计算机科学
COMPUTER SCIENCE

基于可信隐式第三方的机载软件审计方法

岳猛, 朱世博, 洪雪婷, 段冰艳

引用本文

岳猛, 朱世博, 洪雪婷, 段冰艳. 基于可信隐式第三方的机载软件审计方法[J]. 计算机科学, 2024, 51(6A): 230400088-6.

YUE Meng, ZHU Shibo, HONG Xueting, DUAN Bingyan. Airborne Software Audit Method Based on Trusted Implicit Third Party [J]. Computer Science, 2024, 51(6A): 230400088-6.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230400088

基于可信隐式第三方的机载软件审计方法

岳 猛[1] 朱世博[1] 洪雪婷[2] 段冰艳[1]
1 中国民航大学安全科学与工程学院 天津 300300
2 中国民航大学电子信息与自动化学院 天津 300300

摘 要 分布式云存储技术为数量日益庞大的机载软件提供了新的分发与存储方式,这意味着航空公司失去了对软件的直接控制,因此机载软件安全成为了航空公司十分关注的问题。为了提高云存储环境下机载软件的安全性,提出了一种基于可信隐式第三方(Trusted Implicit Third Party,TITP)的机载软件审计方法对云上机载软件进行监控与管理,以确保机载软件的完整性。此外,由部署在云端的可信硬件代替用户进行审计工作,解决了可公开验证审计机制中第三方审计者不完全可信的问题,并以日志的方式记录审计结果以供用户查询。运用可信硬件进行完整性验证不仅降低了用户计算成本,而且缩短了用户在线时间。与其他可信隐式第三方审计方法进行实验对比,所提方法在审计计算过程中节省了10%的时间消耗。

关键词:机载软件;云存储;可信隐式第三方;审计方法;日志
中图分类号 V328.3

Airborne Software Audit Method Based on Trusted Implicit Third Party

YUE Meng[1],ZHU Shibo[1],HONG Xueting[2] and DUAN Bingyan[1]
1 School of Safety Science and Engineering,Civil Aviation University of China,Tianjin 300300,China
2 School of Electronic Information and Automation,Civil Aviation University of China,Tianjin 300300,China

Abstract The distributed cloud storage technology provides a new distribution and storage method for an increasingly large number of airborne software. This means that airlines have lost direct control over the software,therefore the security of airborne software has become one of the most concerned issue of airlines. In order to improve the security of airborne software in the cloud storage environment,an airborne software audit method based on trusted implicit third party is proposed. Trusted hardware deployed in the cloud is used to audit instead of users,which solves the problem that the third party auditor is not completely trusted in the publicly verifiable audit mechanism,and records the audit results in the form of logs for users to query,which not only reduces users' computing costs,but reduces users' online time. Compared with other trusted implicit third party audit methods,it saves 10% of the time consumption in the audit calculation process.

Keywords Airborne software,Cloud storage,Trusted implicit third party,Audit methods,Log

1 引言

现代民用飞机日益自动化和数字化,使得机载软件所包含的数据以及其功能结构变得越来越复杂,其应用范围也变得更加广泛[1]。机载软件在航电系统中的占比日益增大。数量庞大的机载软件不仅增加了航电系统在物理上的存储压力[2],而且在分发以及存储的过程中容易遭受攻击。机载软件的完整性和安全性是安全飞行的保障,然而攻击者对机载软件的篡改严重影响了其完整性和安全性,进而会影响飞机的飞行效率和飞行安全。因此,对机载软件进行完整性验证可以确保加载到飞机上的软件是完整的,从而确保软件可以控制硬件正确地实现航班功能,使飞机按计划正常飞行,降低飞行风险。

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[3-4]。Fan等[5]设计了一种适合航空电子系统数据访问特点的云存储方案,该方案验证了分布式云存储技术在航空领域的可行性。将分布式云存储技术运用到机载软件分发与存储的过程中,可以减轻航电系统在物理上的存储压力,但随之而来的软件安全问题亟待解决。例如,云服务提供商(Colud Service Provider,CSP)可能会因不可抗力因素中断服务,或者为了节约资源而删除部分使用率低的软件,甚至有些CSP与第三方勾结,将软件信息贩卖等。

引入可公开验证的第三方审计机制,对云上机载软件进行持有性证明,以验证机载软件的完整性,通过第三方审计结果可判断云上机载软件是否丢失、损坏或被篡改。但是,将机载软件审计任务外包给第3方审计者,虽然减轻了用户自身的计算负担,但是第三方审计者并不是完全可信的[6],因此机

基金项目:国家自然科学基金(62172418,U1933108,U2133203);天津市自然科学基金(21JCZDJC00830);天津市教委科研计划项目(2019KJ117)
This work was supported by the National Natural Science Foundation of China(62172418,U1933108 U2133203),Tianjin Natural Science Foundation(21JCZDJC00830) and Tianjin Education Commission Research Program Project(2019KJ117).
通信作者:岳猛(myue_23@163.com)

载软件仍然面临着安全威胁。引入第三方审计者主要存在以下3个问题:1)第三方审计者并不完全可信,可能会与云服务提供商勾结,泄露用户隐私或给用户提供虚假验证信息;2)如何确保第三方审计者审计的信息准确无误地传递给软件用户;3)第三方审计者的加入会提高系统部署成本以及运营维护成本。考虑到以上问题以及机载软件的高安全性要求,提出了基于可信隐式第三方的机载软件审计方法,将审计任务部署在云端,在不增加软件用户计算压力的同时,提高了机载软件持有性证明的可靠性。

2 民航机载软件

机载软件是机载计算机各种程序的总称,不仅存储了大量与飞行安全相关的重要数据,而且可以通过发送告警信息协助飞机驾驶员做出合理的判断,甚至可以根据驾驶员的输入或设置控制飞机的飞行,引导飞机以正确的姿态朝着正确的方向执行飞行任务[7]。

由于机载软件的内容以及格式具有一定的特殊性,所以在审计过程中结合软件特性进行持有性证明,可以更有效地保障机载软件的完整性和安全性。如表1所列,ARINC665-3标准[8]规定了一系列后缀名不同的文件,每个文件的后缀名不同,其功能也不同,机载软件则由这些文件构成。

表1 文件后缀名

Table 1 File name suffix

后缀名	主要内容
LUH	Load Upload Header:上传加载头文件,记录软件基本信息
LUP	Load Upload Part(Data File):数据文件,存储数据
LUR	Load Upload Request:上传加载请求,记录软件请求加载信息
LUM	Load Upload Media:上传加载媒体,描述软件传输方式
LUB	Load Upload Batch:批处理文件,批量上传加载
LUI	Load Upload Initialization:加载上传初始化,记录软件初始化信息
LUS	Load Upload Status:上传加载状态,记录软件状态信息

需要注意的是,每个机载软件都必须拥有后缀名为LUH的头文件以及后缀名为LUP的数据文件。后缀名为LUH的头文件主要包括机载软件的PN号、软件签名值、目标硬件号等信息。其中PN号是唯一的,任何时候更新修改机载软件,其对应的PN号都需重新分配。PN号的格式为MMMCC-SSSS-SSSS,其中MMM是制造商代码,是分配给每个开发飞机软件的软件供应商的识别代码;SSSS-SSSS是软件供应商定义的唯一产品标识符;CC是由PN中的其他两个字符生成的"检查字符"。后缀名为LUP的数据文件用于存储数据,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,但包含基本信息的头文件不应被压缩。

3 基于可信隐式第三方的机载软件审计方法

3.1 SM2算法

SM2算法是国产密码算法,包括椭圆曲线公钥加解密算法、椭圆曲线密钥协商算法、椭圆曲线数字签名算法等[9]。本文主要研究椭圆曲线数字签名算法,在使用SM2算法之前需要进行相关参数的约定,包括有限域、椭圆曲线E及基点$G=(x_0,y_0)$。签名算法如下:

1)产生密钥

步骤1 产生一个随机数d作为私钥;

步骤2 计算公钥$P=[d]G$;

2)签名生成

步骤1 生成随机数k,计算$(x_1,y_1)=kG$;

步骤2 计算$r=(e+x_1) \bmod N$,其中$e=H_v(m)$,m为待签名的消息;

步骤3 计算$s=(1+d)^{-1}(k-r*d)$,并将r和s作为签名结果。

3)签名验证

步骤1 签名的接收者收到r,s和消息m,计算$(x_1',y_1')=sG+(r+s)P$;

步骤2 计算判断签名值是否相等,如果相等则签名验证通过,否则签名验证失效。

3.2 可信隐式第三方审计架构

可信隐式第三方是软件用户(Software User,SU)部署在云上的可信硬件[10],由云服务提供商运行维护。在系统部署之前,将密钥、公共验证参数、状态时钟以及相关程序载入TITP中,载入后外界(包括CSP)无法获取或修改。TITP有特定的输入接口用于接收数据,并且能够对有关数据进行数学运算,运算结果通过特定的输出接口输出。此外,TITP具有自我保护机制,当TITP检测到外部物理攻击时,会启动自我销毁程序[10],防止用户隐私泄露。

可信隐式第三方审计架构如图1所示,主要由SU,CSP,TITP组成,在物理位置上主要分为用户端和云端,在逻辑信任关系上分为可信域和不可信域。SU为用户端;TITP集成在CSP中,与CSP一同部署在云端。

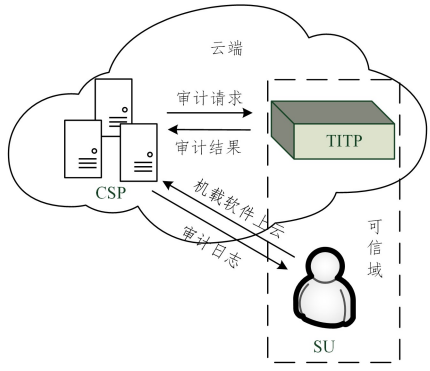


图1 可信隐式第三方审计架构

Fig. 1 Trusted implicit third party audit architecture

TITP包含有关软件的信息,并且外界无法获取,是完全可信的。因此,TITP与SU构成可信域,审计架构中唯一危险来源CSP构成不可信域。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用于提供机载软件存储服务并且周期性地向TITP发送审计请求;TITP需要在规定的时间内完成对云端机载软件的持有性证明,并且真实地记录审计结果,审计过程无需SU参与,审计结果由CSP以日志的方式记录,SU可不定期向CSP申请查看日志并使用云端机载软件。

3.3 审计日志

引入TITP帮助SU进行机载软件持有性证明,可以缩短SU在存储和计算方面的开销,同时以日志的方式记录审计结果可以减少SU在线时间。每完成一次审计,会生成一个日志项(Log Entry,LE)。LE的字段组成如表2所列。

表 2　LE 组成

Table 2　Composition of LE

字段名	描述
Tid	当前日志项的唯一标识
Pre_Tid	同日志内上一日志项的标识
Time	审计时间
Result	机载软件持有性证明结果，软件完整为“1”，不完整为“0”
Uid	用户标识
Cid	CSP 标识
Utype	用户对机载软件的操作类型
Sign	对日志进行签名

同一日志的日志项组成日志链表，项与项之间通过“Pre_Tid”字段连接。同时引入项引用(Entry Reference，ER)作为链表头，此数据结构可以解决 CSP 丢弃新产生的日志项的问题，从而保证日志的完整性。图 2 为审计日志链示意图。

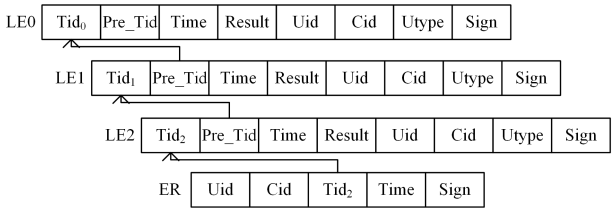


图 2　审计日志链

Fig. 2　Audit log chains

3.4　机载软件持有性证明协议

三方交互协议涉及 SU，CSP，TITP 3 个实体，协议主要包括注册阶段、创建日志阶段、审计阶段以及阅读阶段，如图 3 所示。其中，在注册阶段运用了无需第三方的基于标识的 SM2 密钥对生成方法(Identity Public Key，IPK)，该方法将标识与密钥绑定，实现了去第三方的密钥对生成，降低了中间人攻击的风险。

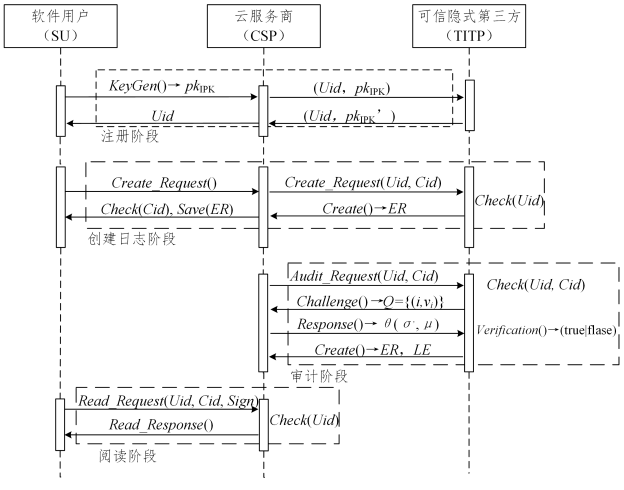


图 3　三方交互协议

Fig. 3　Tripartite interaction protocol

注册阶段：在用户加入时执行，SU 首先运用 *KeyGen*() 计算 SM2 签名值(r,w)以及签名者标识 $\varphi=$(PART_NUMBER‖CODE‖|CC)，再由(r,w)和 φ 生成公钥 *IPK* 发送给 CSP，CSP 为用户生成唯一的 ID 号 *Uid*。CSP 将(Uid,pk_{IPK})存储并发送给 TITP，TITP 存储(Uid,pk_{IPK}')，并用私钥对其签名后发送给 SU，SU 接收后利用公钥解析，确认 TITP 存储的 pk_{IPK}'是否与本地存储的一致。

创建日志阶段：由 SU 向 CSP 发出创建申请 *Create_Request*()，CSP 为用户的创建请求生成 *Cid*，再将(Uid,Cid)发送给 TITP，TITP 对 SU 的身份 *Uid* 进行验证，验证通过后 TITP 进行日志审计的相关工作。TITP 用 *Create*() 生成 *ER*，*ER.Uid*←*Uid*，*ER.Cid*←*Cid*，*ER.Sign*←NULL。由于尚未进行审计工作，所以 *ER.Time* 为 NULL，*ER.Tid* 为 −1。*ER* 创建完成后由 CSP 发送给 SU，SU 对 *Cid* 进行验证，验证完成后将 *ER* 存储在本地。

审计阶段：CSP 以 *Uid* 和 *Cid* 为参数向 TITP 发出审计申请 *Audit_Request*()，TITP 接收申请之后，先对 *Uid* 和 *Cid* 进行验证，验证通过后开始执行审计任务。首先，TITP 执行 *Challenge*() 算法，选择随机挑战集 I 和随机系数 $v_i\in\mathbb{Z}_p$，向 CSP 发送挑战系数对集合 $Q=\{(i,v_i)\}_{i\in I}$。接着 CSP 对 TITP 发出的挑战进行响应，执行 *Response*()，计算 $\sigma'=\prod_{(i,v_i)\in Q}\sigma_i^{\gamma\cdot v_i}$，$\mu_j=\lambda_j+\gamma\cdot\sum_{(i,v_i)\in Q}v_i\cdot m_{i,j}$，$\mu=\{\mu_j\}_{j\in[1,k]}$，计算完成后将响应 $\theta=(\sigma',\mu)$发送至 TITP。然后 TITP 对接收到的响应进行持有性证明，执行 *Verification*()，计算 $\pi\cdot e(\sigma',h)\overset{?}{=}e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i},H_1')\cdot e(\prod_{j=1}^{k}u_j^{\mu_j},H_2)$，若等式成立则返回 true 值，说明机载软件完整，否则不完整。最后根据审计结果用 *Create*() 创建审计日志，*Verification*() 返回值若为 true，则 *LE.Result*←1，若为 false，则 *LE.Result*←0。*LE.Time*←*Get_Time*() 获取日志生成时间，*LE.Pre_Tid*←*ER.Tid*，以 *LE.Result*，*LE.Time*，*LE.Pre_Tid*，*LE.Tid* 为参数，用 *Sign*() 计算出 *LE.Sign*，*ER.Time*←*LE.Time*，*ER.Tid*←*LE.Tid*，以 *ER.Time*，*ER.Tid*，*ER.Uid*，*ER.Cid* 为参数用 *Sign*() 计算出 *ER.Sign*。TITP 将生成的(ER,LE)对发送至 CSP 进行存储。

阅读阶段：SU 以 *Uid*，*Cid*，*Sign* 为参数向 CSP 发送阅读日志的请求 *Read_Request*()，CSP 接受请求并检查 *Uid* 是否正确，若正确则执行 *Read_Response*() 向 SU 返回其需要查询的日志。

三方交互协议结束后，SU 可根据查询到的日志判断机载软件在云存储期间的完整性。

4　安全性分析

4.1　机载软件持有性证明的正确性

在审计阶段，CSP 定期向 TITP 发送审计请求，TITP 验证 CSP 身份后发起挑战，CSP 根据挑战计算响应并发送给 TITP，最后 TITP 对响应进行验证并记录。其中，在验证阶段执行 *Verification*()，即计算 $\pi\cdot e(\sigma',h)\overset{?}{=}e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i},H_1')\cdot e(\prod_{j=1}^{k}u_j^{\mu_j},H_2)$。对该式进行推导，若该式成立，说明机载软件持有性证明是正确的，则完整性挑战成功。

证明：在 $\pi\cdot e(\sigma',h)\overset{?}{=}e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{v_i},H_1')\cdot e(\prod_{j=1}^{k}u_j^{\mu_j},H_2)$中：

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

$$=e(\prod_{j=1}^{k}u_j^{\lambda_j},H_2,H_2)\cdot e(\sum_{(i,v_i)\in Q}(\xi^{(1)})^{\alpha}\cdot g^{\sum_{j=1}^{k}\tau_j\cdot m_{i,j}\cdot\beta},h)$$

$$=e(\sum_{(i,v_i)\in Q}\sigma_i^{\gamma\cdot v_i},h)\cdot e(\prod_{j=1}^{k}u_j^{\lambda_j},H_2,H_2)$$

$$=\pi\cdot e(\sigma',h)$$

$$=\text{左式}$$

由证明可知，用于验证阶段的公式成立，所以本文提出的机载软件持有性证明协议是正确的。若 SU，CSP 和 TITP 严格按照该协议进行交互，则可以正确记录机载软件完整性验证的结果。

4.2 不可伪造性和抗重放攻击

在审计阶段运用了 IPK 机制，其安全性主要依托于算法所涉及到的数学难题——Diffie-Hellman 问题，即当 $x,y\in\mathbb{Z}_p$ 且 $g,g^x,g^y\in G_1$ 时，求解 g^{xy} 是困难的。基于此，设计了如下安全游戏。

定义：攻击者为不完全可信的 CSP，挑战者为 TITP，若攻击者利用错误信息生成的响应可以通过挑战者的验证，那么就称攻击者赢得了该游戏，否则失败。

证明：假设攻击者已获得多个挑战-响应对$\{\langle Q_i,\theta_i\rangle,i\in N_Q\}$，其中 N_Q 表示已获得挑战-响应的数量。

假设 $Q_x=\{(i,v_i)\}_{i\in I}$ 是挑战者生成的新挑战信息，其中挑战集 I_x 可拆分为 I_{x_1} 和 I_{x_2}，并且攻击者已经获得相应的挑战-响应对$\langle Q_{x_1},\theta_{x_1}\rangle$和$\langle Q_{x_2},\theta_{x_2}\rangle$。

挑战者希望获得的响应为 $\theta_x=(\sigma_x',\mu_x)$。

假设攻击者发送给挑战者的响应为 $\theta_y=(\sigma_y',\mu_y)$，$\sigma_y'$满足$\sigma_y'=\prod_{(i,v_i)\in Q_x}\sigma_i^{\gamma_x\cdot v_i}$。将 $\sigma_i=(\xi^{(1)})^{\alpha}\cdot g^{\sum_{j=1}^{s}\tau_j\cdot m_{i,j}\cdot\beta}$代入并变形可得：$g^{\gamma_x\cdot v_i\cdot\sum_{j=1}^{s}\tau_j\cdot m_{i,j}\cdot\beta}=g^{\gamma_{x_1}\cdot v_i\cdot\sum_{j=1}^{s}\tau_j\cdot m_{i,j}\cdot\beta}\cdot g^{\gamma_{x_2}\cdot v_i\cdot\sum_{j=1}^{s}\tau_j\cdot m_{i,j}\cdot\beta}$。由 Diffie-Hellman 问题知，该式求解困难。

因此，攻击者无法赢得该安全游戏，所以本文提出的审计方法具有不可伪造性。

此外，审计阶段 TITP 执行 *Challenge*()算法时，每次发出的挑战对都是随机的，即使对同一随机块发起多次挑战，其返回的响应也是不同的。因此，本文提出的审计方法具有抗重放攻击的能力。

4.3 审计日志显篡改能力

审计日志记录每一次持有性证明的结果，以便 SU 在使用软件前对机载软件的完整性进行验证。审计日志的正确性直接影响 SU 的判断。对审计日志的安全性进行分析，首先给出如下 4 个定义。

定义 1 项引用 *ER* 的可验证性。

ER 是可验证的当且仅当：1）*ER* 是崭新的，即 *ER*. *Time* 所在的 epoch 距离当前时间最近；2）文件的相关信息可以被 *ER* 中的 *Uid* 真实地描述；3）*ER*. *Sign* 可被 TITP 的 *IPK* 公钥验证。

定义 2 日志项 *LE* 的可信性。

LE 是可信的当且仅当利用 TITP 所持有的 IPK 公钥能验证 LE. Sign。

定义 3 签名者标识 φ 的可信性。

签名者标识 φ 是可信的当且仅当：1）签名者标 φ 可以正确地描述某一机载软件；2）利用 SU 的公钥可以验证软件中的签名。

定义 4 审计日志历史的一致性。

某个机载软件的审计历史是一致的当且仅当：1）*ER* 具有可验证性；2）*ER*. *Tid* 连接的 *LE* 是存在且可信的；3）日志内每一项 *LE* 的 *LE*. *Prev_Tid* 都不为空且连接的 *LE* 都是存在且可信的[10]；4）签名者标识 φ 是可信的。

定理 1 当 CSP 与 TITP 严格按照本文设计的交互协议进行审计工作时，则针对某一机载软件的审计日志历史一定是一致的。

证明：根据三方交互协议审计过程可知，TITP 接收到 CSP 的审计请求时会对其身份信息进行验证，验证通过才可以进行周期性的审计以及日志的创建。假设日志在生成过程中被篡改，攻击者篡改日志的行为主要有以下 3 种：1）假设攻击者对签名者标识 φ 或临近 epoch 内的 ER 进行篡改，攻击者因为无法掌握 SU 与 TITP 的私钥，所以无法伪造软件的签名和 *ER*. *Sign*，因此当二者的任何一个字段被修改时必将导致二者的不可验证和不可信；2）假设 *LE* 中的任何一个字段被攻击者篡改，由于攻击者无法掌握 TITP 的私钥，所以其 *LE* 的任何修改都会导致 *LE*. *Sign* 无法通过验证，从而导致 *LE* 不可信；3）假设攻击者丢弃 *ER* 或日志链中任何一个 *LE*，都将导致日志链表的断裂[11]。结合定义 1－定义 4 易知，当 CSP 与 TITP 按照交互协议正确地进行交互时，审计日志历史一定是一致的。因此，审计日志具有显篡改的能力。

此外，可信硬件的相关参数在系统部署前已经载入，包含 CSP 在内的所有外界用户都无法修改。综上所述，本文提出的基于可信隐式第三方的机载软件审计方法中的第三方隐式硬件是可信的。

5 实验结果与分析

5.1 实验环境

实验的设备是配置为 Intel(R)Core(TM)i7-8700 CPU 的台式工作站。首先基于 Hadoop 搭建分布式云存储架构，利用 VMware Workstation 搭建若干个 Linux 虚拟机，并通过 Xshell 进行远程操控。然后利用 VMware Workstation 搭建 3 个系统为 Ubuntu，内存为 2.5 GB 的虚拟机，其分别代表 SU、CSP 以及 TITP。用程序模拟可信硬件的计算速度比实际可信硬件计算速度快，所以引入了减速因子以便更加真实地还原隐式可信硬件的特性。IBM4764 安全协处理器每秒钟能产生 2.16 个长度为 1024 bit 的 RSA 密钥对[12]，对比在测试平台获得的数据(14.92 个/秒)，可算得可信硬件对大数运算的减速因子为 6.91[12]。

5.2 计算开销

TITP 在审计交互过程中参与了生成挑战阶段、验证响应阶段以及生成日志阶段，并在这 3 个阶段产生了一定的计算开销。在相同的实验条件下，以挑战块数量为自变量，计算开销为因变量，观察两者之间的关系。选取大小为 100 kB 的机载软件，在扇区数 k 为 50，软件块数量为 30 的情况下，挑战块数量从 10 逐渐递增到 500。实验结果如图 4 所示，在生成挑战阶段以及验证响应阶段，计算开销均随着挑战块数量的增加而增加。因此，在生成挑战和验证响应阶段，计算开销主要取决于挑战块的数量，这是由于随着挑战块数量的增加，在挑战以及验证响应阶段需要计算的数据增多了，所以花费了更多的计算开销。

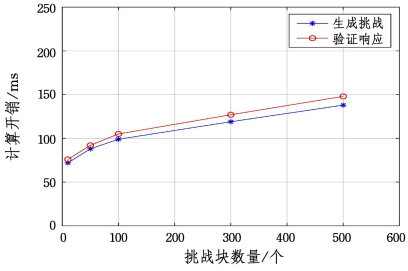


图 4　挑战块数量在挑战-响应阶段的计算开销

Fig. 4　Calculation costs of the number of challenge blocks in challenge-response phase

在日志生成阶段，计算开销与挑战块数量的关系如图 5 所示。可以看出，无论挑战块数量为多少，计算开销都没有变化。这是因为日志记录的项数是固定的，数据的增多不会导致需要记录的数据变多。因此，TITP 生成审计日志的计算开销是恒定的。

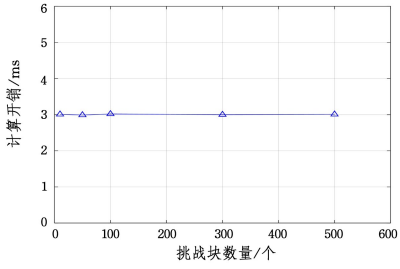


图 5　挑战块数量在日志生成阶段的计算开销

Fig. 5　Calculation costs of the number of challenge blocks in log generation phase

5.3　通信开销

审计过程中产生的通信开销主要来自 CSP 与 TITP 之间的交互。如表 3 所列，在审计请求阶段、生成挑战阶段、验证响应阶段以及生成日志阶段都会产生一定的通信开销。其中 $|\cdot|$ 为生成相应参数时所花费的通信开销，n 为随机抽取的挑战块数量。生成挑战阶段的通信开销随着 n 的增大逐渐增大，但其他阶段的通信开销受 n 的影响较小，因此除生成挑战阶段外，其他阶段的通信开销几乎是恒定的。

表 3　TITP 的通信开销

Table 3　Communication costs of TITP

产生原因	通信开销
审计请求	$\|n\|+\|Name\|+\|Sign\|$
生成挑战	$n\cdot(\|i\|+\|v_i\|)n$
验证响应	$\|\theta\|=\|\sigma'\|+\|\mu\|,\|ER\|=\|Uid\|+\|Cid\|+\|Tid\|+\|Sign\|$
生成日志	$\|LE\|=\|Rueslt\|+\|Time\|+\|Tid\|+\|Prev\_Tid\|+\|Sign\|$ $\|ER\|=\|Uid\|+\|Cid\|+\|Tid\|+\|Sign\|$

5.4　方法对比

本方法与可公开验证的第三方审计方法主要有以下几点区别。

1）引入了可信隐式第三方审计机制，解决了可公开验证审计机制中第三方不完全可信的问题。

2）可信隐式硬件由 SU 部署在云端，降低了系统布署的复杂度以及系统的运行和维护的成本。

3）审计日志具有显篡改的功能，可以准确地记录每一次持有性证明的结果。

4）利用可信硬件可能会带来更高的计算开销和通信开销，但在生成日志阶段所花费的计算开销是恒定的，且只有 3 ms 左右，因此对整个架构的计算开销影响不大。在 CSP 与 TITP 交互过程中生成审计日志所花费的通信开销最大，但由于其不随挑战块数量 n 的增加而增高，是一个定值，所以其对架构整体的通信开销影响不大。

下面将所提方法与现有的基于可信隐式第三方的数据持有性证明方法进行比较。文献[11]和文献[13]都采用了基于 PRF 的数据持有性证明，且分别与基于 BLS 的数据持有性证明和基于 RSA 的数据持有性证明进行了审计计算时间的比较。选取大小为 100 kB 的机载软件，扇区数 k 为 50，软件块数量为 30，挑战块数量为 10 个、50 个、100 个、300 个、500 个，分别对基于 BLS 的数据持有性证明、基于 RSA 的数据持有性证明、基于 PRF 的数据持有性证明、基于 IPK 的数据持有性证明进行实验，记录了响应验证时间。由于引入了可信硬件代替用户进行数据持有性证明，所以在审计过程中 TITP 的时间消耗主要存在于生成挑战、验证响应以及生成日志阶段。首先由于生成挑战阶段未涉及签名算法，所以 4 种方法在生成挑战阶段花费的时间相同，但随着挑战块数量的增加，挑战时间也在不断变长，这是因为需要计算的数据变多，所以计算开销有所增高。接着在日志生成阶段，同挑战阶段一样不涉及签名算法，因此对 4 种方法使用了相同的日志生成方法，消耗的计算时间不随挑战块数量的增加而变长，大约在 3 ms 左右，这是因为日志生成的项是固定的，不会因为挑战块的增加而需要多记录数据。最后在验证响应阶段，如表 4 所列，由于所使用的签名算法不同，所以花费的计算时间也不相同。

表 4　审计计算时间对比

Table 4　Comparison of audit calculation time

(ms)

挑战块数量/个	BLS 响应验证[13]	RSA 响应验证[11]	PRF 响应验证[11-13]	IPK 响应验证
10	2934	96.38	77.02	69.35
50	2948	123.45	98.56	88.64
100	3194	146.25	116.67	105.34
300	4563	177.54	142.30	126.48
500	7259	206.36	164.92	149.27

从实验数据可以得出，BLS 响应验证时间最长，相比其他 3 种方法多了 1～2 个数量级，RSA 响应验证时间较 BLS 有了大幅度的缩短，PRF 响应验证时间在 RSA 的基础上缩短了 20%，IPK 响应验证时间又在 PRF 的基础上缩短了 10%。由上述实验对比分析可知，本文提出的审计方法相较于其他方法，在审计计算时间上缩短了 10%，提高了审计效率。

结束语　本文提出了基于可信隐式第三方的机载软件审计方法。首先，引入了可信隐式硬件，将硬件部署在云端进行审计工作，有效解决了第三方审计者的信任问题。为了准确记录持有性证明的结果，采用审计日志对证明结果进行记录，并设计了一种机载软件持有性证明协议用于 SU，CSP 以及 TITP 三者之间的交互。接着，对该方法进行了安全性分析，证明了机载软件持有性证明协议的正确性，并且证明了该方法具有不可伪造性和抗重放攻击的能力。此外，本方法采用的审计日志具有显篡改的能力。最后，对本方法的计算开销以及通信开销进行分析，并与其他可信隐式第三方审计方法

进行比较。本方法运用了 IPK 机制进行持有性证明以确保机载软件的完整性，在可靠性上具有一定的优势，并且在审计计算时间上减少了 10%。

参 考 文 献

[1] 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.

[2] 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.

[3] LI W H, HAN C, ZHAO Y K, et al. Research on Architecture Designof Distributed Integrated Modular Avionics System[C]// Proceedings of the 9th China Aviation Society Youth Science and Technology Forum. Beijing: China Aviation Publishing & Media CO., 2020: 971-977.

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

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

[6] ZHOU R, HE M X, CHEN Z M. Certificateless Public Auditing Scheme with Data Privacy Preserving for Cloud Storage[C]// 2021 IEEE 6th International Conference on Cloud Computing and Big Data Analytics. Chengdu, China: IEEE, 2021: 675-682.

[7] LUO L P, SUN W. Exploration of Airborne Software Management Scheme for Airlines[J]. Aviation Maintenance & Engineering, 2016(9): 110-111.

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

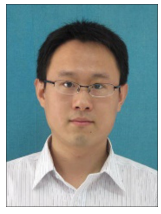
[9] 信息安全技术 SM2 椭圆曲线公钥密码算法(第 2 部分)：数字签名算法：GB/T 32918.2-2016[S].

[10] XIAO D, YANG L Y, SUN B, et al. Provable Data Possession System for Realistic Cloud Storage Environments[J]. Journal of Software, 2016, 27(9): 2400-2413.

[11] AN B Y, GONG Z, XIAO D, et al. Data possession audit with animplicit trusted third-party for cloud storage[J]. Journal of Harbin Engineering University, 2012, 33(8): 1039-1045.

[12] IBM. IBM 4764 PCI-X Cryptographic Coprocessor [EB/OL]. [2011-05-15]. http://www-03. Ibm. com/security/cryptocards/pcixcc/overperformance. Shtml.

[13] YANG L Y, SUN B, XIAO D, et al. Data Possession Auditing in with Near-Zero User-Side Overhead[C]// China Institute of Communications, Liaoning Provincial Communications Administration. Proceedings of the 10th Academic Annual Conference of the China Communications Society. National Defense Industry Press, 2014: 341-347.



YUE Meng, born in 1984, Ph.D, professor, is a member of CCF(No. 39960S). His main research interests include aeronautical telecommunication network and data security.