S}=(p,\mathbb{G},G_T,e)$ 和抗碰撞哈希函数 $H_\varepsilon(\cdot)$，选取椭圆曲线 $G(x,y)$，用随机数发生器产生 $\alpha,\beta\in[1,n-1]$，计算 $H_1=h^\alpha$，$H_2=h^\beta\in\mathbb{G}$，计算 SM2 签名值 (r,w)。计算签名者标识 $\varphi=(\mathrm{PART\_NUMBER}\parallel\mathrm{CODE}\parallel CC)$，计算 $pk\leftarrow(\varphi,r,w)$。

$SignGen\rightarrow(u_i,\sigma_i)$：将机载软件 F 划分为 $s\times k$ 个扇区，$F=\{m_{i,j}\}\in\mathbb{Z}_p^{s\times k}$，随机选取一组数 $\tau_1,\tau_2,\cdots,\tau_k\in\mathbb{Z}_p$，计算公共验证参数 $u_i=g^{\tau_i}$，$i\in[1,k]$，设表示机载软件属性的 PN 号为变量 PN_s，计算 $\xi^{(1)}=H_\xi(\text{“}PN_s\text{”})$，计算验证标签 $\sigma_i=(\xi^{(1)})^\alpha\cdot g^{\sum_{j=1}^{k}\tau_j\cdot m_{i,j}\cdot\beta}$。将 $u=(\xi^{(1)},u_1,\cdots,u_k)$ 存储在 TPA 中，将 $\sigma=(\sigma_1,\sigma_2,\cdots,\sigma_s)$ 传输到 CSP。

$Commitment$（CSP→TPA）：CSP 随机选取 $\gamma\in\mathbb{Z}_p$，$\lambda_j\in_R\mathbb{Z}_p$，其中 $j\in[1,k]$，并将承诺 $C=(H_1',\pi)$ 发送至 TPA，其中 $H_1'=H_1^\gamma$，$\pi=e(\sum_{j=1}^{k}u_j^{\lambda_j},H_2)$。

$Challenge$（TPA→CSP）：TPA 选择随机挑战集 I 和随机系数 $v_i\in\mathbb{Z}_p$，TPA 向 CSP 发送挑战系数对集合 $Q=\{(i,v_i)\}_{i\in I}$。

$Response$（CSP→TPA）：CSP 向 TPA 发送响应 $\theta=(\sigma',\mu)$。计算 $\sigma'=\prod_{(i,v_i)\in Q}\sigma_i^{\gamma\cdot v_i}$，$\mu_j=\lambda_j+\gamma\cdot\sum_{(i,v_i)\in Q}v_i\cdot m_{i,j}$，$\mu=\{\mu_j\}_{j\in[1,k]}$。

$Verification\rightarrow(\mathrm{true}\mid\mathrm{flase})$：TPA 计算 $\pi\cdot e(\sigma',h)\stackrel{?}{=}$

$e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i}, H_1') \cdot e(\prod_{j=1}^{k} u_j^{\mu_j}, H_2)$。若等式成立则返回 true 值，说明文件完整，否则反之。

3 安全性分析

3.1 不可伪造性和抗重放攻击

为机载软件提供存储空间的 CSP 并不是完全可信的，CSP 可能期望用已经存在的信息伪造数据持有性证据，甚至直接使用已经过期的数据持有性证据来响应 TPA 的挑战，这对云上机载软件的安全产生了极大的威胁。但本文提出的数据持有性证明协议中，在挑战生成阶段引入了随机挑战集 I 和随机系数 $v_i \in \mathbb{Z}_p$，TPA 每次审计都随机向 CSP 发出挑战系数对，即使是对同样的数据块发起挑战，其所对应的证据也完全不同。因此 CSP 无法对证据进行伪造，且抵挡了抗重放攻击。

安全游戏定义如下：

将不完全可信的 CSP 假定为敌手 A，将 TPA 视为挑战者 C，若敌手 A 能够利用错误的信息生成一个证明，并且该证明可以通过挑战者 C 的验证，那么敌手 A 就赢得该安全游戏，否则反之。

证明：

假定敌手 A 已经获得多组挑战-响应对 $\{\langle Q_i, \theta_i\rangle, i \in N_Q\}$，其中 N_Q 代表已经获得的挑战-响应对个数。

假设挑战者 C 生成的新挑战信息是 $Q_x = \{(i, v_i)\}_{i\in I_x}$，其中挑战集 I_x 可以拆分为 I_{x_1} 和 I_{x_2}，并且敌手 A 已经获得相应的挑战-响应对 $\langle Q_{x_1}, \theta_{x_1}\rangle$ 和 $\langle Q_{x_2}, \theta_{x_2}\rangle$。

挑战者 C 希望获得的响应为 $q_x = (s_x', m_x)$。

假设敌手 A 发送给挑战者 C 的响应为 $\theta_y = (\sigma_y', \mu_y)$，$\sigma_y'$ 满足 $\sigma_y' = \prod_{(i,v_i)\in Q_x} \sigma_i^{\gamma_x \cdot v_i}$。将 $\sigma_i = (\xi^{(1)})^{\alpha} \cdot g^{\sum_{j=1}^{k} \tau_j \cdot m_{i,j} \cdot \beta}$ 带入并变形可得：$g^{\gamma_x \cdot v_i \cdot \sum_{j=1}^{k} \tau_j \cdot m_{i,j} \cdot \beta} = g^{\gamma_{x_1} \cdot v_i \cdot \sum_{j=1}^{k} \tau_j \cdot m_{i,j} \cdot \beta} \cdot g^{\gamma_{x_2} \cdot v_i \cdot \sum_{j=1}^{k} \tau_j \cdot m_{i,j} \cdot \beta}$。

又由 Diffie-Hellman 问题可知，当有 $x, y \in \mathbb{Z}_p$ 且 $g, g^x, g^y \in G_1$，求解 g^{xy} 是困难的。

因此不存在敌手 A 能够以不可忽略的概率由已知信息求得挑战者 C 希望获得的响应证据，因此本文方案具有不可伪造性且可以抗重放攻击，证毕。

3.2 数据持有性证明的正确性

证明数据持有性证明的正确性即验证 $\pi \cdot e(\sigma', h) \stackrel{?}{=} e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i}, H_1') \cdot e(\prod_{j=1}^{k} u_j^{\mu_j}, H_2)$ 是否成立。

证明：

$$
\begin{aligned}
\text{右式} &= e(\prod_{j=1}^{k} u_j^{\mu_j}, H_2) \cdot e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i}, H_1') \\
&= e(\prod_{j=1}^{k} u_j^{(\lambda_j + \gamma \cdot \sum_{(i,v_i)\in Q} v_i \cdot m_{i,j})}, H_2) \cdot e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i}, H_1') \\
&= e(\prod_{j=1}^{k} u_j^{\lambda_j}, H_2) \cdot e(\prod_{j=1}^{k} u_j^{\gamma \cdot \sum_{(i,v_i)\in Q} v_i \cdot m_{i,j}}, h^{\beta}) \cdot e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i}, h^{\alpha \cdot \gamma}) \\
&= e(\prod_{j=1}^{k} u_j^{\lambda_j}, H_2) \cdot e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{\alpha} \cdot g^{\sum_{j=1}^{k} \tau_j \cdot m_{i,j} \cdot \beta}, h) \\
&= e(\sum_{(i,v_i)\in Q} \sigma_i^{\gamma \cdot v_i}, h) \cdot e(\prod_{j=1}^{k} u_j^{\lambda_j}, H_2) \\
&= \pi \cdot e(\sigma', h) = \text{左式}
\end{aligned}
$$

由于本文方案具有不可伪造性和抗重放攻击的特性，因此当 TPA 和 CSP 严格按照本文所提出的数据持有性证明协议进行审计，那么在机载软件没有被篡改以及损坏的情况下，CSP 响应 TPA 挑战的证据就一定能通过数据持有性证明，以验证机载软件的完整性。

3.3 分发方式对比

ARINC835-1 分别介绍了空客和波音保障机载软件安全分发的方案。首先，两者均采用数字证书和数字签名的方式进行机载软件完整性验证，主要验证方式是哈希对比，即利用相同的哈希算法根据接收到的机载软件数据计算出哈希值，并与接收到的哈希值对比，若一致则机载软件是完整的，否则反之。其中数字证书由 PKI 提供，数字签名使用 SHA 算法和 RSA 算法生成。其次，空客利用物理媒介将机载软件分发到航空公司的服务器上进行存储。波音利用物理媒介或电子分发的形式将机载软件分发到航空公司的服务器上进行存储。

将本文提出的机载软件安全分发方案与空客和波音的方案相对比。如表 1 所列，首先，本文方案采用 SM2 算法生成数字签名，相对于 SHA 算法以及 RSA 算法安全强度更高，且密钥长度较低。并且在公钥生成阶段采用 IPK 公钥机制，与 PKI 公钥机制相比，IPK 将密钥与标识绑定，不需要第三方 CA 的参与，降低中间人攻击风险，从而提高公钥安全性。其次，本文方案运用分布式云存储技术，将机载软件电子分发到云端进行存储，相比于物理媒介，电子分发传输时间较短，受环境因素影响较小，传输可靠性和有效性更好。同时，航空公司无需部署大量机柜存储机载软件，因此减少了航空公司的存储开销，减轻了存储压力。最后，本文方案使用第三方抽样审计的方式进行完整性验证。空客、波音采用的哈希对比方式需要对机载软件内所有数据进行哈希计算，而本文方案采用抽样审计只需对部分数据进行持有性证明，从而判断机载软件是否完整，无需对机载软件所有数据进行完整性验证，因此减少了航空公司的计算开销，减轻了计算压力。

表 1 机载软件安全分发方案对比

Table 1 Comparison of airboard software security distribution solutions

	公钥机制	数字签名	传输介质	存储方式	完整性验证方式
空客[5]	PKI	SHA 算法/ RSA 算法	物理媒介	服务器存储	哈希对比
波音[5]	PKI	SHA 算法/ RSA 算法	物理媒介/电子分发	服务器存储	哈希对比
本文方案	IPK	SM2 算法	电子分发	分布式云存储	第三方抽样审计

4 实验分析

本文的实验在配置为 Intel(R) Core(TM) i7-8700 CPU 的主机上进行，使用 Java 语言以及 Python 语言进行编译。实验首先基于 Hadoop 搭建完全分布式云存储架构，利用 VMware Workstation 搭建若干个 Linux 虚拟机，并通过

Xshell 进行远程操控，实现了机载软件的云存储以及不同节点之间的机载软件共享。然后利用 VMware Workstation 搭建 3 个系统为 Ubuntu，内存为 2.5 GB 的虚拟机，分别代表 SU，CSP 以及 TPA，在 charm-crypto 库的基础上采用椭圆曲线，假定 $\mathbb{Z}_p$ 中的一个元素大小 $|p|=100$ bit。接着在不同的实验假设条件下，分别对持有性证明过程中的计算开销以及通信开销进行计算分析，最后与基于身份密码的审计方案[17]和基于身份的 RDIC 方案[18]进行比较分析。基于身份密码的审计方案对文件的预处理与本文方案相似，而基于身份的 RDIC 方案则是数据持有性证明方案中较新的方案，因此本文方案与以上两个方案具有可比性。

4.1　计算开销

本文方案在完成一个验证任务的过程中，TPA 首先生成一些随机数用于构建验证消息，产生随机数的过程仅引入非常小的计算开销。然后在收到验证消息后，CSP 需要计算一个验证证明 $\theta=(\sigma',\mu)$，需花费计算开销 $tExp_{G_1}+tMul_{G_1}+2Mul_{\mathbb{Z}_p}$。最后在验证 $\theta=(\sigma',\mu)$ 正确性时花费的计算开销为 $2kExp_{G_1}+2kMul_{G_1}+tExp_{\mathbb{Z}_p}+tMul_{\mathbb{Z}_p}+4Pa$。将本文方案与基于身份密码的审计方案和基于身份的 RDIC 方案进行比较。如表 2 所列，基于身份密码的审计方案在生成证明时花费的计算开销为 $n(2Exp_{G_1}+Mul_{G_1}+Hash_{\mathbb{Z}_p})$，在验证证明阶段需要花费的计算开销为 $(3+t)Exp_{G_1}+(3+t)Mul_{G_1}+(t+2)Hash_{\mathbb{Z}_p}+2Pa$。基于身份的 RDIC 方案生成证明时需要花费 $(2t+1)Exp_{G_1}+Mul_{G_1}+tHash_{\mathbb{Z}_p}$ 的计算开销，在验证证明阶段需要花费的计算开销为 $Exp_{G_1}+(2t-1)Mul_{G_1}+Exp_{G_2}+2Pa$。

表 2　计算开销的对比

Table 2　Comparison of computational costs

	SignGen()	Verification()
基于身份密码的审计方案[17]	$n(2Exp_{G_1}+Mul_{G_1}+Hash_{\mathbb{Z}_p})$	$(3+t)Exp_{G_1}+(3+t)Mul_{G_1}+(t+2)Hash_{\mathbb{Z}_p}+2Pa$
基于身份的 RDIC 方案[18]	$(2t+1)Exp_{G_1}+Mul_{G_1}+tHash_{\mathbb{Z}_p}$	$Exp_{G_1}+(2t-1)Mul_{G_1}+Exp_{G_2}+2Pa$
本文方案	$tExp_{G_1}+tMul_{G_1}+2Mul_{\mathbb{Z}_p}$	$2kExp_{G_1}+2kMul_{G_1}+tExp_{\mathbb{Z}_p}+tMul_{\mathbb{Z}_p}+4Pa$

表 2 中，Pa 表示 $e=G_1\times G_2\to G_T$ 计算一个线性对的运算时间。Exp·表示在·中计算一个幂指数的运算时间，Mul·表示在·计算一个乘法的运算时间，Hash·表示在·中进行一次哈希计算的时间。本文方案在生成验证证明阶段具有一定优势，与其他两个方案相比，在生成验证证明时无需额外的哈希操作，计算开销有所降低。

4.2　通信开销

本文方案中通信开销主要是由验证消息和验证证明组成。对于每一个验证消息 (i,v_i)，其所需的通信开销为 $t(|p|+|n|)$ 比特，每一个验证证明 $\theta=(\sigma',\mu)$ 的通信开销为 $(k+t)|p|$ 比特。将本文方案与基于身份密码的审计方案和基于身份的 RDIC 方案进行比较，如表 3 所列。

表 3　通信开销的对比

Table 3　Comparison of communication costs

	Challenge()	Response()
基于身份密码的审计方案[17]	$n(\|p\|+\|q\|)$	$2k\|p\|+t\|q\|$
基于身份的 RDIC 方案[18]	$(t+1)\|p\|+t\|n\|+2\|q\|$	$(t+1)\|q\|+\|p\|+t\|id\|$
本文方案	$t(\|p\|+\|n\|)$	$(k+t)\|p\|$

在 TPA 向 CSP 发出挑战的过程中，基于身份密码的审计方案需要花费的通信开销为 $t(|q|+|n|)$，基于身份的 RDIC 方案需要花费的通信开销为 $(t+1)|p|+t|n|+2|q|$。在 CSP 响应阶段，基于身份密码的审计方案花费的通信开销为 $2k|p|+t|q|$，基于身份的 RDIC 方案花费的通信开销为 $(t+1)|p|+|id|$。其中，$|n|$ 代表每一个索引的大小，$|id|$ 代表用户标识的大小，$|p|$ 代表 $\mathbb{Z}_p$ 和 $\mathbb{G}$ 中的元素大小，$|q|$ 代表 $\mathbb{Z}_q$ 中元素的大小。相比其他两个方案，本文方案在 CSP 响应阶段所涉及的元素较少，通信开销有所降低。

4.3　实验结果

本实验在不同的参数下进行性能测试，主要参数有抽样概率、扇区数(k)、机载软件大小以及软件块数量。此外，在同等条件下，计算表 1、表 2 中所有方案的计算开销和通信开销，并进行比较。

首先，针对抽样概率对计算开销以及通信开销的影响进行了如下实验：选取大小为 100 kB 的机载软件，k 为 50，在软件块数量分别为 10，20，30 的情况下，抽样概率从 10% 递增到 50%。实验结果如图 10、图 11 所示，计算和通信开销都随着抽样概率而增加。抽样概率的增加导致挑战-响应对的增加，从而计算和通信开销也在增加。因此当抽样概率达到 100% 即全样本检测时，所花费的计算开销和通信开销比抽样检测的大。

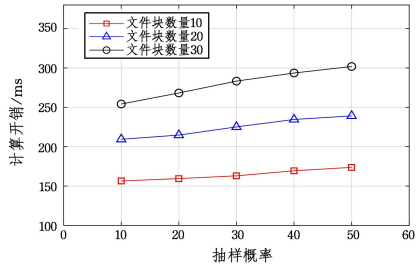


图 10　抽样概率对计算开销的影响

Fig. 10　Impact of sampling probability on computational costs

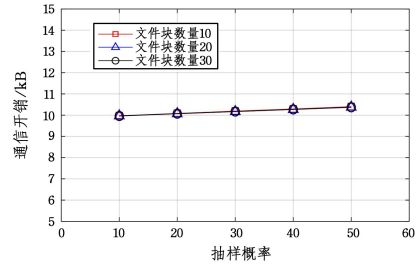


图 11　抽样概率对通信开销的影响

Fig. 11　Impact of sampling probability on communication costs

其次，针对 k 对计算开销以及通信开销的影响进行了如下实验：选取大小为 100 kB 的机载软件，软件块数量为 10，在抽样概率分别为 10%，30%，50% 的情况下，取 k 为 20～100。实验结果如图 12、图 13 所示，计算开销和通信开销都随着 k

的增加而增加。当抽样概率不变，但 k 变大时，TPA 发送的挑战对就会相应地增加，从而导致在验证阶段花费的计算开销增大，并且在 TPA 与 CSP 交互阶段花费的通信开销也会增大。因此，面对不同的抽样概率需要选择合适的 k 对机载软件进行预处理，才能有效减少计算以及通信方面的开销。

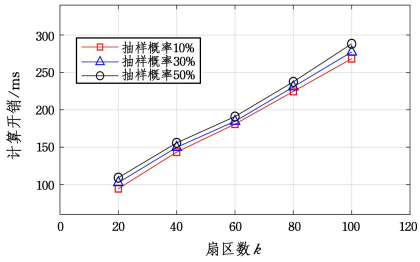


图 12 k 对计算开销的影响

Fig. 12 Impact of k on computational costs

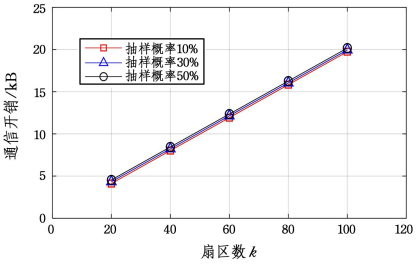


图 13 k 对通信开销的影响

Fig. 13 Impact of k on communication costs

接着，针对机载软件大小对计算开销以及通信开销的影响进行实验。ARINC665 标准列举了头文件中所包含的数据内容以及数据大小，头文件中必须拥有的数据信息大小之和为 1kB。数据文件以及支持文件为十六进制文件，若涉及较大的数据信息，则可在压缩后传输。基于此，选取软件块数量为 10，k 为 50，在抽样概率分别为 10%，30%，50%的情况下，机载软件大小从 10kB 选取到 1000kB。实验结果如图 14、图 15 所示，计算开销几乎不随机载软件大小变化而变化，通信开销则有小幅度上涨趋势。

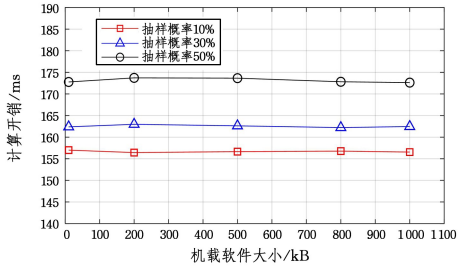


图 14 机载软件大小对计算开销的影响

Fig. 14 Impact of airboard software size on computational costs

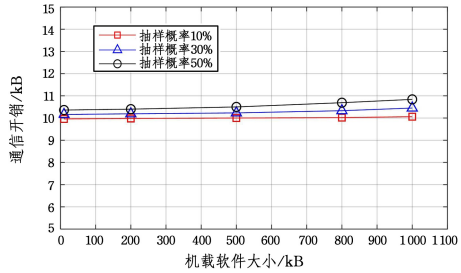


图 15 机载软件大小对通信开销的影响

Fig. 15 Impact of airboard software size on communication costs

对不同大小的机载软件用同样的 k 进行划分，在挑战-响应阶段需要计算的软件块数量相同，因此无论机载软件大小有何变化，其计算开销都不会有上升或下降的趋势。由于机载软件大小不同且 k 相同，因此软件块的大小会随机载软件的增大而增大，故在 TPA 与 CSP 的交互阶段通信开销会有所增加。

然后，针对软件块数量对计算开销以及通信开销的影响进行了如下实验：选取大小为 100kB 的机载软件，k 为 50，在抽样概率分别为 10%，30%，50%的情况下，取软件块数量为 10～50。实验结果如图 16、图 17 所示，计算开销随着软件块数量的增加而增加，通信开销则几乎不随软件块数量的变化而变化。在生成验证证明阶段，需要对每一个软件块进行验证参数的计算，随着软件块数量的增加，计算开销也会随着增加，因此需要根据机载软件大小选择合适的软件块数量对机载软件进行预处理，否则会增加计算开销。但是在 TPA 与 CSP 的交互过程中，由于 k 是一定的，元素的数量以及大小只与 k 有关，并不会随着软件块数量的变化而变化，因此软件块数量对通信开销几乎没有影响。

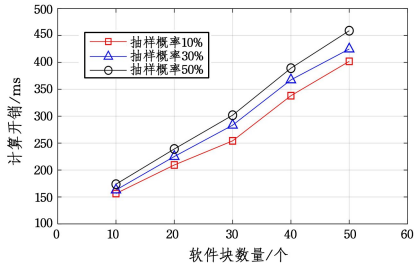


图 16 软件块数量对计算开销的影响

Fig. 16 Impact of the number of software blocks on computational costsCosts

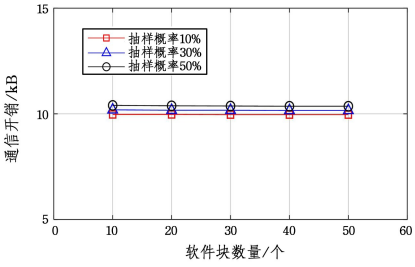


图 17 软件块数量对通信开销的影响

Fig. 17 Impact of the number of software blocks on communication costs

最后，在抽样概率为 30%，机载软件大小为 100kB，k 为 50，软件块数量为 10 的情况下，计算表 1、表 2 中方案的具体开销。计算开销对比如图 18 所示，通信开销对比如图 19 所示。

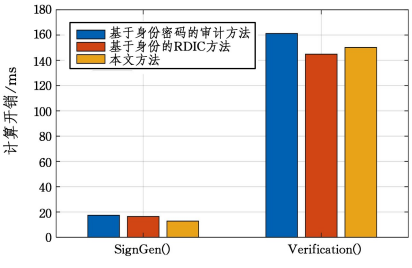


图 18 计算开销对比

Fig. 18 Comparison of computational costs

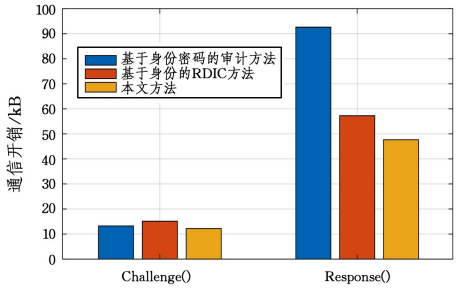


图 19　通信开销对比

Fig. 19　Comparison of communication costs

在同等条件下对 3 种方案进行计算开销和通信开销的对比。在签名和验证阶段，基于身份密码的审计方案所花费的时间大约为 179 ms，基于身份的 RDIC 方案所花费的时间大约为 162 ms，本文方案所花费的时间大约为 161 ms。本文方案相较于基于身份密码的审计方案，在计算开销方面减少了 10%。但由于在验证阶段计算开销比基于身份的 RDIC 方案大，因此在计算开销方面与基于身份的 RDIC 方案基本持平。在挑战-响应阶段，基于身份密码的审计方案所花费的通信开销为 105 kB，基于身份的 RDIC 方案所花费的通信开销为 72 kB，本文方案所花费的通信开销为 59 kB。基于身份的 RDIC 方案比基于身份密码的审计方案在通信开销上减少了约 31%，本文方案在此基础上又减少了 20%的通信开销。综上所述，本文方案降低了数据持有性证明过程中的开销。

结束语　本文针对机载软件分发的安全问题，设计了一种在云存储架构下的数据持有性方法，以确保机载软件的完整性。

1）针对机载软件数量庞大，且存在不同机组共享同一组数据的情况，设计了一种基于 Cloud-P2P 的云存储架构，实现了机载软件云存储以及不同机组之间的软件共享。

2）基于上述架构，本文还提出了一种基于 IPK 的数据持有性协议。运用 IPK 技术将标识与公钥绑定，解决依赖第三方生成密钥的问题，可以有效抵抗中间人攻击。此外，通过抽样审计对云上机载软件进行持有性证明，可以减少审计成本。安全性分析表明，该协议具有正确性、不可伪造性并且可以抗重放攻击。在计算开销和通信开销方面优于现有的持有性证明方法。

将机载软件部署到云架构下使得机载软件的共享和分发更加高效、便捷。可以缓解航电系统在物理上的存储压力，系统的规模也可以适当减小，有助于飞机轻量化设计。而机载软件完整性的保证则更有利于推动机载软件共享分发方式的变革。

参 考 文 献

[1] Best Practices for Loadable Software Management and Configuration Control[R]. International Air Transport Association Engineering and Maintenance Group, 2013: 1-3.

[2] QUAN Y Q. Research on Architecture Design and Safety Analysis of Avionics Architecture Databus Network for Commercial Aircraft[C]// Proceedings of 2018 3rd International Workshop on Materials Engineering and Computer Sciences (IWMECS 2018). 2018: 39-45.

[3] AMARNATH J, SURYA M, BHARGAV P, et al. Cloud computing in Aircraft Data Network[C]// 2011 Integrated Communications, Navigation, and Surveillance Conference Proceedings. Herndon: IEEE Press, 2011: E7-1-E7-8.

[4] FAN C, HAN Z, ZHAO L. Research on Cloud Storage Technology of Avionics System[J]. Electronics Optics & Control, 2022, 29(3): 69-74, 80.

[5] Arinc Report 835-1: Guidance for Security of Loadable Software Parts Using Digital Signatures[S]. ARINC Airlines Electronic Engineering Committee. 2014: 10-29.

[6] LIU J, HUANG K, RONG H, et al. Privacy-Preserving Public Auditing for Regenerating-Code-Based Cloud Storage[J]. IEEE Transactions on Information Forensics and Security, 2015, 10(7): 1513-1528.

[7] BARSOUM A F, HASAN M A. Provable Multicopy Dynamic Data Possession in Cloud Computing Systems[J]. IEEE Transactions on Information Forensics and Security, 2015, 10(3): 485-497.

[8] YU J, REN K, WANG C, et al. Enabling Cloud Storage Auditing with Key-Exposure Resistance[J]. IEEE Transactions on Information Forensics and Security, 2015, 10(6): 1167-1179.

[9] YANG K, JIA X. An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing[J]. IEEE Transactions onParallel and Distributed Systems, 2013, 24(9): 1717-1726.

[10] ZHU Y, HU H, AHN G J, et al. Efficient Audit Service Outsourcing for Data Integrity in Clouds[J]. Journal of Systems and Software, 2012, 865(5): 1083-1095.

[11] WANG J, CHEN X, HUANG X, et al. Verifiable Auditing for Outsourced Database in Cloud Computing[J]. IEEE Transactions on Computers, 2015, 64(11): 3293-3303.

[12] LIU C, RANJAN R, YANG C, et al. MuRDPA: Top-Down Levelled Multi-Replica Merkle Hash Tree Based Secure Public Auditing for Dynamic Big Data Storage on Cloud[J]. IEEE Transactions on Computers, 2015, 64(9): 2609-2622.

[13] WANG H Q. Identity-Based Distributed Provable Data Possession in Multicloud Storage[J]. IEEE Transactions on Services Computing, 2015, 8(2): 328-340.

[14] WANG H Q, WU Q H, QIN B, et al. Identity-Based Remote Data Possession Checking in Public Clouds[J]. IET Information Security, 2014, 8(2): 114-121.

[15] GUDEME J R, PASUPULETI S K, KANDUKURI R. Attribute-Based Public Integrity Auditing for Shared Data with Efficient User Revocation in Cloud Storage[J]. Journal of Ambient Intelligence and Humanized Computing, 2020, 12(2): 2019-2032.

[16] YU Y, LI Y N, YANG B, et al. Attribute-Based Cloud Data Integrity Auditing for Secure Outsourced Storage[J]. IEEE Transactions on Emerging Topics in Computing, 2020, 8(2): 377-390.

[17] ZHANG Y, YU J, HAO R, et al. Enabling Efficient User Revocation inIdentity-Based Cloud Storage Auditing for Shared Big Data[J]. IEEE Transactions on Dependable and Secure Computing, 2020, 17(3): 608-619.

[18] REHMAN A, LIU J, YASIN M Q, et al. Securing Cloud Storage by Remote Data Integrity Check with Secured Key Generation [J]. Chinese Journal of Electronics, 2021, 30(3): 489-499.

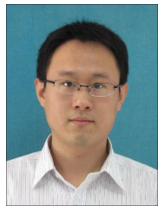
[19] YOOSUF M S, ANITHA R. LDuAP: lightweight dual auditing protocol to verify dataintegrity in cloud storage servers[J]. Journal of Ambient Intelligence and Humanized Computing, 2022, 13: 3787-3805.

[20] LI WH, HAN C, ZHAO Y K, et al. Distributed Integrated Modular Avionics System Architecture Design[C]// Proceedings of the Ninth China Aviation Society Youth Science and Technology Forum. Beijing: China Aviation Publishing & Media CO. 2020: 971-977.

[21] LIU Y, JIN X, WEI X H. System Management Function Design Based on Distributed Avionics System[J]. Electronics Optics & Control, 2022, 29(9): 74-77, 95.

[22] Arinc Report 667-2: Guidance for Management of Field Loadable Software[S]. ARINC Airlines Electronic Engineering Committee. 2017: 9-23.

[23] Arinc Report 665-3: Loadable Software Standards[S]. ARINC Airlines Electronic Engineering Committee. 2005: 6-26.



YUE Meng, born in 1984, Ph.D, professor, Ph.D supervisor, is a member of CCF(No. I8925M). His main research interests include network intrusion detection and defense and civil aviation information security.