基于可信计算平台的审计日志安全存储系统

成茂才 徐开勇
(解放军信息工程大学密码工程学院 郑州 450001)

摘 要 针对计算机审计系统中存在的日志的安全性问题,结合TPM(Trusted Platform Module)提供的安全存储、密钥生成和密码运算功能,提出一种面向可信计算平台的审计日志安全存储系统。该系统的意义在于保证日志传输过程中和存储状态下的安全性,并对密钥的存储管理结构进行优化,解决了可信计算平台密钥管理体制中存在的密钥同步问题,从整体上增强了平台密钥管理的安全性。最后进行日志完整性认证算法的安全性分析和密钥使用的复杂度分析,通过实验表明该日志存储系统有较好的安全性和实用性。

关键词 可信计算平台,安全审计,安全存储,密钥管理

中图法分类号 TP309 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.6.030

Audit Log Secure Storage System Based on Trusted Computing Platform

CHENG Mao-cai XU Kai-yong
(School of Cryptography Engineering, PLA Information Engineering University, Zhengzhou 450001, China)

Abstract Aiming at the log security issues existing in computer audit system, this paper proposed an audit log security storage system, combined with the secure storage, key generation and cryptographic operation functions provided by TPM (Trusted Platform Module). The significance of this system is to ensure the security of log transfer and storage, optimize the key storage management mechanism, and solve the key synchronization problem existing in the trusted computing platform key management mechanism, which enhances key management security of platform as a whole. In the end, we analyzed the security of log integrity authentication algorithm and the complexity of key usage. Experimental result shows that this log storage system is safe and practical.

Keywords Trusted computing platform, Secure audit, Secure storage, Key management

1 引言

随着计算机系统变得越来越庞大和复杂,其可靠性和安全性的发挥对于复杂科学、工程和商业应用起着关键作用。它们给用户和应用程序提供各种计算服务,因此经常暴露在大量的危险之中[1],如意外的故障、病毒感染和恶意攻击。其结果是计算机系统发生故障,产生破坏或性能较差,变得不再可靠。因此,安全操作系统一直备受人们的关注。

安全审计[2]是一种事前记录、事后发现威胁的分析技术。CC国际标准和我国信息安全的国家标准都对审计提出了明确要求,它在检测[3]违反安全规则的事件以及事后的分析方面发挥着巨大的作用。安全审计员为系统管理员提供及时的警告信息,实现对系统事件的追踪、审查、统计和报告等功能。

审计系统作为安全操作系统的一个重要组成部分[4],对于获悉用户行为、检测安全隐患、进行事后追查和分析都具有十分重要的意义。审计系统的判断[5]都是基于收集到的审计日志,因此日志的安全性很大程度上决定了审计系统存在的意义。本文提出一种改进的密钥管理机制[6],同时设计了一个实现审计安全存储的算法,其有效地解决了日志的安全性和机密性问题,为审计日志的分析提供了有力保障。

2 可信计算平台

信任在安全领域的基础性角色由来已久[7],最值得注意的是在20世纪70年代末和80年代初发展的可信计算机系统评估标准(TCSEC)。90年代中期,信任管理的概念流行起来。在20世纪90年代末和21世纪初,"可信平台"是安全领域中另一个显著发展的概念,这个概念由可信计算平台联盟(TCPA)引入,即目前已知的TCG(Trusted Computing Group)。可信计算平台[8]是一个包含基于硬件的在信任和安全之间维持平衡的子系统的机器。

TPM是可信计算平台的核心,可信软件栈(TSS Trusted Software Stack)与TPM相结合为可信平台提供支持,用来保证可信计算平台能够为其它应用提供基于硬件保护的安全存储和各种密码运算功能。可信计算平台主要有两个特点:1)基于可信计算平台上的任何操作必须是经过授权和认证的,任何非法用户和恶意用户[9]都无法利用该平台进行非法操作;2)可信计算平台会对系统进行一致性检查[10],平台内部各元素之间也需经过严密的相互认证。

到稿日期:2015-05-19 返修日期:2015-10-26 本文受国家自然基金项目:密码片上系统安全模型结构与验证方法研究(61072047)资助。
成茂才(1991-),男,硕士生,主要研究方向为信息安全,E-mail:765823744@qq.com;徐开勇(1963-),男,博士,研究员,主要研究方向为信息安全、可信计算。

TPM 由输入/输出、密码协处理器、散列消息认证码 HMAC 引擎等组件构成。TPM 芯片首先验证自身固件的完整性，如果验证正确，则对系统进行正常的初始化[11]，再由 TPM 固件依次对 BIOS 和操作系统的完整性进行验证，验证正确则正常运行操作系统，如果验证不正确，平台[12]将自动停止运行。之后，再利用 TPM 芯片内的加密模块生成系统中的各种密钥，对应用模块进行加密和解密，向体系的上一层提供安全通信接口，以保证体系中上一层应用模块的安全。

3 数据安全存储

存储技术和办公自动化的普及，使得越来越多的文件以电子文档的形式保存。但存储技术的发展在带来便利的同时，数据安全问题也逐渐凸显出来，特别是对于军事、政府、金融等涉及大量机密数据的领域。存储介质一旦失控，造成的损失将不可估量。而攻击者往往通过窃取授权用户身份信息越权访问系统，非法篡改、删除、伪造存储系统的重要信息，给整个存储系统数据带来了重大安全隐患。

安全存储系统大致可以分为 4 类：安全网络文件系统、加密文件系统、可生存存储系统和基于存储的入侵检测。它们分别从 4 个不同的层次提供存储系统的机密性、完整性和可用性。实现技术方面包括：从硬件方面实现的自安全存储系统，在存储磁盘内部有一个嵌入式的子系统，通过内置的操作指令对存储的数据进行管理；采取数据分割算法实现分布式存储；由多台互连主机共享实现的存储局域网 SAN、组密钥管理以及容侵能力。

易飞提出在 Linux 文件系统下基于滚动加密的安全存储技术[13]；钱权针对分布式存储的多用户密钥的分发和更新问题[14]，提出基于文件共享组的周期性密钥更新方案；Bellare 对时间链概念进行进一步深化[15]，提出了“前向完整性”的概念；文献[16]介绍了一种基于备份的 NIGELOG 日志保护机制；Schneier Kelsey 先后提出了一系列的安全机制来保证日志的安全性，主要思想是将本地不可信系统中的日志文件传输到其它可信系统中来保证日志的安全；在文献[17]中通过设定不同的访问策略来进行入侵检测，即只有授权者可以访问日志，从而保证了日志记录的机密性。

因此必须健全审计日志的保护机制，将安全芯片应用到审计日志系统中，利用可信计算技术，在安全芯片提供的密码功能、安全存储功能、身份认证功能的基础上，建立数据安全保护机制、访问控制机制。由于密钥生成、身份认证、加密解密等核心安全操作都在安全芯片保护下进行，从物理上保证了密钥和数据的安全，这样就从根本上增强了审计日志的机密性和完整性。

4 审计日志安全存储系统

4.1 总体架构

审计日志的安全存储系统包括 4 部分：日志获取模块、日志添加模块、安全日志文件、密码模块，如图 1 所示。

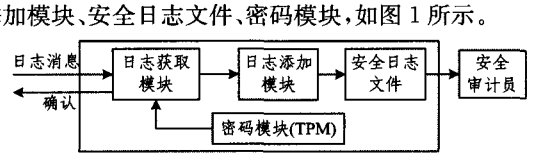

图 1 审计日志安全存储系统

日志获取模块：接收日志消息，消息必须来源于受系统信任的程序，对日志消息进行完整性校验，检查是否可以将其添加到安全日志文件中以及证书是否合法。

日志添加模块：在通过完整性校验之后，将消息发送给日志添加模块，将其转换成为受保护的日志项后添加到安全日志文件中，用到的协议在下文有说明。

安全日志文件：记录系统中发生的事件，并且保证其安全性，使其不轻易遭到篡改。

密码模块：TPM 负责提供安全密钥，保证密钥使用的安全性，存储软件可信表等机密信息，提供必要的加密运算支持。

审计日志安全存储初始化，在密钥树中找到对应结点[18]，输入认证的授权数据 AuthData，从 TPM 中激活安全存储系统专用的密钥对 PubAudit 和 PriAudit，生成 $G_0 = Hash(PriAudit)$ 和 Software Authorization and Key Lookup table(SAKL)。SAKL 表中包含可信软件的简要描述、序列号和各个软件专属的公私钥对[19]，只有系统可信的软件才可以发送消息到审计日志存储系统；该表需要存储在 TPM 中，当系统安装新的软件时，需要对它进行更新。

日志获取模块接收来自软件的日志消息：

可信软件 → 日志获取模块：$\{S_j, \{P\}_{PriS}\}_{PubAudit}$；$S_j$ 表示可信软件在日志审计系统中拥有的唯一的序列号；P_i 表示第 i 个日志消息的内容；$PubS$ 表示 S 的公钥，$PriS$ 表示 S 的私钥；K_i 表示对称密钥；$\{X\}_{PriS}$ 表示用 $PriS$ 签名 X；$\{X\}_{PubS}$ 表示用 $PubS$ 加密 X。

日志获取模块接收的消息具有固定的格式，日志内容用可信软件的私钥签名，与软件的序列号一起用存储系统的公钥加密。日志获取模块会对收到的日志消息[20]进行完整性校验，保证消息的发送是来自可信的软件，并且没有遭到篡改。完整性校验首先用 $PriAudit$ 解密，对照 TPM 存储的 SAKL 表确认 S_j 是否为合法可信的软件，之后验证签名是否通过，如果可以则表示消息内容通过完整性校验[21]。

4.2 日志的添加及认证算法

通过日志获取模块的完整性校验后，日志文件会被传递到日志添加模块，该模块对每个日志项都进行加密，并且每一项用的都是唯一的认证密钥 G。消息内容 P_i 被 N_i 加密，而引入密钥 N_i 可以增加消息的安全性，即使攻击者获得了某个 G_i，它也无法取得里面的内容，因为 S_j 已经被系统用公钥加密。为了对日志消息进行加密，保证密钥的安全性，这里采用哈希链的结构。相比于普通的密钥一直保持不变，密钥 G 是一个不断变化的密钥，所有密钥（G_i 和 G_{i+1}…）全都不一样。哈希链是一个连续的结构，系统可以通过获取初始项重新计算日志文件每项的内容来验证后面的日志文件是完整的或者链式结构的完整性已被破坏。

E_i 表示第 i 个日志项。日志项有自己固定的格式：

$E_i = (\{P_i\}_{N_i}, \{HC_i\}_{PriAudit})$

假设此时日志添加模块收到了日志获取模块发送的日志内容 P_i、认证密钥 G_i 和软件序列号 S_j，执行下面的操作生成安全审计日志文件：

(1) $G_{i+1} = Hash(G_i)$ 生成第 $i+1$ 个日志项的认证密钥；

(2) $N_i = Hash(G_i, S_j)$ 生成第 i 个日志项的加密密钥；

(3)$\{P_i\}_{N_i}$用N_i来对消息内容进行加密；

(4)$HC_i=Hash(\{P_i\}_{N_i},\{HC_{i-1}\}_{PriAudit})$为哈希链上的第$i$个链；

(5)$\{HC_i\}_{PriAudit}$为对第i个项的签名。

其中，$Hash(X)$表示对X进行哈希运算。

认证算法的目的是判断该审计日志文件是否仍具有完整性。通过检查哈希链是否完整，利用前后关联的哈希链结构对日志项的安全性进行检测，从而排除部分形式的恶意篡改行为。对日志文件进行认证，需要初始认证密钥G_0和当前的认证密钥$G_{current}$。下面的算法中需要对整个日志文件进行检查和计算，最终返回变量$Inty$和Re，$Inty$是一个布尔变量，即日志文件是否完整，Re返回不同的值表示不同的含义；$eof()$函数用来判断日志文件是否结束。具体流程如图2所示。

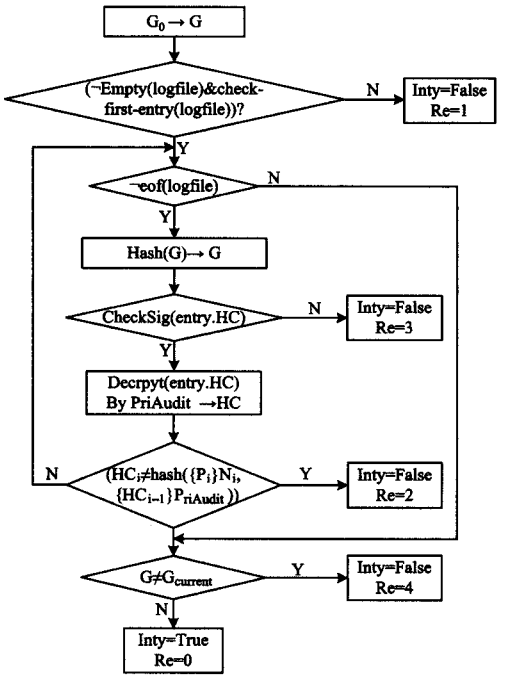

图2　日志完整性认证算法流程

根据算法返回的结果Re，得出以下不同的结论。

0：没有检测到篡改的行为；

1：初始目录已被篡改；

2：存储值和计算值不同，哈希链断开；

3：$PriAudit$私钥签名无效；

4：哈希链长度不匹配。

日志完整性认证步骤如下：

Step1：E_0作为哈希链的初始项，先单独进行检测。如果日志文件为空或者首项检测没有通过，则说明完整性遭到破坏，文件被篡改，返回$Inty$=False，Re=1；若文件非空，且初始检测通过，则进入下面的循环。

Step2：判断日志文件是否结束，如果已结束，跳到Step5；否则进行下一步。

Step3：更新认证密钥G，验证日志项的签名是否有效，失败返回$Inty$=False，Re=3；通过则进行下一步。

Step4：用审计系统私钥$PriAudit$解密，取得哈希链数据，将存储的HC_i和计算得到的哈希值进行比较，若匹配返回Step2继续进行循环；否则返回$Inty$=False，Re=2。

Step5：判断最后计算得到的认证密钥G和当前的认证密钥$G_{current}$是否相同，若相同说明日志文件完整性良好，返回$Inty$=True，Re=0；否则说明日志项的数目不对，表明日志文件被删改，返回$Inty$=False，Re=4。

由于哈希链是一种连续的存储结构，而审计日志文件也在不停地添加，并且存储量较大，因此该审计日志安全存储系统会规定一个固定大小的存储空间，如10M。当存储的日志文件总量超过存储空间之后，系统会把已有审计记录转移到别的区域进行安全存储；对于清空的10M存储空间，后续接收到的日志消息重新开始计算哈希链，生成安全日志文件并存储在该固定区域中。

4.3　认证算法安全性分析

日志消息传输过程中的安全性：由于在TPM中保存着一张SAKL表，这张表的安全性是可以得到保证的，只有系统管理员才有权限对表进行更新操作。在这个表中，只有被系统认证的软件才可以向审计系统提交日志文件，攻击者即使获得了某个传输中的消息，由于不知道日志系统的私钥$PriAudit$和软件唯一拥有的私钥$PriSj$，验签失败，也是无法获得消息内容的。如果攻击者利用截获的已发送消息对系统进行重放攻击，日志获取模块发现消息已存在，会进行拦截，不会影响其他审计日志文件的安全性。

日志文件存储状态下的安全性：日志完整性认证算法可以用来保证各日志项的完整性，检测出可能存在的攻击行为，日志文件以一个序列的形式(E_0，E_1，…，E_n)进行存储。假设攻击者能力比较强，它可能进行的攻击包括对安全审计日志文件的修改、删除、添加操作。下面对3种攻击情况逐个进行分析。

(1)修改：攻击者使用截获的某个已有的日志项E_i去替换日志文件中的日志项E_j，在认证算法中Step4会发现计算得到的哈希值和存储值不匹配，哈希链断开；若攻击者用自己生成一个日志项E_j'来替换E_j，Step3中$PriAudit$验签不过。

(2)删除：攻击者删除首项E_0，则在认证算法Step1中，E_1无法通过初始项的检测；攻击者删除日志文件的中间某项E_i，同样在认证算法中Step4发现计算得到的哈希值和存储值不匹配，哈希链断开；攻击者删除最后一项E_n，该攻击对前面的日志项没有影响，但是在认证算法最后的Step5中会发现哈希链长度和预期不同，认证密钥G不匹配。

(3)添加：攻击者把截获到的某个已有日志项E_i添加到首项位置上，如果i=0，则首项的初始检查没有问题，第(2)步由于E_1和E_0值相同，在Step4中会发现计算得到的哈希值和存储值不匹配，哈希链检查出错；如果i>0，则Step1中首项的初始检查无法通过；攻击者在E_i和E_{i+1}中间添加E_j，若E_j为截获的已有日志项，则哈希链断开，无法通过Step4检查；若为攻击者自己生成的，Step3中的验证签名无法通过；如果是在末项E_n后添加，判断方法和上一种情况相同。

5　改进的基于TPM的密钥存储管理机制

5.1　可信计算平台中的密钥管理架构

TCG共定义了7种密钥类型，可以将其粗略地分为签名密钥和存储密钥SK(Storage Key)；更进一步的分类有：平台密钥、身份认证密钥AIK(Attestation Identity Key)、绑定密钥、普通密钥和继承密钥[22]，对称密钥被单独分类为验证密

钥。TCG 为每种类型的密钥都附加了一些约束条件，以限制其应用。

黄宁玉提出从可信计算技术出发[23]，应用平台完整性检验和报告、加密和密钥树保护 3 方面机制来提高环境存储的安全性，将现实中的等级管理制度用密钥树的方式加以映射。若子节点为对称密钥，则加密唯一的密钥；若子节点为非对称密钥，则加密其私钥。在可信密码模块内部利用父密钥加解密的运算过程和结果不会外泄。

在已有的密钥存储管理机制中，密钥的激活和使用需要经历以下几个步骤：

(1)用户向系统申请某个节点的密钥；

(2)TPM 从存储根密钥开始，利用父亲节点的私钥逐层解开子节点的隐私信息(*AuthData*，*PriKey*)，最终获得用户指定节点的 *AuthData*，*PriKey*；

(3)用户按要求输入授权数据 *AuthData*；

(4)TPM 判断用户输入的授权数据和自己计算得到的数据是否一致，如果相同，则用户获得使用该密钥的授权。

图 3 所示为改进的密钥存储结构，其中灰色部分为本文添加的部分。由图可知，可信计算平台是通过树型结构对上述密钥数据进行保护和管理的，事实上，随着 TSS 不断被使用，会生成越来越多的密钥[24]。TPM 是一个资源有限的模块，内部仅能存储少量密钥和数据。为了有效利用资源，TCS (Trusted Core Service)在 PC 机的硬盘上划分出了一块空间用于在 TPM 外部存储密钥。只有两组密钥：背书密钥 EK (Endorsement Key)和存储根密钥 SRK(Storage Root Key)永久地存储在 TPM 中，其它密钥不使用时由其父亲节点的公钥加密并最终由 SRK 加密存储于外部存储区中，由 TCS 的密钥管理服务统一调度。在需要时，通过 KCM 密钥缓冲存储管理器将其加载到 TPM 内部的保护区域进行激活才可以使用，处于使用状态的密钥被称为激活密钥。如果此时 TPM 内部没有足够的空间提供给新载入的密钥，KCM 将从 TPM 中释放一部分资源，例如，将一些密钥从 TPM 中删除或转存到安全存储设备中。

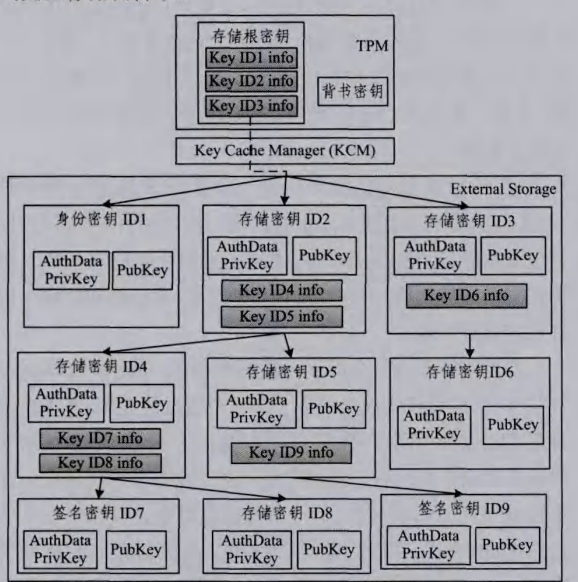

图 3　受保护的存储密钥层级

密钥激活的过程中需要用到授权数据，用户必须通过授权数据的认证才可以获得该结点密钥的使用权，TCG 使用授权数据机制来控制建立 TPM 所有权、密钥使用、对象的迁移等行为。TCG 规定：授权数据＝*Hash*(共享的秘密数据‖随机数)，该授权数据是在密钥产生时一起生成的，并且和密钥一起存放在外部存储空间中。对任何一个结点上密钥的使用，都有一个唯一的获取路径，只有逐级访问才可以解密被父亲结点公钥加密的数据(*AuthData*，*PriKey*)，获得公私钥对。

该系统中默认采用的是 256 位 SHA2 算法。作为一个被全面而广泛验证的单向 HASH 算法，它的主要特征为：不可逆性。从计算角度讲，无法从 MAC 推演出输入信息；无法找到一种以上的输入以产生同一个给定的输出；高雪崩效应，即输入的任何一点变化都会使 MAC 结果产生巨大的变化。MD5 与 SHA-96 已经被攻破，到目前为止，SHA2 被认为是最具安全性的 Hash 函数。

在密钥使用过程中存在 TPM 内外部的不同步问题，某个密钥的所有者认为自己的授权数据不安全，需要更新 *AuthData*，原始的授权数据应当同时失效。但是，如果攻击者获得了原始节点并获悉 *AuthData*，攻击者使用旧密钥节点数据仍然可以通过 TPM 的认证，因为它的父亲节点只负责加解密，除了该密钥节点自身，其他部分都不知道它的 *AuthData* 发生了更改，这就导致了密钥在使用管理上存在不安全问题。

5.2　改进的密钥管理机制

针对上面发现的同步问题，本文对密钥存储管理机制进行了改进，即在父亲密钥节点中加入一个对子密钥授权数据做哈希的数据项 *Keyinfo*(即图 3 中颜色加深的部分)，这样可以更好地解决可信计算平台中密钥存储管理的安全问题。

当用户创建一个新的密钥节点时，将 TPM 计算出密钥授权数据的哈希值 *Keyinfo* 存储到父亲节点，*Keyinfo*＝*Hash*(*ID*，*AuthData*)。TPM 验证某个密钥节点的使用权限时，不仅需要校验 *AuthData* 是否一致，还需要对父亲节点中存储的 *Keyinfo* 和根据密钥节点数据计算得到的 *Keyinfo* 进行校验，这可以保证：攻击者即使使用旧密钥节点信息，也无法获得该密钥的使用权。

对授权数据的更新操作步骤如下：

(1)用户输入授权数据 *AuthData*；

(2)TPM 根据密钥树来访问该结点密钥，对计算得到的 *AuthData* 和用户输入进行比较，如果一致，则用户可以进行下一步操作，否则操作结束；

(3)输入新的授权数据 *newAuthData*，计算 *newKeyinfo*＝*Hash*(*ID*，*newAuthData*)，并将其填入到父亲节点中。

5.3　改进前后密钥管理机制复杂度的分析

在算法改进前，密钥树中有父子关系的节点之间只需要用父亲节点的私钥去解开子节点，获取其私密数据(*AuthData*，*PriKey*)；改进后上下父子节点在解密的同时需要验证

Keyinfo 的真实性，所以会使得算法的复杂度增加一倍，如表1所列。

表1 改进前后用户操作复杂度的比较

用户操作	改进前复杂度	改进后复杂度
添加密钥节点	O(n)	2*O(n)
删除密钥节点	O(n)	2*O(n)
修改密钥节点	O(n)	2*O(n)
使用密钥节点	O(n)	2*O(n)

6 审计系统性能分析

在设计、测试时，同国内外的一些较成熟的安全存储产品进行了比较，包括：TrueCrypt、DriveCrypt、文件保险柜等。

6.1 安全性功能测试

安全审计日志安全存储系统有较强的安全机制来保证审计数据库的安全和审计数据的完整性，由专门的系统审计管理员负责策略订制、数据分析、数据库备份和故障恢复，实现系统访问行为的控制。审计日志信息在被系统初步验证接收后，由日志添加模块实现进一步的认证及统一管理，确保了审计数据的完整性、机密性、准确性。

在测试过程中，随机从审计日志文件中修改、添加或者删除其中的一条记录，验证系统能否及时、准确地发现并提交给审计日志管理员。考虑到实验中不可避免的随机性，测试过程采取逐渐增加样本数量的方法来增强其实验结果的可靠性，结果如表2所列。

表2 系统安全性测试结果

测试数据量	系统成功识别	成功率(%)
100	89	89
500	462	92.4
2000	1916	95.8
10000	9643	96.43

由表2可以发现，随着实验数据的增加，排除个别数据对系统的影响，系统识别的成功率在不断提高，最后成功率趋近于96%左右，说明与同类产品相比，该审计日志存储系统的安全性还是可以接受的。

6.2 系统性能测试

为了测试该基于可信计算平台的审计日志安全存储系统对Linux性能的影响，对该系统进行基准测试(Benchmark Test，BMT)来考察系统的性能。基准测试就是通过一定的测试方法和手段对特定的对象进行测试，从而提供测试对象的性能参数，以作为分析比较的标准。平台配置情况如表3所列。

表3 平台配置情况

Requirements	Minimum	Recommended
CPU architecture	Intel	—
No of CPUs	1	2
Operating System	Linux	NeoKylin6.0
CPU speed	1.5GHz	2GHz
Primary memory	2GB	4GB
Hard disk size	1	2
Backup media	DVD±RW burner	DVD±RW burner dual layer
Network card	—	1(Ethernet 10/100/1000)
Graphics card	1024×768	1280×1024
Sound card	—	Supported by Linux kernel

Lmbench基准测试程序对各种Linux应用程序编程接口进行测试。它包含一套宽带和延迟测量方法的微基准测量程序，设计了高速缓存文件读取、内存读写和延迟、管道、上下文切换、联网、文件系统的创建和删除、进程的创建等测试程序。着重测试内核组件：调度程序、进程管理、通信、联网、内存映射和文件系统，测试结果表4所列。

表4 Lmbench测试结果

测试项目	标准内核	加载审计	性能下降(%)
null I/O	0.53	0.54	1.9
stat	3.16	3.21	1.6
open/close	4.25	4.28	0.7
pipe	4200	4360	3.7
fork	134	151	11.3
exec	596	613	2.8
AF_UNIX	8.94	9.12	2.0
UDP	21.4	22.1	3.2
TCP	29.8	30.6	2.6
TCP connect	89	91	2.2

根据表3中部分实验结果绘制出图4实验数据对比图。从图4的比较及表3的计算数据中可以看出，各测试项的性能下降不多，总体上都小于4%，表明该审计日志安全存储系统对于系统的低层各项操作影响较小，在现有的硬件条件下可以忽略。

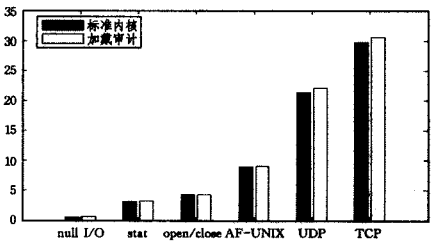

图4 实验数据对比图

结束语 本文在研究了可信计算平台和安全审计系统之后，提出了基于可信计算平台的审计日志安全存储系统，对其中的部分功能进行了设计实现，做了如下贡献：

(1)在可信计算平台的基础上，对审计日志的安全传输及存储协议进行了设计，利用哈希链特有的数据结构，将安全日志文件的前后各项连接起来，并且各项有自己独特的认证密钥，提高了审计日志的安全存储等级，为审计分析提供了坚实的数据基础。

(2)针对现有可信计算平台上密钥管理体制中存在的TPM内部和外部密钥不同步的问题，对TPM中的密钥存储管理结构进行了适当的改进，提高了密钥存储的安全性。由于保持了树形结构，运算量没有显著增加，对硬件和软件来说都比较容易实现。

安全审计作为安全操作系统标准中的一部分，很早就已经被提出，但是长期以来处于一个比较尴尬的地位，总体来说被重视程度都不高，多数情况下，只是作为一个日志收集的功能出现在操作系统中。提出审计日志安全存储系统旨在加强审计日志存储的安全性，下一步将重点对审计日志中蕴含的信息进行挖掘，获得软件的行为模式，寻找到我们需要的信息，为计算机的安全提供帮助，也希望更好地做到事后分析这个关键环节。

参 考 文 献

[1] Shi Wen-chang. Development of secure operating system research[J]. Computer Science, 2002, 29(6): 5-12(in Chinese)
石文昌. 安全操作系统研究的发展[J]. 计算机科学, 2002, 29(6): 5-12

[2] Ding Li-ping, Zhou Bo-wen, Wang Yong-ji. Capture and Storage of Digital Evidence Based on Security Operating System[J]. Journal of Software, 2007, 18(7): 1715-1729(in Chinese)
丁丽萍, 周博文, 王永吉. 基于安全操作系统的电子证据获取与存储[J]. 软件学报, 2007, 18(7): 1715-1729

[3] Rashidi P, Cook D J, Holder L B, et al. Discovering activities to recognize and track in a smart environment[J]. IEEE Transactions on Knowledge and Data Engineering, 2011, 23(4): 527-539

[4] Yuan Chun-yang, He Ye-ping, Pan Xue-jian, et al. Developing High-assurance Secure Information System According to Common Criteria[J]. Computer Science, 2007, 34(2): 17-21(in Chinese)
袁春阳, 贺也平, 潘学俭, 等. 使用 CC 标准开发的高保证安全信息系统[J]. 计算机科学, 2007, 34(2): 17-21

[5] Liu Hai-feng, Qing Si-han. Design and Realization of Auditing in Secure OS [J]. Computer Research and Development, 2001, 38(10): 1262-1268(in Chinese)
刘海峰, 卿斯汉. 安全操作系统审计的设计与实现[J]. 计算机研究与发展, 2001, 38(10): 1262-1268

[6] Vossaert J, Lapon J, De Decker B, et al. User-centric identity management using trusted modules[J]. Mathematical and Computer Modelling, 2013, 57(7): 1592-1605

[7] Shen Chang-xiang, Zhang Huan-guo, Wang Huai-min, et al. Research and development of trusted computing [J]. Chinese Science: Information Science, 2010, 40(2): 139-166(in Chinese)
沈昌祥, 张焕国, 王怀民, 等. 可信计算的研究与发展[J]. 中国科学: 信息科学, 2010, 40(2): 139-166

[8] Peng Cheng, Yang Lu-ming, Man Jun-feng. Research on Partly Tokenized Software Behavior Footprints [J]. Computer Systems, 2013, 34(3): 466-470(in Chinese)
彭成, 杨路明, 满君丰. 带部分标记的软件行为踪迹研究[J]. 小型微型计算机系统, 2013, 34(3): 466-470

[9] Söderström O, Moradian E. Secure Audit Log Management[J]. Procedia Computer Science, 2013, 22: 1249-1258

[10] Accorsi R, Wonnemann C, Stocker T. Towards forensic data flow analysis of business process logs[C]// 2011 Sixth International Conference on IT Security Incident Management and IT Forensics (IMF). IEEE, 2011: 3-20

[11] Shen Guo-hua, Huang Zhi-qiu, Qian Ju, et al. Research on Software Trustworthiness Evaluation Model and Its Implementation [J]. Computer Science and exploration, 2011, 5(6): 553-561(in Chinese)
沈国华, 黄志球, 钱巨, 等. 软件可信评估模型及其工具实现[J]. 计算机科学与探索, 2011, 5(6): 553-561

[12] Song Cheng. Research on Some Key Technologies of Trusted Computing Platform[D]. Beijing: Beijing University of Posts and Telecommunications, 2011(in Chinese)
宋成. 可信计算平台中若干关键技术研究[D]. 北京: 北京邮电大学, 2011

[13] Yi Fei. Safe Storage with Auto-Rolling Encryption Under Linux Ooperating System[D]. Shanghai: Shanghai Jiaotong University (in Chinese)
易飞. 基于滚动加密在 Linux 文件系统下的安全存储技术[D]. 上海: 上海交通大学, 2010

[14] Qian Quan, Wang Tian-hong, et al. Cyclic Key Update Scheme Based on Sharing Group for Distributed Secure Storage [J]. Journal of Shanghai University, 2013, 19(1): 39-43(in Chinese)
钱权, 王天宏, 等. 分布式安全存储中基于共享组的周期性密钥更新[J]. 上海大学学报, 2013, 19(1): 39-43

[15] Bellare M, Yee B S. Forward Intergrity for Secure Audit Logs [D]. USA: University of California, 1997

[16] Tarada T, Koike H. NIGELOG: Protecting Logging Information by Hiding Multiple Backups in Directorie[C]// Proc. of International Workshop on Electronic Commerce and Security. IEEE Press, 1999: 874

[17] Hommes, State S R, Engel T. A distance-based method to detect anomalous attributes in log files[C]// Network Operations and Management Symposium (NOMS). IEEE, 2012: 498-501

[18] Schmidt A U, Leicher A, Brett A, et al. Tree-formed verification data for trusted platforms[J]. Computers & Security, 2013, 32: 19-35

[19] Franklin M. A survey of key evolving cryptosystems[J]. International Journal of Security and Networks, 2006, 1(1): 46-53

[20] Ma D, Tsudik G. A new approach to secure logging[C]// Proceedings of the 22nd Annual IFTP WG 11. 3 Working Conference on Data and Applications Security. 2008: 48-63

[21] Teeter R, Alles M, Vasarhelyi M A. Remote Audit: A research framework[J]. Ssm Electronic Journal, 2010

[22] Huang Ning-yu, Li Shuang, Song Shi-bin. Research on Secure Storage Scheme for Public Data Platform[J]. Journal of Wuhan University (Natural Science Edition), 2012, 58(S1): 61-64(in Chinese)
黄宁玉, 李爽, 宋式斌. 公共数据平台上数据安全存储方案研究[J]. 武汉大学学报(理学版), 2012, 58(S1): 61-64

[23] Wu Xiao-ping, Zhao Bo, Zhang Huan-guo. Secure Key Management of Mobile Agent Based on TPM[J]. Computer Science, 2009, 36(5): 65-67(in Chinese)
武小平, 赵波, 张焕国. 基于 TPM 的移动代理安全密钥管理[J]. 计算机科学, 2009, 36(5): 65-67

[24] Accorsi R, Stocker T. Automated privacy audits based on pruning of log data[C]// Enterprise Distributed Object Computing Conference Workshops, 2008 12th. IEEE, 2008: 175-182