可信密码模块的密钥服务兼容性研究与实现

蒋敏慧[1] 黄宁玉[1] 祝 璐[1,2]
(北京工业大学计算机学院 北京 100124)[1] (武汉大学计算机学院 武汉 430079)[2]

摘 要 可信密码模块TCM芯片提供了非对称算法ECC以及对称算法SMS4来支持密钥机制,它与国际可信计算组织TCG推出的可信平台模块TPM功能相同,但密码算法和密钥管理不同,导致了密钥功能可信应用的兼容性问题。分析两种芯片的密钥特点和可信软件栈的密钥管理方式,提出了可信软件栈TCG服务提供者层TSP与TCG核心服务层TCS的重构方案,以及基于密钥生成流程的兼容方案,以解决密钥服务兼容问题。

关键词 可信平台模块,可信密码模块,可信软件栈,密钥管理,兼容性
中图法分类号 TP309 **文献标识码** A

Research and Implementation of Key Generation Services Compatibility Based on TCM

JIANG Min-hui[1] HUANG Ning-yu[1] ZHU Lu[1,2]
(Department of Computer Science, Beijing University of Technology, Beijing 100124, China)[1]
(School of Computer, Wuhan University, Wuhan 430079, China)[2]

Abstract TPM (Trusted Platform Module) launched by TCG (Trusted Computing Group) is the core module of trusted computing based on cryptographic technology. TPM provides asymmetric algorithm RSA to support key management scheme. Nowadays, China has raised Trusted Cryptography Module (TCM), TCM provides asymmetric algorithm ECC and symmetric algorithm SMS4 to support key management scheme. The two chips can't be compatible, therefore the compatibility problem of key generation trusted application emerged. Through analyzing key management scheme in both TCM and TPM, as well as TCG software stack (TSS), provided the modification scheme on TSP (TCG Service Provider) and TCS(TCG Core Services) of TSS and the scheme based on key creation flow to solve the problem of key services application compatibility.

Keywords TPM, TCM, TSS, Key management, Compatibility

随着信息技术的快速发展,信息安全问题变得日趋复杂与关键。传统的信息安全技术,如防火墙、入侵检测、病毒防范等已经不能很好地解决当前计算机系统所面临的安全威胁,因此提出了可信计算来从一个新的角度解决信息安全问题。

与传统的安全技术不同,可信计算思想主要是通过增强现有的终端体系结构的安全性来保证整个系统的安全。可信计算的核心是称为可信平台模块[1](Trusted Platform Module,TPM)的安全芯片。作为可信计算技术的核心,TPM被业内喻为安全PC产业链的"信任原点",旨在将PC变成一个安全可信的计算平台。TPM安全芯片为各种计算平台提供数据安全存储、身份认证、完整性存储与报告三项服务[1,2]。

TPM的密钥管理机制为以上三大服务提供了核心基础支撑功能。2007年,中国提出了基于不同密码算法的可信密码模块TCM[3],它与TPM提供相同的密钥功能服务,但基于不同的密码体制,更换了非对称密码算法、杂凑算法,引入了对称密码算法,更改了协议,因此与TCG的TPM密钥机制无法兼容。原有的上层密钥应用均是针对TPM密钥服务开发的,这样,当底层芯片被TCM所替代后,上层密钥应用的推广受到了严重阻碍。

如何使原有基于TPM的密钥应用仍然适用于TCM,使得不改动或少改动就能使用TCM的密钥服务,是一个非常重要的研究课题。

1 TPM与TCM密钥机制的相关研究

在TPM规范中,密钥以密钥树的结构进行存储管理。可信计算平台设置一个存储根密钥(SRK)[1],它是非对称密钥,具有唯一性,SRK在TPM中是不可被移除的,其生成是作为平台所有者建立过程的一个重要步骤。

以SRK为可信存储根密钥,SRK之后平台产生的所有密钥都在一棵树下基于树形结构建立管理关系,即:任何一个子密钥都是在其父密钥授权下产生和使用。每个密钥实体都附有权限数据。

TPM采用RSA[1,2]引擎来生成非对称密钥,并使用非对

到稿日期:2009-07-21 返修日期:2009-09-19 本文受国家高技术发展计划(863)项目(2006AA01Z440,2009AA012437),哈尔滨工程大学核安全与仿真技术国防重点学科实验室开放课题HEUFN0801,国家重点基础研究发展计划(2007CB311100)资助。
蒋敏慧(1984-),女,硕士生,主要研究方向为可信计算,E-mail:huihui_quqi@hotmail.com;黄宁玉(1984-),女,硕士生,主要研究方向为可信计算;祝 璐(1981-),女,博士生,主要研究方向为可信计算。

称算法 RSA 来完成所有加密与数字签名操作。图 1 给出 TPM 密钥存储层次结构。

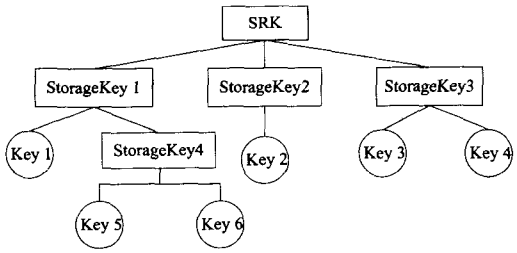

图 1　TPM 密钥存储层次结构

可信密码模块 TCM 是另外一种形式的 TPM，它实现了 TPM 的基本功能，但支持与 TCG 规范中不同的密码机制。与 TPM 相比较，TCM 引入了对称密钥机制，同时更换了非对称密码算法，以及杂凑算法等。

· 对称密钥的引入

TCM 中引入了对称密码算法 SMS4[3]，减少了 TPM 中只使用 RSA 公钥算法和 Hash 算法实施各个环节安全保护而导致的局限性问题。使用对称算法 SMS4 作为加密算法，提高了加密运算速度。在 TPM 中，解密和产生非对称的存储根密钥 SRK 使用的是 RSA 算法；在 TCM 中，解密使用的是 ECC 算法，而产生对称的存储主密钥 SMK（作用相当于 SRK）使用的是 SMS4 算法。

· 签名算法的更替

TCM 采用 ECC[4] 代替 RSA 来作为签名算法，虽然二者都是很常用的非对称算法，但 RSA 与 ECC 相比起来，在达到相同比特安全级别时，ECC 的密钥长度远小于 RSA，即采用 ECC 算法既可以缩小密钥存储空间，又可提高安全性能，使模块整体运算速度加快。在 TPM 中，AIK 称为平台身份密钥，是 RSA 公私钥对，用于进行平台身份标识；而在 TCM 中，AIK 被 ECC 公私钥对 PIK 所替代，作为 TCM 的平台身份密钥，执行相同的密钥功能。

· 密钥存储机制的变更

在 TPM 中，使用 RSA 算法产生非对称的密钥，然后用父密钥加密子密钥的私钥；而在 TCM 中可以根据用户的需求生成两种密钥，使用 ECC 算法产生非对称密钥，然后用父密钥加密子密钥的私钥，或者使用 SMS4 算法产生对称密钥并用父密钥对其加密。

以上种种差异导致了 TCM/TPM 两者所提供密钥服务的不兼容性，即便大部分的接口在语义上是兼容的。然而，底层芯片及其驱动只能由厂商所提供，第三方无法进行任何修改。在 TCG 体系中，上层密钥应用无法直接获得 TPM 密钥服务，需要通过可信软件栈来实现。因此，为了使得原有基于 TPM 的上层密钥应用仍旧可以在 TCM 上运行，本文对可信软件栈进行重构，以实现兼容性。

2　TSS 的密钥服务管理机制

2.1　TSS 体系结构

可信软件栈 TSS[5,6] 是支持 TPM 平台的支撑软件，提供访问 TPM 功能的功能函数，它是 TPM 与上层应用之间的桥梁。TSS 主要提供以下核心功能：提供应用程序对 TPM 功能的单一的访问入口；允许对 TPM 的同步访问；对应用隐藏构建命令流的底层；管理 TPM 的资源。

TSS 由上到下分为 4 个模块，分别为 TSS 服务提供者 TSP，TSS 内核服务 TCS，TCG 设备驱动库 TDDL，以及 TPM 设备驱动器 TDD。

2.2　TSS 服务流程

TSS 服务提供者（TSP）是最顶层的模块，它为应用程序提供许多面向对象的接口，应用程序使用接口 Tspi 来访问 TSP，并通过 TSP 来使用 TPM 提供的功能。TSP 并不能直接和 TPM 通信，而是需要通过 TCS 来进行。TCS 提供一组通用的服务给服务提供者 TSP。TCS 模块利用参数生成器对来自 TSP 模块的参数进行分析和操作后写成一个 TPM 可以识别的字节流，通过 TDDL 传送到 TPM 里面，TPM 响应后把结果以字节流的形式通过 TDDL 返回到 TCS 里去，TCS 把结果传给 TSP，由 TSP 把正式的结果返回到应用程序。

2.3　TSS 密钥与证书管理器（TCSKCM）

密钥和证书都被视为敏感信息，需要进行保密，因此 TCS 密钥与证书管理器提供了对密钥和证书的保护功能。所有在 TSS 中需要被管理的密钥，都应该在 TCS 的永久性存储数据库中进行注册，注册之后的密钥有一个固定的 UUID 值，可以通过它的 UUID 值进行索引，而这些 UUID 值不会提供任何有关密钥注册系统的信息。应用只需要提供 key 的 UUID 值，密钥管理服务就会根据其来调入所需的 parent keys，这些过程对于应用来说可以是隐藏的。一个应用只需要使用 KCM 将密钥调入 TPM，就可以使用这个密钥了。应用必须使用 KCM 调入密钥。KCM 返回一个 application key handle，并且管理一个 application key handle 和 TPM key handle 之间的映射。

2.4　TCG 体系中的密钥创建流程

图 2 是 TCG 定义的密钥创建过程。创建密钥前，需首先在 TSP 层创建 RSA 结构类型的对象结点，结点中存放即将创建密钥的基本属性信息，不包括密钥本身，此结点将添加存储到 TSP 层维护的 RSA 密钥链表中进行资源管理。TSP 向 TCS 层发送密钥生成请求，TCS 将请求数据发往 TPM 并获得 TPM 所提供的密钥生成服务，将所创建的密钥数据最终存入 TSP 层 RSA 对象结点之中，并创建文件存储密钥数据。

Tspi_Context_CreateObject()：创建 RSA 密钥对象

Tspi_Key_CreateKey()：TSP 层密钥生成服务接口

TCSP_CreateWrapKey()：TCS 层密钥生成服务接口

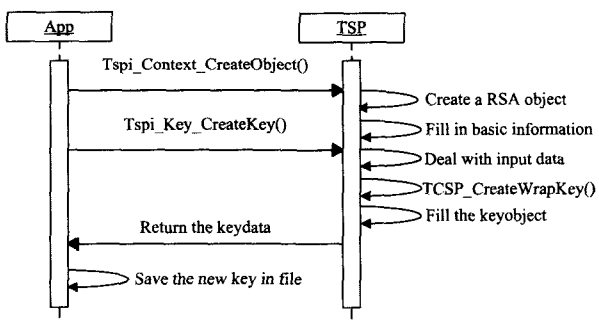

图 2　TCG 定义的密钥创建过程

3　TSS 的密钥服务兼容性方案

根据 TSS 规范，应用会依 TSP→TCS→TDDL→TDD→TPM 的顺序逐级执行。在兼容性研究的场景下，TPM 被 TCM 取代。因此，必须要在 TSP，TCS，TDDL，TDD 中进行

重构，以达到兼容的目的。其中，TDD 和 TDDL 由 TPM/TCM 厂商提供，对于一个第三方，重构它们是困难的。因此，通过重构 TSP 层与 TCS 层是实现密钥应用兼容的有效方向[7,8]。

根据 TCG 体系定义的密钥创建流程以及 TCM 与 TPM 中密钥机制的差异性，为了实现兼容性，除了替换 TSS 中的基础密码算法与数据结构外，还需在 TSP 层、TCS 层中重新定义个别流程步骤。提出可信软件栈密钥服务兼容性实现的重构方案，如图 3 所示(以 SMS4 为例，ECC 同理)。

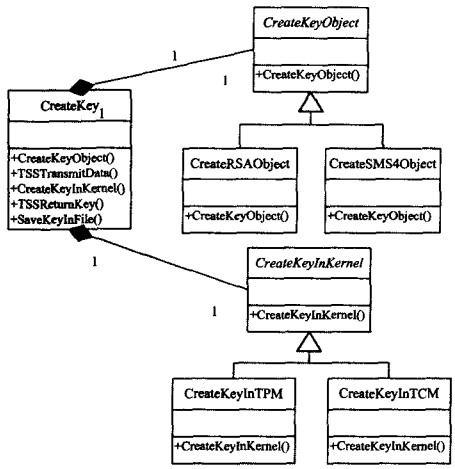

图 3　TSS 密钥服务的兼容性实现方案

密钥生成流程在两种体系结构中固定：创建密钥类型对象→向软件栈发送请求→内核执行密钥生成功能→新密钥信息返回应用层→TSP 层创建文件存储密钥数据。将 CreateKey 定义为一个类，其中的方法为密钥生成的各个流程操作。对于 TPM/TCM，有些方法执行不同的实现内容：密钥类型结点的创建与内核响应。将上述两个方法定义为抽象方法，只提供接口定义，具体的方法实现由继承该方法的子类 TPM 操作与 TCM 操作自行定义，方法名不做更改，保持向上的兼容性。

关于命令的实施分配策略，TCS 中定义了一个相关功能模块，实现方式为设立一个指针数组形式的表格 table。数据传送到 TCS 层时，先将其中的命令号(ordinal)从数据包中解析出来，然后以 ordinal 作为指针数组 Tcs_Func_Table 的下标，搜寻到需要执行的函数功能接口。确定要执行的命令后，TCS 层把从 TSP 层接收到的数据作为输入参数传送到该命令的接口中，以实现最终使用正确的底层芯片安全服务。策略设计[9,10]如表 1 所列。

表 1　TCS 层命令响应分配表

Ordinal	Command Execution
1	OpenContext
2	CloseContext
3	Error
4	TCSGetCapability
5	RegisterKey
6	UnregisterKey
7	EnumRegisteredKeys
8	Error
9	GetRegisteredKeyBlob
10	GetRegisteredKeyByPublicInfo
.....	.....

每项服务在 TCS 中对应一个序列号(Ordinal)，只要保持 CreateKey 的接口定义不变，对应序号不变，内在实现重构为基于 TCM 的设计，就可以在保证上层密钥应用不多做变动的情况下解决兼容性问题。

基于 TCM 重构的软件栈基本保持服务接口定义不变，均为 Tspi_XXX，内容针对 TCM 密钥生成机制实现。其中，为了符合 TCM 需求，TSS 中添加 SMS4 相关数据结构与操作：如 SMS4 对象的数据结构定义等。

4　方案分析

在 TCG 体系中，TPM 只支持非对称密码算法 RSA，其所有密钥、签名算法、加密算法均为 RSA；而 TCM 同时支持非对称算法 ECC 与对称算法 SMS4。TCM 的一次服务响应只能根据用户需求创建一种密钥。由于可信软件栈 TSS 为 TCG 针对 TPM 所设计，只提供了一个密钥创建服务接口，对 TCM 同时实现两种密钥创建服务的兼容性有所阻碍。针对这种情况，提出以下两种解决方案：

(1)在 TSS 重构过程中，添加密钥创建接口，使得原有接口实现 SMS4(ECC)，新增接口实现 ECC(SMS4)。

(2)在 TSS 重构过程中，不添加密钥创建接口，在应用程序中增设选择性变量 Flag，执行如下代码：

```
Swich(Flag)
    { case ECC:Create ECC Key;break;
    case SMS4:Create SMS4 Key;break;}
```

采取方案(1)：上层密钥应用需要更改调用的接口，与兼容性实现的不做修改或少作修改原则相违背，但却能够将两种密钥的创建独立实现，便于最终的单元测试与集成测试实施。采取方案(2)：由于密钥创建的流程相对复杂，选择性变量 Flag 需要传入多个子函数，需要在若干函数接口中增设变量，更改了许多接口定义，过于复杂。在同一接口定义内实现两种密钥创建功能也不利于单元测试实施。但此方案能够保证 TSS 的上层服务接口定义基本不变，上层密钥应用无需做明显改动。

5　兼容性方案测试

实验环节仿真系统的模拟平台为：linux 操作系统，Fedora 7 版本。平台使用可信平台计算模块的模拟器，对于 TPM，采用 tpm_emulator-0.5.1 版本；对于 TCM，使用由北京工业大学可信计算实验室开发实现的 tcm_emulator-0.3 版本；模拟器模仿实现硬件芯片的功能，但成本低于实际硬件，两种模拟器均实现了 TCG 所定义的核心安全功能模块，能够满足实验过程中的需求。模拟器的实现，为可信计算课题的研究提供了良好的开发和测试平台。对于上层可信基础支撑软件栈 TSS，使用 trousers-0.2.9 版本。

为了有效地测试兼容性实现情况，设计如下测试步骤：

1)加载可信平台模块 TPM。

2)启用可信软件栈 TSS，与 TPM 连接，等待上层请求。

3)设计上层密钥测试应用，通过接口调用对 TSS 发送密钥创建请求。

4)卸载 TPM，加载可信密码模块 TCM。

5)启用根据兼容性方案重构实现的可信软件栈 TSS_for_TCM，与 TCM 连接，等待上层请求。

6)运行同样的上层密钥测试应用，对 TSS_for_TCM 发送密钥创建请求。

其中，针对密钥生成的 Tspi 接口应用流程如图 4 所示。

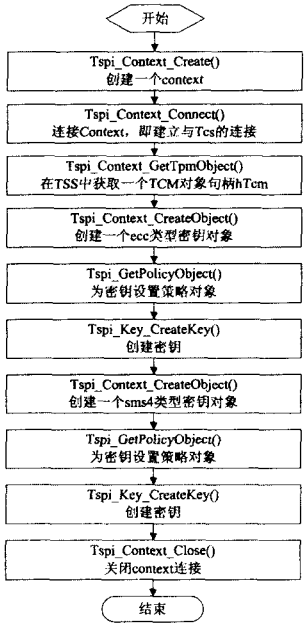

图 4　应用层密钥生成服务接口调用

结束语　本文对 TPM，TCM 的密钥机制差异以及对两种芯片的密钥管理方式进行了对比分析，对可信软件栈的体系结构与密钥管理功能进行分析，对 TCG 体系中的密钥创建流程进行描述，得出了通过重构可信软件栈对上层实现密钥服务应用兼容的思想。在此基础上，提出了对可信软件栈 TSP 层和 TCS 层的具体改进方案，即提供支持 TCM 的密钥机制，重新实现密钥生成流程中的部分环节，以解决基于不同密码算法开发的可信密码模块 TCM 所面临的密钥服务应用兼容问题。

参 考 文 献

[1] Trusted Computing Group. TPM specification version 1. 2. Part 1 Design Principles Revision 103[EB/OL]. 2007:19-21. http://www. trustedcomputinggroup. org/resources/tpm_specification_version_12_revision_103_part_1__3

[2] Strasser M. Software-based TPM Emulator for Linux[D]. 2004: 29-35

[3] 郭菲菲. 可信密码模块的密码配用研究[D]. 北京：北京交通大学，2008：20-30

[4] Zhang Xing, Zhou Ming, Zhuang Jun-xi. Implementation of Ecc-based Trusted Plaform Module[C]// Proceedings of the Sixth International Conference on Machine Learning and Cybernetics. 2007:2168-2173

[5] Trusted Computing Group. TCG Software Stack Specification Version 1. 2 Level1 ErrataA[EB/OL]. https://www. trustedcomputinggroup. org/specs/TSS/TSS_1_2_Errata_A-final. pdf, USA, 2007:516-543

[6] Trusted Computing Group. TCG Specification Architecture Overview Specification Revision1. 4 [EB/OL]. https://www. trustedcomputinggroup. org/groups/TCG_1_4_Architecture_Overview. pdf, USA, 2007:5-41

[7] 董玉娟. 支持异构可信平台的可信软件栈研究[D]. 北京：北京工业大学，2008：29-46

[8] 刘毅. 一种可信软件栈的兼容性改进方案[J]. 武汉大学学报：理学版，2009，55(1)：57-61

[9] Yoder K. Linux TCG Software Stack Low Level Design Version 0. 8r2[EB/OL]. http://trousers. sourceforge. net, USA, 2007: 20-30

[10] IBM. TrouSerS 0. 2. 9[EB/OL]. http://trousers. sourceforge. net, USA, 2007

(上接第 81 页)

纽节点缓存副本复制机制的优势愈发明显。

在式(6)中，设 $A \geqslant B$，因而可知，当 R 的值小于 $A+1$ 时，中心枢纽缓存副本复制机制具有优势。但当 $S_T \cdot C \cdot (A-B) + S_L \cdot C \cdot (A+1-R) \leqslant 0$，即 $R \geqslant [S_T \cdot (A-B) + S_L \cdot (A+1)] / S_L$ 时，源节点缓存机制则开始显示其优势。当查询请求数量 R 达到一定数值并还在不断增大时，源节点缓存有直接明显的优势，因为更多请求可以在本节点内处理。但这种方法将会带来副本安置问题。考虑到节点的存储能力有限，使用源节点缓存在一定时间内将造成网络中副本过多，因此本研究只将源节点缓存作为 JRM 机制的补充，即设置阈值 $A>B$，以保证优先执行 JRM 机制。

综上所述，枢纽节点缓存与源节点缓存相结合对于非结构化离散型对等网络有较好的副本复制性能。同时，我们需要将查询路由信息保存一定的时限，在降低数据库容量的同时为更新枢纽节点提供数据。下一步工作中，需要找出平衡数据库容量和准确提取枢纽节点的方法。在理论上，枢纽节点副本复制机制可以随着非结构化离散型对等网络的扩展而扩展。

结束语　本文提出基于枢纽节点的副本复制机制为对等网络的研究起到了补充作用，综合了传统副本复制机制的优点，并通过降低超级节点的运算量进一步平衡了网络中的负载，降低了网络流量并使得节点工作效率更高。

为了使得系统更加完善，仍然需要进行一些后续工作。首先，需要考虑 JRM 的安全性问题。在当前的 P2P 网络中，节点位置容易暴露的问题尚未得到很好的解决，而且匿名访问也对网络安全构成极大挑战。其次是副本一致性的维护。本文只利用超级节点和源节点保证了副本不会唯一存在，但副本更新时的一致性维护也是非结构化网络所必须面对的问题。最后，随着计算机的发展，超级节点的性能与普通节点的性能之间的差距越来越小，因此需要设计出更多的方法来进一步平衡网络中的负载，以达到提高网络效率的最终目标。

参 考 文 献

[1] The Exabyte Era White Paper[R]. Cisco Systems, January 2008

[2] Ripeanu M. Peer-to-peer architecture case study: Gnutella network[C]// Proceedings of P2P'01. 2001

[3] Tewari S, Kleinrock L. Search and Replication in Unstructured Peer-to-Peer Networks [R]. UCLA-CSD-TR050006. UCLA Computer Science Dept, March 2005

[4] Tewari S , Kleinrock L. Analysis of Search and Replication in Unstructured Peer-to-Peer Networks[C]// SIGMETRICS'05. June 2005

[5] Lo V, Zhou D. Scalable supernode selection in peer-to-peer overlay networks[C]// Second International Workshop. July 2005

[6] Mohammad, Salimullah, Raunak. A Survey of Cooperative Caching, [EB/OL]. http://lass. umass. edu/-raunak/survey. ps

[7] Clarke I, Sberg O, Wiley B. Freenet: A distributed anonymous information storage and retrieval system[C]// Workshop on Design Issues in Anonymity and Unobservability. 2000:311-320

[8] Kubiatowicz J, Bindel D, Chen Y. Oceanstore : Architecture for global-scale persistent storage[C]// Proceedings of th ASPLOS' 2000. Cambridge, MA, 2000:190-201