# TPM虚拟化的研究及进展

谭良1,2，齐能1

1（四川师范大学计算机学院，成都，610068）

2（中国科学院计算技术研究所，北京，100190）

**摘要：**云计算与可信计算相结合是构建可信云环境的重要方法，其最为关键的问题是对TPM（Trusted Platform Module，TPM）的虚拟化。但就当前的研究成果来看，TPM虚拟化不仅存在部分不符合TCG规范的困难，而且还存在诸多安全问题，正成为云计算和可信计算的融合构建可信云环境的瓶颈。介绍TPM虚拟化的基本概念、类型和基本要求，提出了TPM虚拟化的技术分类模型，详细阐述了TPM虚拟化的系统架构、密钥管理、证书信任扩展以及迁移等关键技术的主要研究工作进展，并对这些工作进行了总结。结合已有的研究成果，探讨了TCG架构下TPM虚拟化的研究方向及其面临的挑战。

**关键字：**可信计算；云计算；可信平台模块；TPM虚拟化；vTPM

**Development of Trusted Platform Module Virtualization**

TAN Liang1,2, QI Neng1

(College of Computer Science, Sichuan Normal University, Chengdu, Sichuan, 610068) 1

(Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190)2

**Abstract：**Combination of Cloud Computing and Trusted Computing is an important method to build a trusted cloud environment, and the most critical problem is the virtualization of TPM (Trusted Platform Module, TPM). But in view of the current research，TPM virtualization still not only does not meet the whole TCG specification, but also has a lot of security issues, and it is becoming the bottleneck of building a trusted cloud environment by combination of Cloud Computing and Trusted Computing. This paper introduces the basic concepts, types and basic requirements of TPM virtualization. The classification model of TPM virtualization is put forward by the I/O device virtualization technology. The main research work of the key technologies of TPM virtualization, such as architecture, key management, certification trust extension, migration and so on, are described in detail. Combined with the existing research results, the research direction and challenges of TPM virtualization under TCG architecture are discussed.

**Key Words**: Trusted Computing; Cloud Computing; Virtual Platform Module (TPM); TPM Virtualization; vTPM

# 1 引言

可信平台模块TPM(Trusted Platform Module, TPM)是可信计算的核心[1-9]，是可信计算机系统的信任根，是可信计算的关键技术。TCG（Trusted Computer Group, TCG）[1]定义的可信计算平台的核心功能：度量、存储和报告等均依赖于TPM；可信计算平台的3个信任根：可信测量根RTM(Root of Trust for Measurement, RTM )、可信存储根RTS(Root of Trust for Storage, RTS )和可信报告根RTR(Root of Trust for Report, RTR)等均与TPM有直接的关系，其中可信测量根RTM由CRTM(Core Root of Trust for Measurement, CRTM)和TPM中的一组PCR存储器组成，可信存储根RTS由TPM和存储根密钥SRK组成，可信报告根RTR由TPM和EK组成。TCG不仅是可信计算思想的发起者，也是可信计算技术的组织者和推动者，为了推动可信计算的广泛应用和进一步延拓，TCG成立了一系列工作组，最初包括Embedded System、Infrastructure、IoT、Mobile、PC Client、Server、Software Stack、Storage、Trusted Network Communications和TPM等等[1]，其中TPM工作组是基础和核心的工作组，负责TPM规范的起草、修订和发布，其规范已从TPM1.0、TPM1.2发展到现在的TPM2.0[2-5]。

近年来，云计算[10-20]这一新兴的计算服务方式，以其宽带互联、资源池共享、弹性配置、按需服务和按量收费等独特优势，在各行业应用中快速兴起。用户通过将计算任务和数据委托给云服务商，大大减轻了用户计算和存储的负担。值得注意的是，云计算提供给用户（Tenant）的运行环境是以虚拟机作为载体的[10,11]，用户的运行环境和数据都存放在云端，从而失去了对物理环境的直接控制，如何为用户提供安全的云计算服务是亟待解决的问题[21,22]。国内外可信计算方面的学者敏捷地意识可以通过可信计算来增强云计算环境的可信性，其中通过TPM虚拟化来构建虚拟机可信环境是解决此问题的有效办法之一，如vTPM（virtual TPM，vTPM）[22]。可信计算组织TCG也迅速跟进相关的工作，首先，在2008年相继新增两个工作组Cloud和Virtualized Platform[1]。其中Cloud主要负责多租户基础设施TMI规范的起草、修订和发布，而Virtualized Platform主要负责可信虚拟平台规范的起草、修订和发布，其核心工作之一就是TPM虚拟化；其次，在2013年公布的TPM 2.0规范明确支持TPM虚拟化[2-5]。

对TPM虚拟化的研究具有重要的意义。在云环境中TPM虚拟化为客户虚拟机环境提供可信保障。TPM虚拟化使得每个客户虚拟机在逻辑上都能拥有单个“独有”的TPM简称vTPM，就像拥有一个真实的物理TPM一样。客户虚拟机环境可以使用vTPM提供的度量、存储和报告等功能。特别是可通过vTPM的完整性校验功能实现客户虚拟机环境的信任链传递，通过vTPM的数据保护功能实现客户虚拟机环境数据的密封存储，以及通过vTPM的远程证明功能实现客户虚拟机环境的身份证明。在学术界，自S Berger在2006年提出vTPM架构以来[22]，对TPM虚拟化问题研究已逐渐得到了国内外众多学者、研究机构的广泛关注。在一些国际会议中，如：TRUST2008[23], ESI2008[24](Proceedings of the 2008 Second International Conference on Emerging Security Information), UCC2015[25] (IEEE/ACM 8th International Conference on Utility and Cloud Computing), CSS 2015[26] (2015 IEEE 7th International Symposium on Cyberspace Safety and Security)、TrustCom2012.[27] ( [2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6294581))，MINES 2012[28]（[2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6294581)）, CIS 2012[29] （2012 Eighth International Conference on [Computing Platform](http://ieeexplore.ieee.org/document/6405916/),Computational Intelligence and Security），Proceedings of the 2013 ACM workshop on Cloud computing security workshop[45]等等，均有不少高质量的文章讨论vTPM问题，包vTPM的系统架构、密钥管理、证书信任扩展和迁移等。在国内最近几年召开的“全国可信计算和信息安全”、“全国计算机网络与信息安全”、“中国信息和通信安全”的学术会议上，TPM的虚拟化问题(或与之相关的问题)也是一个主要讨论的热点。在产业界，各大IT巨头也同样关注TPM虚拟化，具有典型代表意义的是微软，尽管微软先前已有虚拟可信平台NGSCB(Next-Generation Secure Computing Base,NGSCB)，但为了进一步提升云环境的安全性，在Windows Server 2016和Windows 10中的 Hyper-V[30]中支持vTPM。Oricale的虚拟化平台Vbox[31]在2009年就已经支持vTPM。最为值得关注的是虚拟化平台的产业巨头VMware在2015年也宣布在vSphere 6.0[32]中支持vTPM。另外，著名的开源虚拟化平台如XEN[33]、KVM[34]等均支持vTPM。

尽管如此，可信计算和云计算融合的程度远远落后于市场预期，vTPM还没有成为各大云平台的主要安全解决方案，其中最为关键的问题之一就是TPM虚拟化，TPM虚拟化是影响可信虚拟化平台产品的应用、推广和普及的主要瓶颈之一，严重地阻碍了可信计算在云环境内进行延伸和拓展。特别是我国尽管也制定了具有自主知识产权的TCM(Trust Cryptographic module，TCM)相关标准[9]，与TPM标准类似，但与TPM2.0相比，TCM标准还没有考虑TCM虚拟化的相关问题。目前国内外也还尚未有详细而全面介绍TPM虚拟化的综述论文。为了对此研究进展有一个总体把握，并促进国内在该方向上的研究，综述TPM虚拟化问题的研究进展工作十分有意义。

本文将介绍TPM虚拟化的基本概念、类型和基本要求，提出了TPM虚拟化的技术分类模型，详细阐述了TPM虚拟化的系统架构、密钥管理、证书信任扩展以及迁移等关键技术的主要研究工作进展，并对这些工作进行了总结。结合已有的研究成果，探讨了TCG架构下TPM虚拟化的研究方向及其面临的挑战。

2 TPM虚拟化的基本概念

为了满足不同的功能需求，目前已出现了许多针对x86架构的不同虚拟化解决方案。结合计算机系统结构和虚拟机所采用的抽象层次的角度，虚拟机系统可分为指令级虚拟化、硬件级虚拟化、操作系统虚拟化、编程语言级虚拟化以及程序库虚拟化等。对于硬件级虚拟化，主要包含三种技术，包括全虚拟化(full virtualization)、半虚拟化技术(para-virtualization)和硬件辅助虚拟化（Hardware-Assistedel virtualization)，这三种技术均包含CPU、内存和I/O虚拟化。本文讨论的是x86架构下的TPM虚拟化，属于硬件级虚拟化的I/O设备虚拟化范畴。

### 2.1 TPM虚拟化的定义

**定义1**：TPM虚拟化，就是在虚拟计算平台中将I/O设备虚拟化技术应用于TPM，使得该平台中的每一个虚拟机，都能安全、独立和自主地使用可信计算功能来提高自身运行环境的可信性和安全性，满足虚拟机用户的可信需求。

从定义1可以看出，TPM虚拟化的概念有三方面含义，首先，在虚拟化技术方面，TPM虚拟化本质上是I/O设备虚拟化；其次，在功能方面，每个虚拟机都能使用TPM提供的可信计算功能；第三，在安全目标方面，通过TPM虚拟化，每个虚拟机能够提高自身运行环境的可信性和安全性，满足虚拟机用户的可信需求。

### 2.2 TPM虚拟化的基本类型

对于硬件级虚拟化，根据VMM（Virtual Machine Monitor, VMM）采用的虚拟化实现方式不同，I/O设备虚拟化采用的实现方式也会不尽相同，其核心在于I/O设备原生驱动的存放位置以及VMM对I/O设备的处理方式。TPM是一个通过LPC总线挂接在主板上慢速I/O设备，采用什么样的I/O设备虚拟化实现技术当然依赖于VMM虚拟化实现方式。因此，根据VMM的虚拟化方式不同，我们将TPM虚拟化方式分为3类。

1. **全虚拟化下的TPM虚拟化**

在全虚拟化下，VMM可以向虚拟机虚拟出和物理TPM完全相同的硬件环境，所有Guest OS看到的是一组统一的虚拟TPM设备，Guest OS对虚拟TPM设备的每一个I/O操作特权指令都会陷入VMM中，由VMM对此进行解析并映射到物理TPM，然后直接控制TPM完成，因而不需要修改Guest OS。对x86结构，某些特权指令不会陷入VMM，此时会采用二进制代码翻译技术。

虚拟机层

客户操作系统

* TPM原生驱动
* 虚拟化层
  + TPM模拟器
  + 重映射客户和真实TPM I/O地址
  + 复用并驱动TPM
* 物理TPM

TPM驱动

TPM 模拟器

虚拟化层

VMM

I/O栈

TPM驱动

物理层



Fig.2-1 TPM Stand-alone Hpervisor model

图2-1 TPM独立模式

根据TPM原生驱动的位置不同，我们还可以将此类虚拟化方式分为TPM独立模型和TPM宿主机模式。将TPM原生驱动直接放在VMM中的虚拟化方式我们称为TPM独立模型，如图2-1所示。将TPM原生驱动放在宿主机中的虚拟化方式我们称为TPM宿主模型，如图2-2所示。

* TPM原生驱动

客户操作系统

* 虚拟化层
  + TPM模拟器捕获客户虚拟机TPM I/O请求并翻译为本地指令
  + VMM驱动转发TPM I/O请求

TPM驱动

TPM模拟器

虚拟机层

虚拟化层

VMM驱动

* 宿主机层
  + TPM接口接收上层转发的TPM I/O请求
  + 复用并驱动TPM

TPM接口

宿主机层

TPM驱动

* 物理TPM



物理层

Fig.2-2 TPM Hosted-based Model

图2-2 TPM宿主模式

1. **半虚拟化下的TPM虚拟化**

在半虚拟化方式下，需要对Guest OS中所有访问TPM的特权指令进行代码替换，因而要开发不同环境的前后端虚拟设备驱动。当应用程序提出访问请求时，前后端驱动共同配合完成访问。如图 2-3 所示。



虚拟机管理域

客户虚拟机

VMM

Guest OS

TPM前端驱动

TPM后端驱动

TPM emulator

TPM驱动

虚拟机层

虚拟化层

物理层

Fig.2-3 TPM Paravirtualization

图2-3 TPM半虚拟化

1. **硬件辅助TPM虚拟化**

在硬件辅助虚拟化下，VMM需要硬件的协助才能完成对TPM的虚拟化，此时Guest OS的内核也不需要修改。借助于在虚拟机管理域中的TPM虚拟设备模型，通过Intel 公司的 VT-x技术或者AMD 公司的AMD-V 技术，消除了TPM特权指令对主机的影响，将运行在不同特权级的Guest OS和VMM实现不同特权级的切换，从而实现TPM虚拟化，如图2-4所示。

客户虚拟机

虚拟机管理域

Guest OS

虚拟TPM 设备模型

TPM驱动

虚拟机层

虚拟BIOS

TPM驱动

VM Exit/Entry

VMM

虚拟化层

VMX OFF

VMX ON

**VMX**

物理层

**CPU**



Fig. 2-4 Hardware-based TPM virtualization

图2-4 硬件辅助TPM虚拟化

### 2.3 TPM虚拟化的基本要求

TPM虚拟化的目的是为虚拟机提供可信计算服务，协助虚拟机建立可信计算环境，让使用虚拟机的用户认为和使用带有物理TPM的可信计算机系统没有明显区别。因此，在虚拟化TPM时，有以下3条基本要求。

1. **等价性（Equivalence）**

所谓等价性，是指虚拟机中的应用程序或用户在使用TPM功能时，除去时间因素外，其余都必须与单独拥有物理TPM的计算机系统一样，包括可信计算的度量、存储和报告等功能。

1. **安全性(Safety)**

所谓安全性包括两方面含义，其一是物理TPM应该由VMM全权管理，客户虚拟机操作系统、应用程序或用户不能直接访问TPM；其二是TPM虚拟化软件系统及架构的安全性。虚拟平台对TPM虚拟化必须满足这两方面的安全要求。

1. **方便性（Convenience）**

所谓方便性是指在TPM虚拟化完成之后，方便对其进行维护、升级和迁移。

3 TPM虚拟化的技术分类模型

TPM虚拟化本质上是I/O设备虚拟化，我们按照I/O设备虚拟化技术将TPM虚拟化分为三类，其一是软件实例化仿真方式，其二是硬件共享方式，第三是聚合方式。下面我们对这三种类型一一介绍。

### 3.1软件仿真方式

所谓软件实例化仿真方式，就是指VMM为每一个需要为用户提供可信执行环境的虚拟机创建一个软件仿真型虚拟TPM实例，当用户需要可信计算功能时，绝大部分由其对应的软件仿真型虚拟TPM实例提供，物理TPM只提供很少一部分功能，如密钥管理、证书扩展等。基本架构如图3-1所示。

客户虚拟机

客户虚拟机

客户虚拟机

Guest OSn

Guest OS2

Guest OS1

TPM驱动

TPM驱动

TPM驱动

I/O响应

I/O请求

I/O响应

I/O请求

I/O响应

I/O请求

VMM

TPM实例化管理器

TPM

虚拟设备模型

TPM

实例化2

TPM

实例化1

TPM

实例化3

TPM驱动

I/O响应

I/O请求



Fig.3-1 Software Simulation Mode of TPM Virtualization

图3-1 TPM虚拟化软件实例化仿真方式

无论是采用全虚拟化、半虚拟化还是硬件辅助虚拟化，均可采用此方式实现。这种方式实质是为每一虚拟机分配一个和硬件TPM具有相同功能的TPM软件实例—vTPM, vTPM通常运行在VMM、虚拟机管理域或隔离设备域中，可实现TPM大部分功能，只有少部分功能需要硬件TPM的支持。此方式的优点是实现简单、灵活性好、性能较高和方便迁移。缺点是安全性不高，但由于vTPM运行在VMM、虚拟机管理域或隔离设备域中，除了特权安全威胁和共享安全外，能够防止一般的安全威胁[[1]](#footnote-1)，即基本的安全可以得到保障。因此，目前此方式成为各虚拟平台实现TPM虚拟化的主流方式。

### 3.2硬件共享方式

所谓硬件共享方式，就是各虚拟机分时访问物理TPM。通常TPM的服务请求和响应采用异步方式实现，当多个客户虚拟机发出TPM I/O请求形成请求队列时，VMM通过一定的策略进行调度，交由TPM依次处理。当TPM处理完一个请求后，将处理结果放入响应队列，由TPM虚拟设备模型将此响应结果返回给相应虚拟机。基本架构如图3-2所示。

无论是采用全虚拟化、半虚拟化还是硬件辅助虚拟化，均可采用此方式实现。这种方式的实质是由硬件TPM为虚拟机提供可信计算功能，优点是安全性高，能防止一般安全威胁、特权安全威胁和部分共享安全威胁等，但缺点也非常明显，不仅实现复杂，迁移性差，而且由于TPM是一个通过LPC总线挂接的慢速外设，随着虚拟机数量的增多，共享性能急剧下降，根本无法使用。

I/O请求

客户虚拟机

Guest OS

TPM驱动

客户虚拟机

Guest OS

TPM驱动

客户虚拟机

Guest OS

TPM驱动

I/O请求

I/O请求

I/O响应

I/O响应

I/O响应

VMM

**TPM**

**虚拟设备模型**

TPM

I/O响应队列

TPM

I/O请求队列

I/O响应

I/O请求

TPM驱动

I/O响应

I/O请求



图3-2 Hardware Share Model of TPM Virtualization

图3-2 TPM虚拟化硬件共享方式

### 3.3聚合方式

所谓聚合方式，即在一个TPM内聚合出多个TPM虚拟功能VF—vTPM, 然后将vTPM直接分配或按需调度给虚拟机使用。基本架构如图3-3所示。

VMM

I/O请求

I/O响应

TPM

I/O请求

I/O请求

I/O响应

I/O响应

客户虚拟机

Guest OS1

TPM驱动

客户虚拟机

Guest OS2

TPM驱动

客户虚拟机

Guest OSn

TPM驱动

Fig.3-3 Aggregation model of TPM Virtualization

图3-3 TPM虚拟化的聚合方式

聚合方式实质是为每一虚拟机分配一个和硬件TPM具有相同功能的VF—vTPM, 且此vTPM运行在TPM内部。此方式的优点是安全性高、性能好、可伸缩性好。但实现和配置复杂、迁移性较差。实际上，已有的TPM 1.2、TPM2.0芯片根本无法实现此种方式，因为LPC总线根本不支持单根虚拟化技术（Single Root I/O Virtualization, SR-IOV）。

4 TPM虚拟化的关键技术研究及进展

TPM虚拟化涉及系统架构、密钥管理、证书信任扩展和迁移等方面的关键技术，本节将一一详述。由于可信虚拟机的信任链模型和远程证明属于“可信计算平台”范畴，因此，本节不予讨论分析。

### 4.1 TPM虚拟化的系统架构

所谓系统架构是指系统的完整组成架构及其合理的逻辑功能模块。根据第3节中TPM虚拟化的技术分类模型，我们将TPM虚拟化系统架构已有的研究成果归纳为三类，其一是软件仿真型TPM虚拟化系统架构；其二是硬件共享型TPM虚拟化系统架构；其三是聚合型TPM虚拟化系统架构。所谓“软件仿真型TPM虚拟化系统架构”，是指为虚拟机提供访问软件仿真型TPM实例的完整组成架构及合理功能模块。所谓“硬件共享型TPM虚拟化系统架构”，是指为虚拟机提供访问硬件TPM的完整组成架构及合理功能模块。所谓“聚合型TPM虚拟化技术架构”，是指为虚拟机提供访问TPM虚拟功能VF的完整组成架构及合理功能模块。

#### 4.1.1软件仿真型TPM虚拟化系统架构

对于“软件仿真型TPM虚拟化系统架构”，无论是国内还是国外，无论是学术界还是产业界，这方面的研究成果比较多。

在学术界，这方面的研究成果最早是IBM T. J. Watson Research Center的S Berger等人[[2]](#footnote-2)在2006年提出[22]，如图4-1所示，该系统架构包括vTPM、vTPM manager、Client-Side TPM driver、Server-Side TPM driver、TPM等实体，其中，vTPM、vTPM manager和Server-Side TPM driver在特权域，Client-Side TPM driver在虚拟机域，TPM在整个虚拟系统的底层硬件层。vTPM 即软件仿真型TPM，为虚拟机提供大部分可信计算功能，vTPM manager负责所有vTPM的生命周期管理，包括创建、挂起、恢复、删除vTPM实例，转发虚拟机对其绑定的vTPM实例请求并返回对应的响应，每个vTPM可通过vTPM manager与硬件TPM产生关联，关联的内容主要包括PCR映射和证书链扩展。Server-Side TPM driver是特权域端TPM驱动，Client-Side TPM driver是虚拟机端TPM驱动，Server-Side TPM driver与Client-Side TPM driver通过VMM进行通信。本文献是TPM半虚拟化的奠基之作，后来这方面的多数研究均基于这一思路。



Fig.4-1 vTPM Architecture

图4-1 vTPM系统架构

2007年，文献[35]也提出了TPM虚拟化的一个通用系统架构GVTPM，如图4-2所示。该系统架构包括GVTPM、GVTPM manager、GVTPM Factory、GVTPM Protected Storage Service等实体，GVTPM即是提供可信计算功能的软件实体，GVTPM Manager包含Key & Session Manager和TPM Driver， 负责GVTPM的生命周期管理、GVTPM与虚拟机的绑定以及密钥和会话管理等。GVTPM Factory负责为GVTPM manager提供创建和删除GVTPM服务，GVTPM Protected Storage Service负责为GVTPM manager提供非易失性存储功能。从以上分析可以看出，文献[22][35]的基本思路是一致的，不同的是vTPM manager功能的实现方式发生了变化，在文献[22]中vTPM manager负责vTPM生命周期的管理及敏感信息的保护存储，而在文献[35]中由另外两个重要实体GVTPM Factory和GVTPM Protected Storage Service来实现，且文献[35]没有明确GVTPM manager、GVTPM Factory和GVTPM Protected Storage Service位于具体的某个域。文献[22][35]所提出的系统架构的优点是vTPM实例与虚拟机分离，容易实现，部署灵活，对VMM影响小，TPM负担轻，如将vTPM、vTPM manger和GVTPM、GVTPM Manager实例放于管理域，尽管容易受到特权安全威胁、共享安全威胁且特权域负担重，但能够防范一般安全威胁。



Fig.4-2 vTPM General Architecture

图4-2 GVTPM通用架构

为了提高vTPM抗特权安全威胁能力，文献[22]提出了一种vTPM架构变种，如图4-3所示，其中vTPM的功能由一个外部安全协处理器插件——IBM的PCIXCC提供，它能够在一个容易受攻击的环境中为敏感数据（如私钥）提供最大的安全。其中第一个虚拟机是硬件的所有者，它获得一个vTPM实例为自己所有，其他的vTPM实例则留给其他虚拟机使用，一个Proxy进程负责在服务端驱动和外部安全协处理器插件间传递TPM I/O指令信息。值得注意的是，运行在PCIXCC中的vTPM仍然是软件型vTPM，代码相对较少，许多安全功能，如密码运算等由PCIXCC中的硬件提供。该系统架构的优点是vTPM实例及vTPM manager位于安全处理器内，不易受到特权安全威胁，但需要增加额外的硬件，成本较高。



Fig.4-3 Security Coprocessor-based VTPM architecture

图4-3 运行在安全协处理器中的 vTPM系统架构

2007年，文献[36]基于Xen首次把vTPM实例从特权域中分离，如图4-4所示，vTPM manager、vTPMD守护进程仍然运行在Domain0，vTPM实例运行在基于LibOS的Trusted VM域中。DomU中的应用将TPM指令通过vTPMmanager和vTPMD传递给vTPM实例，vTPM实例也能将运行的结果通过vTPMD和vTPMmanager返回给DomU中的应用。特权域和隔离域之间通过域间通信机制IDC进行信息交互。该系统架构最大的优点是提高vTPM抗共享安全威胁的能力，减轻Domain0的负担，且vTPM迁移方便。但vTPM与TPM没有联系，底层虚拟平台的信任无法扩展到虚拟机。



Fig.4-4 Single Isolated Domain vTPM Architecture for Xen

图4-4 基于Xen的vTPM单隔离域架构

2008年，文献[37]对vTPM manager、vTPM实例与特权域进一步分离，如图4-5所示。文中首先将特权域中的一个重要功能——域管理功能分离出来，基于MiniOS构建Domain Builder域，简称为DomB，并将vTPM manager和vTPM实例从特权域中分离到DomB，除此以外，DomB还包含了TPM的原生驱动和虚拟平台配置存储，原生驱动是为了方便vTPM manager与硬件TPM交互，虚拟平台配置存储主要是保存vTPM的状态信息以便恢复，DomU可以与DomB中vTPM manager直接进行信息交互，DomU中的vTPM前端不变。与文献[36]相比较，尽管都只增加了一个隔离域，但文献[37]的工作更进一步，文献[36]的隔离域仅仅只运行一个vTPM实例，vTPM实例生命周期的管理、与VM的绑定以及对虚拟机进行完整性度量等均仍在特权域完成。而文献[37]的DomB，不仅将vTPM实例从特权域中分离出来，而且把vTPM实例生命周期的管理、与VM的绑定以及对虚拟机进行完整性度量等相关功能均从特权域中分离出来了。该系统架构降低了特权域的性能负担和共享安全威胁。



Fig.4-5 Single Isolated Domain vTPM Architecture

图4-5 vTPM单隔离域架构

2009年，欧盟委员会赞助的一个可信计算项目OpenTC提出了一个有关TPM虚拟化架构的研究报告[38]，该报告不仅介绍了TPM虚拟化的背景和动机，定义了vTPM状态数据结构，分析了当前已有vTPM架构、限制、vTPM的生命周期、vTPM与VM的绑定等，而且将vTPM基于属性的证明和迁移功能[39]增加到vTPM架构中，并在Xen平台上提出了双隔离域系统架构，进一步分离了文献[37]中的DomB的功能。如图4-6所示。在该系统架构中包括两个域，其一是DomB，该DomB与文献[37]中的DomB类似，但功能减少了，此DomB中没有vTPM manager和vTPM 实例。DomB负责安全地创建可信虚拟域，包含VP-Bulider、HIM、CMS以及BMSI等实体，其中VP-Bulider负责虚拟域的创建，HIM是可信虚拟域完整性度量管理，CMS是虚拟域的配置存储管理，BMSI是基本管理和安全接口；其二是DomU-vTPM，DomU-vTPM域运行vTPM实例，包括VTPM-DM、TIS和H-BMSI，其中VTPM-DM是vTPM设备模型，TIS是可信计算接口规范，H-BMSI与TPM相关的基本管理和安全接口。当产生一个vTPM实例时，vTPM通过H-BMSI产生EK，其私钥由DomB的BMSI保护，并只能被vTPM通过H-BMSI唯一访问，其公钥的hash密文，作为vTPM通过H-BMSI接口执行引用操作的一个外部nonce，隐含地将Xen、DomB和vTPM的完整性度量等关联在一起；当启动新的虚拟机时，首先需将该虚拟机关联的vTPM 域ID附加到该虚拟机的配置表中，接下来VP-Builder对虚拟机的vCRTM进行度量，度量的结果作为vPCR0值存入HIM数据库中，然后VP-Builder验证vTPM的设备模型，确保虚拟机能够正确访问vTPM设备模型提供的TIS接口。最后VP-Builder跟新HIM数据库，建立vTPM与VM之间的相互依赖关系。该系统架构进一步降低了特权域的性能负担和共享安全威胁。



Fig.4-6 Double Isolated Domain vTPM Architectur

图4-6 运行在双隔离域的vTPM架构

在国内，类似的文献出现在2010年，文献[40]认为，将整个特权域都看成TCB会威胁到vTPM的安全，因为TCB太大容易产生漏洞。为了解决这个问题，文中提出了一种新的vTPM架构，如图4-7所示。通过创建一个新的管理域Dom A，将原来在特权域中的vTPM、vTPM manager以及TPM原生驱动分离到管理域DomA中。创建管理域有两个目的，其一是使得vTPM及其相关组件免受非法访问和调用，其二是修改TPM的访问流程，并通过TPM来保护DomA，提高vTPM及其相关组件的安全性。该系统架构的思路与文献[36]、[37]和 [38]的基本相同。

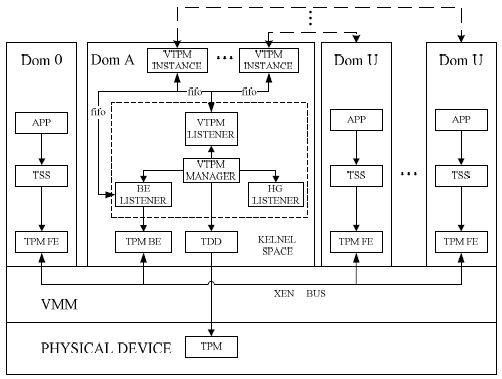


Fig.4-4 Dom A -based vTPM Architecture

图4-7 基于Dom A的vTPM架构

2015年，为了防止云环境中攻击者可能利用虚拟机的回滚机制（一种重要和常用的功能）发起攻击，国内文献[41]基于Xen提出了抵御回滚的可信虚拟平台模块（rollback- resilient TPM，简称rvTPM）系统架构，并在Xen中实现了rvTPM原型系统，如图4-8所示。包含四个模块：（1）信息收集模块，它负责收集有关回滚事件的信息。（2）信息注册模块，这个模块在中，它负责记录和共享由信息收集模块收集的状态信息。（3）回滚模块，它执行快照和回滚操作。（4）回滚日志模块，它记录回滚事件。从图4-5可以看出，信息收集模块、回滚模块和回滚日志模块是在Dom0中，而信息注册模块是在Xen VMM中，vTPM后端驱动与信息收集模块、回滚模块和回滚日志模块交互，而rvTPM manager与信息注册模块交互。与之前的文献相比，该系统架构沿用了早前的TPM半虚拟化的方式，但防止了虚拟机回滚攻击，降低了一般安全威胁，进一步提高了管理域和vTPM的安全性。



Fig.4-8 rvTPM architecture for Xen

图4-8 基于Xen的rvTPM架构

在开源社区，虚拟平台系统Xen和KVM均支持TPM虚拟化。Xen[33]是最早支持TPM虚拟化的开源虚拟平台系统，学术界许多有关TPM虚拟化的研究成果均将Xen作为实验平台。一开始，Xen的TPM虚拟化系统架构如图4-9所示。该系统架构包括vTPM实例、vTPM manager、vTPM后端驱动和前端驱动、TPM等实体，其中，TPM实例、vTPM manager、vTPM后端驱动在Dom0，vTPM前端驱动在虚拟机域，TPM在整个虚拟系统的底层硬件层。



客户虚拟机

Xen

Guest OS

TPM前端驱动

TPM后端驱动

vTPMn

TPM驱动

vTPM2

vTPM1

vTPM manager

Dom0

Fig.4-9 early vTPM architecture for Xen

图4-9开源Xen的早期vTPM架构

2013年7月Xen发布了3.2版本，TPM虚拟化系统架构发生进一步变化。如图4-10所示。Xen将vTPM manager、vTPM与VM的绑定等功能从Dom0中分离出来，减轻Dom0性能负担，降低了共享安全威胁。

Dom0

vTPM

mananer域

vvTMPn域

vvTMP1域

vTPMmgr

Min OS

vTPM1

vTPMn

Min OS

Guest OS

客户虚拟机

TPM front

Min OS

Min OS

IDC

IDC

TPM2 TIS

TPM Back

TPM front

TPM back

TPM Back

TPM front

Xen



Fig.4-10 current vTPM architecture for Xen

图4-10开源Xen当前的vTPM架构

KVM[34]支持TPM虚拟化比Xen晚，KVM的TPM虚拟化主要依赖QEMU。2007年，QEMU通过补丁方式支持对TPM emulator的访问，使得虚拟机可以使用TPM emulator提供的可信计算功能，2013年4月，QEMU增加了新的补丁，使得虚拟机可以通过pass through技术直接使用物理TPM，但这种方式使得在一台物理机上只有一个虚拟机可以使用TPM设备，且物理计算机与虚拟机不能同时使用TPM芯片，限制了可信计算功能在虚拟机上的应用。2013年12月，QEMU再次增加新补丁，实现多个虚拟机同时使用由软件模拟的TPM功能，虚拟TPM的功能与接口与物理TPM芯片保持一致。系统架构如图4-11所示。该系统架构包括btpms、QEMU with TPM、SeaBIOS、Kernel IMA等实体，其中libtpms库用于向QEMU提供TPM功能，包括对称/非对称加密、安全存储、完整性度量和安全签名等。QEMU with TPM实际上是在QEMU 中新增一种TPM设备类型和TPM设备的后端驱动，TPM设备的后端驱动接收虚拟机发送过来的TPM设备操作，解析操作类型并调用libtpms相应的接口完成TPM设备操作，并将操作结果返回给虚拟机。SeaBIOS是 x86 结构下一种BIOS的开源实现，完成初始化硬件工作。该系统架构比较复杂，涉及的组件和实体较多，且libtpms与底层物理TPM联系少，性能与Xen中vTPM相比不高，也存在特权安全威胁和共享安全威胁。



Fig.4-11 vTPM architecture for KVM

图4-11开源KVM的vTPM架构

对于产业界，VMware[32] 在2015年也宣布在vSphere 6.0[32]中支持vTPM；Microsoft[33]在其windows sever 2016中的Hyper-V 中支持vTPM, Orical[31]的VBox通过IBM的PCIXCC支持vTPM。但无论是VMware、Microsoft还是Orical，均没有公开其系统架构。

综上所述，由于软件仿真型TPM虚拟化具有实现简单、灵活性好、性能高和方便迁移的特点，所以无论是学术界、产业界还是开源社区对软件仿真型TPM虚拟化系统架构研究较多，获得了较多的研究成果。总结起来，软件仿真型TPM虚拟化系统架构主要围绕两个个方面展开，首先，如何防止各种安全威胁，特别是特权安全威胁和共享安全威胁，提高系统架构的安全性；其次，如何提高TPM虚拟化的性能，减少对虚拟平台的安全和性能影响。

#### 4.1.2硬件共享型TPM虚拟化系统架构

对于“硬件共享型TPM虚拟化系统架构”，国外学术界研究较多，国内学术界研究相对较少。整个产业界和开源社区还没有类似的设计和实现。

在学术界，最早的研究成果出现在2008年，文献[23]首先提出了物理TPM共享系统架构，属于“硬件共享型TPM虚拟化”。如图4-12所示，该系统架构包括客户虚拟机中的HyperTPMDriver，VMM 中的HyperCallDispatcher、TPM service、Filter Policy、Guest Measurement、Guest Memory Management 和TPM Driver。其中HyperCallDispatcher接受来自各个客户虚拟机的请求并进行分发处理，TPM service 为各客户虚拟机提供TPM功能服务，是整个系统架构的核心，包含Command Blocking、Virtual PCRs、ContextManage、Resource Virtualization和Scheduler。由于物理TPM是一个独占设备，且资源有限，各客户虚拟机在共享物理TPM时不仅需要Resource Virtualization虚拟化有限的关键资源，如：密钥曹、授权会话曹等，而且还需要管理其运行时的上下文Context并进行调度。Filter Policy 是过滤策略，Guest Measurement和Guest Memory Management主要负责客户虚拟机的完整性度量和内存管理。不过该文献并没有具体实现。该系统架构的优点是安全性较高，可以防止一般特权安全威胁和共享安全威胁，但TPM的管理和调度均在VMM，VMM变大，增加了一般安全威胁，且性能不高。



Fig.4-12 Hardware TPM sharing Architecture

图4-12 物理TPM共享架构

同年，文献[24]重新设计一款新的TPM来支持虚拟化，其结构如图4-13所示。与已有的TPM相比，能支持多客户虚拟机共享，即对各虚拟机就像独自拥有TPM。从结构图可以看出，与TCG的TPM相比，增加了一个非易失性存储，其中包含两个关键的数据结构，其一是Active TPM-Control，其二是Root-Data。Active TPM-Control数据结构包含：SRK、PCRs、AIK、EK、单调计数器、其他非易失性存储值、委托授权表、TPM上下文数据、DAA参数*f*以及相关授权数据。Root-Data数据结构里面包含的是具体敏感数据，如封印密钥等。当某虚拟机发出TPM I/O请求时，VMM经过调度将该虚拟机的TPM-Control 数据结构加载到TPM中并解密，TPM按照此TPM-Control 数据结构执行虚拟机的I/O请求并将结果返回给客户虚拟机。一旦有别的客户虚拟机发出TPM I/O请求，VMM将用Root-Data数据结构中的密钥加密保存正在TPM内部TPM-Control 数据结构，并将下一虚拟机的TPM-Control 数据结构载入。文中还提出了CPU硬件保护机制，VM运行在CPU ring 1环，VMM运行在CPU ring 0环，运行在1环的VM只能操作自己的 TPM-Control数据结构，而VMM可以管理所有VM的TPM-Control数据结构。此外，文中还提供了管理TPM-Control数据结构和敏感指令的扩展命令集，并详细介绍了敏感指令处理、TPM-Control上下文切换、TPM调度等算法和流程。该文献与文献[35]最大的不同是通过在TPM中增加新的功能和CPU的硬件保护机制来实现TPM共享，使得TPM从独占设备变为可共享设备。该系统架构的优点一方面是安全性较高，可以防止一般安全威胁和共享安全威胁，但需要防止特权安全威胁；另一方面是性能较高。由于该系统架构是借助于CPU硬件保护机制来管理和调度TPM，VMM增加的软件代码较少。



Fig.4-13 Supporting shared HardwareTPM architecture and CPU protection mechanism

图4-13支持共享的TPM结构及CPU保护机制

文献[42]是2008年5月美国专利局授权的专利，提出了数据处理系统的物理TPM共享方案。如图4-14所示，该系统架构包括逻辑分区系统LPAR中的TPM TSS和TPM DEVICE DRIVER，VMM 中的IPUT QUEUES、OUTPUT QUEUES和HTPM ITERFACE UNIT，以及HTPM中的多个LTPM（逻辑TPM）和一个物理TPM。LPAR中两个实体TPM TSS和TPM DEVICE DRIVER主要作用是为LPAR中的应用访问TPM提供接口和驱动。VMM 中IPUT QUEUES表示所有访问TPM请求队列，OUTPUT QUEUES表示响应队列。HTPM ITERFACE UNIT表示VMM中访问HTPM的接口，LTPM负责为LPAR提供绝大多数可信计算功能，TPM主要负责敏感数据保护。当VMM创建一个LPAR时，立刻对应创建一个LTPM与之绑定。在进行数据处理时，LPAR中的应用将TPM请求发送给VMM，形成请求队列IPUT QUEUES，VMM调度请求队列里的请求到HTPM，由LTPM和TPM共同处理此请求并返回响应OUTPUT QUEUES。该系统架构的最大特点是由TPM生成多个LTPM，并通过VMM将LTPM与LPAR调度绑定。由于LTPM是位于HTPM中，因而安全性较高。



Fig.4-14 sharing HardwareTPM architecture for data process system

图4-14数据处理系统的TPM共享架构

2011年，文献[43]认为，随着技术的发展和应用的更新，如移动计算和云计算，TPM已经不适应新技术需求和用户需求的多样化。为此，该文献提出了一种新的TPM系统架构Dynamic-Context TPM (dcTPM)，如图4-15所示。该系统架构包括FPGA片上系统、各种外围控制接口和多个物理TPM。其中FPGA片上系统环境是uCLinux，有一个dctmp守护进程，不仅负责与主机PC、内存、I/O等链接，控制着多个软件仿真型vTPM，而且还通过LPC总线控制多个硬件型TPM。无论是软件型vTPM还是物理TPM，其上下文Context均由 dctmp管理，dctmp根据用户的具体需求进行调度。该系统架构的特点是集成了软件TPM和硬件TPM，比较灵活，能够较好满足用户的需求，且安全性高。

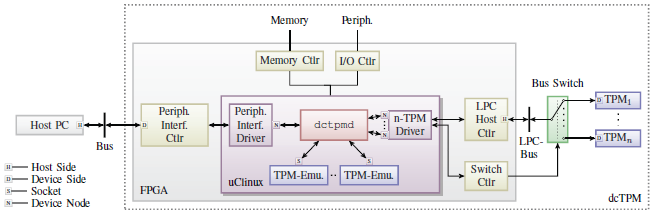


Fig.4-15 dcTPM Architecture

图4-15 dcTPM架构

文献[44]是2012年9月美国专利局授权的专利，提出了一个既支持软件仿真型TPM虚拟化又支持硬件共享型TPM虚拟化的系统架构。如图4-16所示，该系统架构主要包括UOS域、SOS域、vTPM-VM域、TPM Conduit和硬件。其中UOS是用户虚拟机域。SOS域是服务虚拟机域。vTPM-VM域是vTPM管理域，其中包含vTPM0至vTPMn的Context、vTPM Context Manager和TPM ConduitBE，vTPM0 至vTPMn Context主要包含n+1个vTPM及其上下文，vTPM Context Manager负责管理n+1个vTPM及其上下文，TPM Conduit BE是TPM Conduit的后端驱动，负责和TPM Conduit交互信息。而TPM Conduit位于VMM，负责将来自UOS或SOS中的TPM请求进行分发并返回来自vTPM-VM域或Muti-Context iTPM的响应结果。硬件层中，除了传统的CPU、Memory和I/O外，还包括Muti-Context iTPM、ME、VE和ICH，其中Muti-Context iTPM是一个物理TPM，ME是其管理引擎，VE是其虚拟化引擎，ICH是其输入输出控制器。Muti-Context iTPM有两种工作模式，一是独立设备工作模式，此工作模式与传统的TPM一致。二是共享模式，此模式下支持TPM Context管理。当一个UOS或SOS发出TPM访问请求时，TPM Conduit根据此请求的具体要求，如TPM版本、性能要求等，要么转发给vTPM-VM域，由软件型vTPM负责处理并相应，要么转发给Muti-Context iTPM，又Muti-Context iTPM负责处理并响应。与文献[43]一样，该系统架构也集成了软件TPM和硬件TPM，比较灵活，能够较好满足用户的需求，但安全性不如文献[43]，文献[43]对软件和硬件TPM的管理均在FPGA中实现，而该架构是由VMM来实现，显然使得VMM变大，增加了一般安全威胁，用户可以接近完全信任。



Fig.4-16 Architecture for supporting both software simulation vTPM and Hardware sharing vTPM

图4-16既支持软件仿真型又支持硬件共享型的TPM虚拟化架构

2013年，为了给多核系统创建和管理并发安全执行环境，文献[45]提出了HV-TPM系统架构，如图4-17所示，该系统架构包括一个I/O INTERFACE、SCHEDULING、RUN-TIME STATE、STORAGE Per-VM Persistent State和STANDARD TPM COMPONENTS，其中I/O INTERFACE包括两个实体，其一是DRTM Support，支持动态完整性度量；其二是Virtualization Support，支持I/O虚拟化。SCHEDULING负责请求队列的管理和调度，RUN-TIME STATE包含VM状态表和TPM当前上下文，STORAGE Per-VM Persistent State是负责存储每一个虚拟机需要持续保持的状态数据。该系统架构的特点是基于HV-TPM为多核系统创建和管理并发安全执行环境。



Fig.4-17 HV-TPM Architecture

图4-17 HV-TPM架构

2013年，文献[46]提出了TPM 2.0的半虚拟化架构，如图4-18所示。文中认为TPM2.0的核心功能易支持半虚拟化设计，因此没有必要在VMM层采用软件型vTPM对TPM 2.0的功能进行仿真模拟，各虚拟机采用TPM Service方式共享TPM。整个主机计算平台包括虚拟、特权虚拟机、半虚拟化TPM服务和硬件。其中特权虚拟机中TPM manager负责硬件TPM的管理。半虚拟化TPM服务包括Command Filter、Resource Manager、Log Manager、Sheduler和Migration Manager。Command Filter是TPM命令解析和过滤器，Resource Manager负责TPM服务资源管理，Log Manager负责TPM服务日志管理，Sheduler负责TPM调度，Migration Manager负责TPM虚拟化迁移。该系统架构的优点是安全性较高，但TPM的管理和调度均在VMM，VMM变大，影响了VMM性能，增加了VMM一般安全威胁。



Fig.4-18 Virtualization Architecture for Hardware TPM 2.0

4-18硬件共享型TPM 2.0虚拟化架构

在产业界，包括VMWare公司的VMWare、Citrix公式的XenServer、微软的Hyper-V、IBM的PowerVM等，均未见相关成果发布与展示。在开源社区，包括Xen、KVM、QEMU、Bochs等，也没见此实施方案。

综上所述，由于硬件共享型TPM虚拟化优缺点都比较突出，优点是安全性高，缺点是实现复杂、性能差以及不便迁移等，所以学术界研究较多，但产业界和开源社区相关的工程成果还很少。总结起来，硬件共享型TPM虚拟化系统架构主要围绕三个方向展开。首先，如何共享TPM，包括两个方面，一方面，在虚拟化管理层通过增加TPM上下文管理、调度以及改进系统架构来共享TPM；另一方面，通过设计新的TPM来达到共享TPM的目的。第二是如何提高性能，主要是降低复杂性，如TPM上下文的管理模式、TPM调度策略等。第三为了更好满足用户需求和适应新的技术，软件仿真型TPM虚拟化和硬件共享型TPM 虚拟化架构融合趋势明显。

#### 4.1.3聚合型TPM虚拟化系统架构

对于“聚合型TPM虚拟化系统架构”，无论是国内外学术界、产业界还是开原社区，目前研究工作很少，还没见公开发表、发布或展示的成果。

“聚合型TPM虚拟化”的关键技术是基于I /O 硬件虚拟化技术——SR-IOV [47]，SR-IOV 技术是一种基于硬件的虚拟化标准协议，由PCI-SIG 小组提出，是PCI-Express 协议的一个子集。SR-IOV标准允许虚拟机之间高效共享PCI-Express 设备，在PCI-Express 的硬核中嵌入一个模块（该模块是在硬件中实现的），理论上可以获得与本机性能媲美的I /O 性能和利用率。SR-IOV 技术主要应用于网卡设备，提高了虚拟机访问网络的速度和效率。由于TPM不是PCI设备，而是一个LPC设备，因此，已有的PCI设备虚拟化规范不能应用于TPM设备的虚拟化。

目前，TCG虚拟平台工作组TCG VPWG（Virtualized Platform Work Group）的部分工作也涉及到如何在TPM内部构建多个vTPM使其支持虚拟化的问题，但没有公开相关的解决方案[1]。国内文献[48]研究了I/O 硬件虚拟化技术——SR-IOV，并针对虚拟计算环境安全隔离的问题提出了一种基于SR-IOV 技术的虚拟环境安全隔离模型，该模型采用加密卡和pass through技术实现网络数据隔离和数据加密隔离，其中该加密卡实现的功能即是TPM的主要功能之一。

我们认为，尽管TCG TPM规范只规定TPM的功能和架构，没有限定实现技术。但确指明了TPM与计算系统的接口方式：LPC，这一限定实际上妨碍了TPM与计算系统的通信性能提升，无法满足云环境多虚拟机、动态、并发等方面的性能要求。因此，研究满足硬件虚拟化技术SR-IOV的TPM及其系统架构是一个紧迫的任务。

### 4.2 vTPM的密钥管理

vTPM的密钥管理，与TPM的密钥管理一样，也包括设定密钥类型、属性、存储层次结构以及密钥生命周期管理。为了满足用户使用的一致性和程序接口的一致性，vTPM与TPM的密钥管理应基本一致。

最早提出vTPM架构的文献[22]指出每一个vTPM的密钥组织结构类似于TPM的独立密钥层次结构，密钥类型与TPM的密钥类型相同，即除背书密钥外，其他六种密钥包括存储密钥、身份证明密朗、签名密钥、绑定密钥、遗留密朗和对称密钥等组成一颗密钥树，vSRK为该密钥树的根，在该密钥树中，孩子节点密钥的私钥由父亲节点密钥的私钥加密保护。而各种密钥生命周期管理，包括密钥生成、载入、注册和销毁等管理操作均在特权域中，且不依赖于物理TPM，其好处在于便于密钥的快速生成以及便于vTPM迁移，密钥层次结构中的密钥由TPM中的对称存储密钥加密存储在外设上。但该文献并没有讨论vTPM密钥属性。实际上，由于vTPM的实现机制和软件特点，vTPM的密钥属性呈现出新的特征。

文献[35]指出，由于vTPM运行环境的不同，密钥属性应该发生变化，这可能违背TCG规范对TPM密钥属性的限定。比如vTPM的vEK密钥，按照TCG规范规定，它应该是不可迁移的、只能用于申请AIK证书的密钥，但在vTPM中，为了便于vTPM的迁移，vEK是一个可迁移的密钥，不仅用于申请vAIK证书，而且用于加密或签名vTPM迁移时状态数据，因此，vEK密钥只能是遗留密钥。再如vTPM的vAIK密钥，按照TCG规范规定，它应该不可迁移的，只能对TPM内部的PCR值进行签名，而客户虚拟机的vPCR是驻留在外部内存中，vAIK对vPCR的签名是在物理TPM外部发生的，这显然不符合身份密钥的TCG规范，因此，vAIK密钥只能是TCG规范中的签名密钥。再如，vTPM的vSRK存储根密钥 ，按照TCG规范，它应该是不可迁移的，只能在TPM内部保护密钥树，但在vTPM，vSRK是驻留在外部内存中，且为了便于vTPM的迁移，vSRK也是一个可迁移密钥，因此，vSRK也应是一个遗留密钥（或加密密钥）。所以，尽管每一个vTPM有类似于TPM的独立密钥层次结构，其密钥属性如表1所示，但这个密钥树层次结构应该更加扁平。但该文献并没有讨论vTPM密钥的存储结构以及密钥生命周期管理。

Table 1 relationship between the key type of vTPM and the key type of hardware TPM

表1 vTPM的密钥类型与物理TPM的密钥类型的对应

|  |  |
| --- | --- |
| Virtual Key Type | Hardware TPM\_Key\_Type |
| Virtual Endorsement Key | TPM\_LEGACY |
| Virtual Storage Key | TPM\_LEGACY |
| Virtual Attestation Identity Key | TPM\_SIGNING |
| Virtual Signing Key | TPM\_SIGNING |
| Virtual Binding Key | TPM\_BINDING |

文献[49]提出了一种适合于VM-vTPM迁移的密钥存储结构。如图4-19所示。其中，vSRK是vTPM密钥树的根，vAIK是vTPM的身份证明密钥，SRK是TPM密钥树的根，AIK是TPM的身份证明密钥。gSRK和SK属于中间层密钥。vSRK受到中间层密钥gSRK的保护，gSRK受到SRK的保护，是一个不可迁移的非对称密钥，引入gSRK的目的是为了vSRK能够迁移。SK受到AIK的保护，也是一个不可迁移密钥，引入SK的目的是为了将vAIK和AIK关联起来，为迁移的数据签名。另外，文中还将vTPM密钥分为两大类，一类是内部密钥，包括vSRK，部分加密、签名、绑定和遗留密钥等。在虚拟机迁移时，内部密钥不会迁移。另一类是外部密钥，包括部分签名、加密和遗留密钥，这些密钥是在虚拟机迁移时对迁移数据进行签名和加密的密钥。在虚拟机迁移时，为了防止vTPM迁移事务的链接攻击，一个vAIK只能用于VM的一次数据迁移，因此，vAIK也具有不可迁移性。但该文献并没有讨论vTPM密钥的生命周期管理。

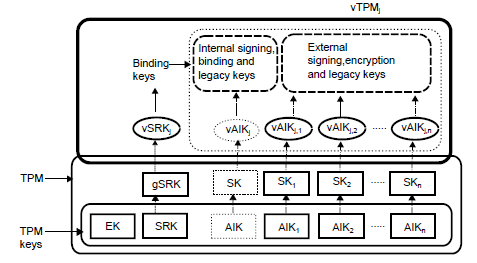


Fig.4-19 Key structure suitable for VM-vTPM migration

图4-19 适合于VM-vTPM迁移的密钥结构

2013年10 月，TCG发布TPM 2.0，在密钥管理方面的主要变化是主密钥生成方式，即：对TPM主密钥包括EK、SRK等通过密钥种子使用密钥派生算法KDF来生成，这不仅节省了TPM存储空间，而且容易为vTPM建立统一的密钥管理机制和迁移机制。TPM 2.0后，vTPM在密钥类型、存储层次结构等方面没有发生变化，主要变化体现在主密钥生成和主密钥属性上。文献[46]率先提出基于密钥种子为虚拟机的vTPM分发密钥并生成背书密钥和密钥树，如图4-19所示。从图4-20中可以看出，虚拟机vTPM的vEK由EPS直接生成，vEK不可以迁移，而虚拟机vTPM的vSRK则由一般的存储密钥代替，即：duplicable storage key，且可迁移。SPS直接产生SRK而不是vSRK，各虚拟机vTPM以vSRK为根组成密钥子树，而所有虚拟机的密钥子树再以SRK为根组成一颗更大的密钥树。但该文献并没有讨论vAIK。



Fig.4-20 vTPM key structure based on key seed

图4-20 基于密钥种子的vTPM密钥结构

2015年，国内文献[56]提出了新一代TPM虚拟化框架设计：Ng-vTPM，其中就设计vTPM密钥组织结构，如图4-21所示。Ng-vTPM中不仅设计了虚拟背书密钥 vEK、虚拟存储根密钥vSRK、虚拟身份识别密钥 vAIK ( virtual attes tation identity key) ，而且还设计了签名密钥、加密密钥、绑定密钥、遗留密钥( legacy key)。其中，vEK 由物理TPM EPS产生，代表真实平台身份，用于标识虚拟机Ng-vTPM并可以抵御假冒的平台身份 。vSRK 为Ng- vTPM中密钥层次的根密钥，为一个可迁移非对称存储密钥，位于 vSRK 存储层次中的密钥都标识为可迁移密钥。vAIK 是虚拟机的身份证明密钥，其身份证书由 vEK 请求一次证书机关验证产生。每个密钥创建以及使用父密钥加密子密钥私密部分时，各自使用不同的授权会话，保证各个密钥间非授权不可访问。而且文中根据密钥的生命周期不同，将这些密钥分为了3 个组: v EK、内部可迁移组和外部不可迁移组。不过，将SRK存放于vTPM管理域，违背TCG规范，而且一个vTPM有多个vSRK，这也不符合TCG规范。

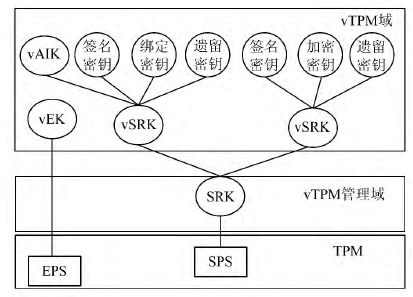
．

Fig.4-21 key structure of Ng-vTPM

图4-21 Ng-vTPM密钥结构

从上面的分析可以看出，vTPM的密钥管理研究，主要在如何设定密钥类型和属性、存储层次结构以及密钥生命周期管理三个方面展开。由于vTPM必须满足迁移需求，因此，在设定密钥类型与满足迁移需求之间始终存在两难，要么需要改变原本满足TCG规范的密钥属性，要么引入密钥冗余，均无法满足vTPM与TPM的密钥管理的一致性要求，降低了用户使用vTPM的方便性。另外，软件仿真型vTPM为虚拟机提供的可信计算服务，特别是密码运算服务均在虚拟平台的内存中（不是TPM内）完成，容易受到一般安全威胁、特权安全威胁和共享安全威胁，这将严重影响了虚拟机和vTPM的安全。

### 4.3 vTPM的证书信任扩展

vTPM的证书信任扩展，是指如何将物理TPM的证书信任关系扩展到vTPM证书，构造TPM到vTPM的证书信任链关系。

目前，国内外对vTPM的证书信任扩展进行了大量研究。文献[22]在研究中提出了四种构建vTPM证书链的设计思路，但最后一种与特殊硬件有关，为不失去一般性，此文不讨论。我们只讨论前三种：（1）vTPM vEK to hTPM AIK Binding，如图4-22所示，此方法中vEK和vAIK均由vTPM产生，vEK由TPM的AIK签名绑定，vAIK由Privacy CA 的私钥签名，验证方用Privacy CA 的公钥进行验证。此方法不仅将底层的证书信任扩展到了虚拟机，而且vTPM的证书结构与TPM一致，便于理解和已有成果的移植，但缺点包括两个方面，一是用TPM的AIK对vEK签名，违背了TCG规范，AIK只能对TPM内部产生的信息进行签名，而vEK是TPM的外部信息；二是AIK的有效期通常很短，AIK的失效导致对vEK的签名失效，从而导致vAIK失效，需要频繁向Privacy CA重新申请vAIK，因此Privacy CA的性能负担重。



图4-22 vTPM vEK to hTPM AIK Binding

（2）hTPM AIK signs vTPM vAIK，如图4-23所示，用TPM的AIK直接对vTPM的vAIK进行签名，不需要Privacy CA。此方法不仅将底层的证书信任扩展到了虚拟机，而且降低了Privacy CA的负担。但该方案依赖于TPM的AIK对vAIK签名，不仅同样违反了TCG规范，而且AIK的失效同样会导致其对vAIK的签名失效，从而会频繁更新AIK对vAIK的签名，增加性能负担。



图4-23 hTPM AIK signs vTPM vAIK

（3）Local CA issue vEK Certificate，即由本地证书机关（注：不是Privacy CA）为vTPM发布vEK，如图4-24所示。本方案的优点是vEK相对稳定，不会随着底层虚拟平台和TPM的变化而变化，缺点是不仅需要增加额外的证书机关，而且与TPM没发生绑定关系，即TPM的可信证书扩展没有传递到vTPM。



图4-24 Local CA issue vEK Certificate

文献[51]提出了“vTPM vEK to hTPM EK Binding”方案，即vEK由TPM的EK签名绑定，如图4-25所示。这个方案的优点是避免由于AIK的失效导致对vEK签名的失效，而且与TPM的绑定关系清楚、简单，将底层的TPM证书信任扩展到了虚拟机。但缺点是违反了TCG规范，即EK证书不能用于签名。



图4-25 vTPM vEK to hTPM EK Binding

另外，为了对前四种方案进行了改进，文献[52]提出了“vTPM vAIKto hTPM SK Binding”方案，即引入了签名密钥SK作为中介实现了AIK 对vAIK的间接签名，使得AIK不再对TPM外部数据进行签名，更好地满足TCG规范，而且vAIK的生成不依赖于PrivacyCA，降低了PrivacyCA的负担，如图4-26所示。但该方案不仅增加了生成 vAIK证书的复杂性，而且没有解决AIK 失效会导致vAIK失效，而重构vAIK需要重新生成SK，从而带来新的性能负担。另外，每一个vAIK证书对应一个SK，会产生大量的密钥冗余。



图4-26 vTPM vAIKto hTPM SK Binding

文献[56]结合TPM 2.0的新特性，提出“hTPM EPS Product vEK”方案，如图所示2-6所示。在该方案中，vTPM 的身份证书vAIK由 EPS 推导产生的vEK 向 Privacy CA验证生成。文中认为，基于vEK和EPS 的映射关系，就能够较为直接的标识虚拟机中的真实物理身份，建立从物理平台到虚拟平台的信任链。但实际上是不行的。EPS仅仅是Endorsement Key的基础密钥种子，用KDF算法生成Endorsement Key，虽然简单、容易，但仅仅是生成了vEK的密钥对而已，不存在信任扩展和传递的问题，外界不能通过这个密钥对的公钥推导出该公钥是底层TPM的EPS产生的。vTPM证书信任链的扩展仍然需要将EPS产生的vEK的密钥公钥、底层虚拟化平台的签名信息和vTPM的相关信息传递给Privacy CA生成vEK证书，再由vEK证书向Privacy CA生成vAIK来实现。



图4-27 hTPM EPS Product vEK

从上面的分析可以看出，无论采用什么方式将底层物理TPM的证书信任关系扩展到vTPM证书均不完善，要么存在违背TCG规范的情况，要么增加密钥冗余，要么增加Privacy CA的性能负担，有的方案甚至不能进行证书信任扩展。这将降低用户使用的方便性和信任。

### 4.4 vTPM迁移

vTPM迁移是指将vTPM从源虚拟平台迁移到目的虚拟平台。可以按照不同的分类方式对vTPM迁移进行分类，按照源虚拟平台和目的虚拟平台是否在同一云内，可分为vTPM云内迁移和vTPM云间迁移；按照源虚拟平台和目的虚拟平台是否相同，可分为vTPM同类迁移和vTPM异构迁移；按照迁移方式，可以分为静态迁移和动态迁移；目前，绝大多数研究采用的区分方式是静态迁移和动态迁移。因此，本节将按照vTPM静态迁移和动态迁移进行阐述。

#### 4.4.1 vTPM的静态迁移

所谓vTPM静态迁移，就是先锁定vTPM，停止vTPM服务，然后对vTPM相关状态数据进行迁移。目前的绝大多数研究主要是设计vTPM静态迁移协议，在已有的研究成果中包含vTPM单独迁移协议以及VM和vTPM捆绑迁移协议。无论是vTPM单独迁移协议还是VM和vTPM捆绑迁移协议，协议的基本步骤基本一致，唯一不同的是锁定vTPM的同时锁定VM，迁移vTPM相关状态数据的同时迁移VM的相关状态数据。因此，本节对于vTPM单独迁移以及VM-vTPM捆绑迁移，我们均视为同一类型的研究成果。

2006年文献[22]率先设计了vTPM的静态迁移协议，该迁移协议将源vTPM实例状态数据经过会话密钥加密安全传递到目标平台，由目标平台恢复vTPM实例，并删除源vTPM实例。如图4-28所示。首先，由平台迁移控制过程引擎在目标平台上创建一个状态为空的vTPM实例，生成与该实例关联的唯一标识符Nonce，并将Nonce加密传递给源平台。Nonce的目的是防止重放攻击和迁移目标的不确定性问题，在整个迁移过程中有效。然后，源平台将需要迁移的vTPM锁定，并与目标平台传递过来的Nonce关联。第三，源平台收集源vTPM的状态数据，包括：NVRAM、密钥会话授权和传输回话状态、授权数据、计数器以及其它相关永久标志位和数据等。为了保证状态数据在迁移过程的完整性和机密性，源平台将这些状态数据序列化并用非对称密钥签名，然后产生一对称密钥对序列化的状态数据和签名值进行加密生成迁移数据。其中对称密钥由源vTPM的父vTPM的某一存储密钥加密保护。第四，源平台删除掉源vTPM实例，并将迁移数据传递给目标平台；最后，目标平台迁解密出源vTPM的状态数据，并进行完整性验证和Nonce验证，最后恢复目标vTPM实例运行。值得注意的是，为了目标平台能够解密源vTPM的迁移数据，源平台的父vTPM存储密钥必须迁移到目标平台。

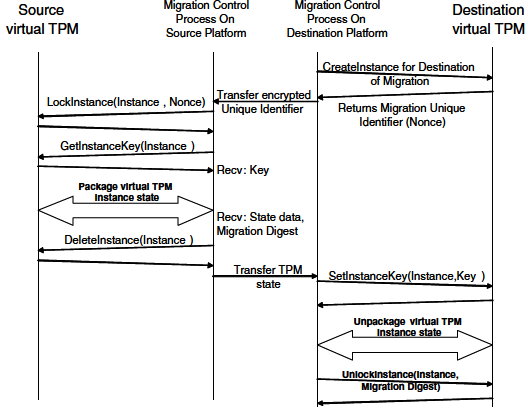


Fig.4-28. Static Migration protocol of vTPM

图4-28 vTPM静态迁移协议

2007年，文献[51]对文献[22]vTPM迁移协议进行了分析，并指出文献[22]在设计vTPM迁移协议时的三个不足，首先，文献[22]采用对称密钥加密源vTPM状态数据，为对称密钥提供保护的是源平台父vTPM的存储密钥。但该协议并没有对该存储密钥的迁移进行描述；其次，协议设计得不够完善。如对于目的平台vTPM的唯一标识Nonce的加密究竟采用什么密钥，协议中并没有说明；最后，无论是vTPM迁移还是密钥迁移，均属于TCG- IWG(Infrastructure Working Group:IWG)工作组下的DMTF（Data Migration Task Force: DMTF）任务组的工作范畴，但该协议设计时与任务组的相关工作基本没有交集。

2008年，文献[39]对文献[22]vTPM迁移协议进行了改进。与文献[22]最大的不同是对会话密钥的保护不再用父vTPM的存储密钥。当源迁移控制过程引擎初始化迁移且目标迁移控制过程引擎创建新的vTPM实例后，源vTPM要求在源平台和目标平台之间建立一条可信通道。通过该可信通道协商会话密钥。之后的过程和文献[22]基本相同，如图4-29所示。

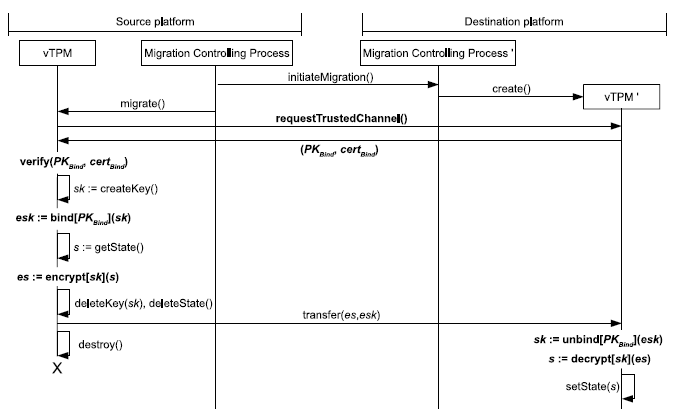


Fig.4-29 Static Migration protocol of vTPM based on trusted channel

图4-29 基于可信通道的vTPM静态迁移协议

2011年，文献[53]再次对文献[22]提出的静态迁移协议进行了改进，如图4-30所示。该文献提出了4阶段的VM-vTPM迁移过程。第一阶段建立TLS会话；第二阶段对源平台和目的平台进行证明；第三阶段是VM-vTPM状态数据迁移，启动目标平台的VM和vTPM，删除源平台的VM和vTPM；第四阶段是结束TLS会话。与文献[39]相比，增加了源和目的可信证明。

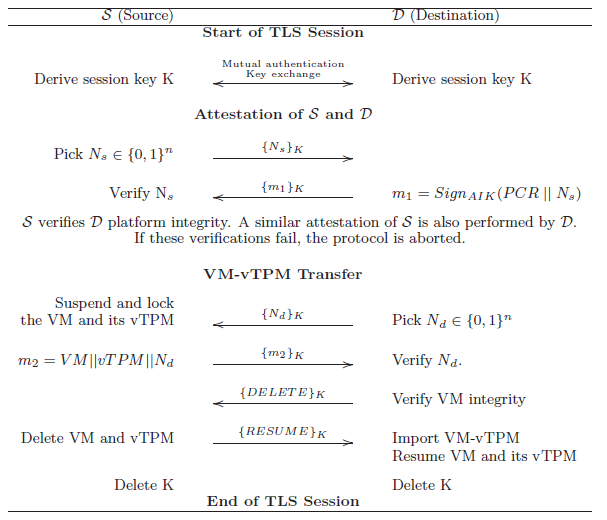


Fig.4-30 Static Migration protocol of vTPM based onTLS

图4-30基于TLS可信通道的VM-vTPM静态迁移协议

2012年，国内文献[54]对文献[22][39][53]进行改进。该文献提出了两阶段的vTPM迁移阶段，第一阶段是可信通道的建立，第二阶段是数据迁移。在可信通道建立阶段，需要进行相互鉴别、基于属性的身份证明、参数协商和会话密钥交换等。数据迁移阶段与文献[22][39]基本相同。如图4-31所示。该文献在设计该协议时，还考虑了源vTPM具有不可否认性、保证整过迁移过程的事务原子性等需求，比之前的研究成果更完善。同年，国内文献[55] 也提出了一种安全VM-v TPM迁移协议的设计与实现。包括4阶段，第一阶段是安全会话建立；第二阶段对目标平台进行远程证明；第三阶段vTPM及VM状态数据传输，第四阶段在元平台上删除并在目标平台上恢复VM及其vTPM。与之前已有的研究成果基本一致。

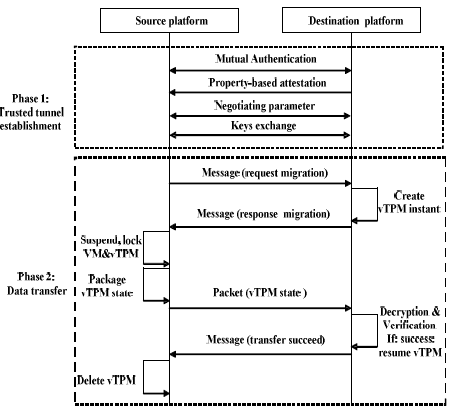


Fig.4-31 vTPM static migration protocol for high security requirements

图4-31 高安全需求的vTPM静态迁移协议

2015年，国内文献[56]基于TPM 2.0重新设计了vTPM迁移协议。该文献将vTPM迁移分两部分，其一是vTPM的密钥迁移；其二是vTPM的状态数据迁移。为了解决VM-vTPM的迁移时序问题，提出一个VM-vTPM实例对迁移协议，分为4个阶段: 双方身份认证阶段，双方远程证明阶段，数据传输阶段以及后续处理阶段。协议过程如图4-32所示。并进行了理论证明。但该文献对vTPM密钥迁移描述过于粗略。

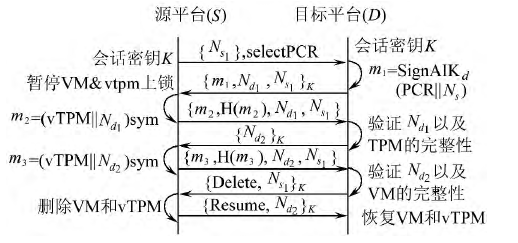


Fig.4-32 VM-vTPM static migration protocol

图4-32 VM-vTPM实例对静态迁移协议

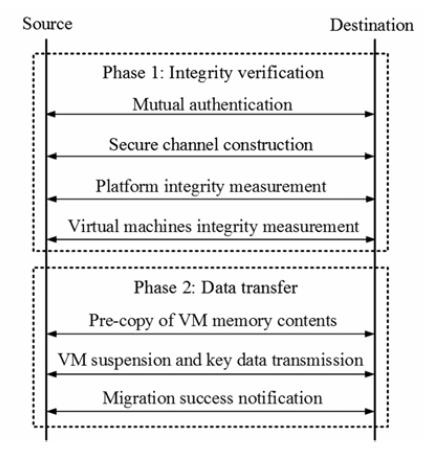
另外，文献[24]、[46]中也研究TPM的迁移。但这两个文献主要针对的是物理TPM的迁移，不是vTPM的迁移。因此在此不再分析。

综上所述，vTPM静态迁移协议取得了较多的研究成果，趋于成熟。但多数成果把vTPM的密钥结构作为vTPM状态数据之一进行迁移传输，这是不合理的，vTPM的密钥迁移必须符合TCG规范。另外，对于vTPM静态迁移，学术界研究成果较多，但产业界和开源社区相关的工程成果还很少。

#### 4.4.2 vTPM的动态迁移

所谓vTPM动态迁移，即在不停止vTPM服务的情况下进行迁移。目前，对VM的动态迁移成果较多，包括预拷贝、后拷贝以及基于检查点恢复和日志回放技术等。而vTPM动态迁移本质上也是内存拷贝技术，可以借鉴VM已有的研究成果。

2015年，文献[57]设计了VM-vTPM动态迁移协议。如图所示4-33所示。把VM-vTPM动态迁移协议分为两阶段，第一阶段是完整性验证阶段；第二阶段是数据迁移阶段。在第一阶段，包含源平台和目的平台的相互鉴别、安全通道的构建、平台的完整性度量以及虚拟机的完整性度量；第二阶段包括采用预拷贝方法迁移VM-vTPM、VM-vTPM锁定和密钥数据迁移以及迁移成功通告等。不过该方案并没有说明VM和vTPM内存拷贝时序，即是先拷贝VM，再拷贝vTPM，或先拷贝vTPM再拷贝VM，还是同时拷贝VM和vTPM。说明这一点很重要，因为vTPM是VM的可信服务提供者，且vTPM和VM属于不同的进程，如果同时拷贝，跨进程的内存拷贝涉及VMM中较多核心技术，有较大的难度。



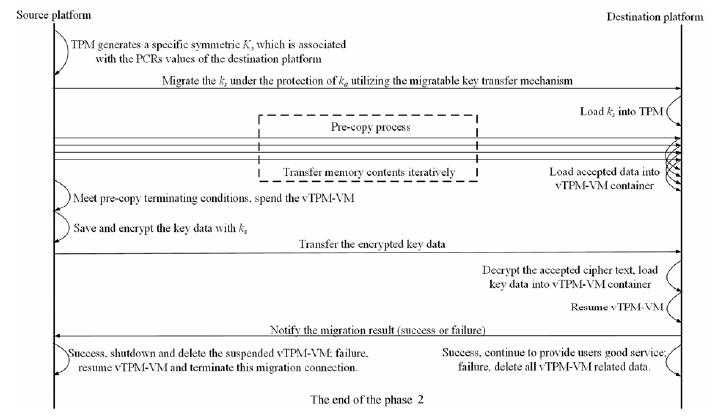
 Fig.4-33 VM-vTPM live migration protocol

图4-33 VM-vTPM动态迁移协议

除此以外，目前还没有看到vTPM动态迁移的其他研究成果。

#### 4.4.3 vTPM迁移系统

所谓vTPM迁移系统是指管理、控制和实施vTPM迁移的软件和硬件系统。通常包括实施迁移的角色、实体、架构及其互操作协议等。

在学术界，还没有看见公开发表的、实用的vTPM迁移系统。在产业界，无论是VMware的 vSphere 6，Microsoft的Hyper-V，还是Orical的VBox，均没有公开展示商用的vTPM迁移系统。在开源社区，只有Xen较好地给出了vTPM迁移。自Xen 3.2版以后，vTPM可运行在Unikernel的独立域里，该独立域实质就是一个轻量级虚拟机，可以采用Xen的虚拟机迁移系统对vTPM进行迁移，但Xen只能在同类平台之间迁移vTPM隔域。

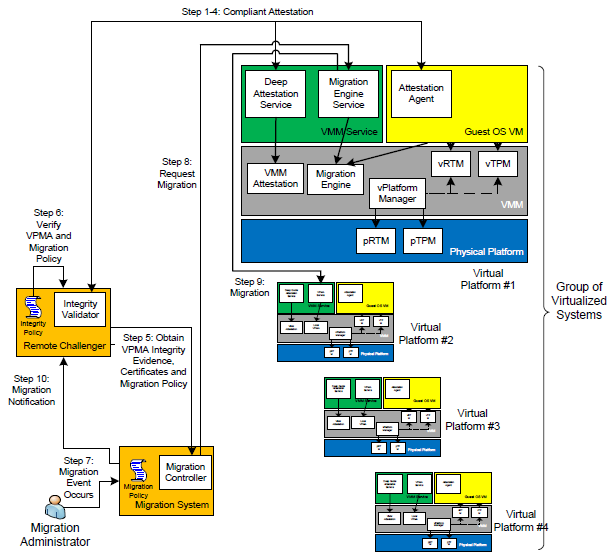


Fig.4-34 vTPM migration architecture of VPWG

图4-34 VPWG的vTPM迁移系统架构

2011年， TCG的VPWG（Virtual Platform Working Group:VPWG）在《Virtualized Trusted Platform Architecture Specification》[58]中提出了可信虚拟平台下的vTPM迁移系统架构及其迁移协议。如图。该迁移架构一共有12个实体，包括迁移控制（Migrate Controller）、完整性校验(Integrity Validator)、VMM证明服务（Deep Attestation Service）、VMM证明（VMM Attestation）、虚拟机证明服务（Attestation Service）迁移引擎服务（Migration Engine Service）、迁移引擎（Migration Engine）、虚拟平台管理器（vPlatform Manager）、虚拟可信根（vRTM）、虚拟TPM（vTPM）以及物理可信根（pRTM）和物理TPM（pTPM）。涉及的角色包括远程挑战者、迁移管理者等。迁移协议的前4步是由迁移挑战者对源平台与目标平台进行兼容性证明；第5、6步是迁移挑战者中的完整性校验者(Integrity Validator)从迁移控制（Migrate Controller）中获取VPAM完整性证据、证书以及迁移策略并进行验证。第7步迁移管理者一旦接收到迁移事件发生，第8步就会向迁移引擎发出迁移指令，第9步迁移引擎对需要迁移的VM-vTPM进行迁移，迁移完成后，由迁移管理者通知远程挑战者迁移成功。与已有的研究成果相比，该规范详细设计了迁移中涉及的各个实体，更接近实用系统，而迁移协议本身论述较少。

总结起来，在vTPM迁移方面，对vTPM静态迁移协议的研究基本成熟，而对vTPM动态迁移还很少，特别是vTPM和VM捆绑迁移时内存拷贝的时序问题还没有解决。而实用、成熟的vTPM迁移系统还没有出现，急需要进一步研究和开发。

5. TPM虚拟化亟待解决的问题与挑战

目前，各种TPM虚拟化方案均存在不同的缺点和优点，但在安全性、性能以及遵守TCG规范方面还存在诸多需要亟待解决的问题。主要包括：

（1）在TPM虚拟化系统架构上，缺少一种高安全、高性能且能够根据用户安全需求动态调整的智能自适应TPM虚拟化架构。另外，对于聚合型TPM虚拟化架构的研究也很滞后，我们的思维不能局限在TCG规范的细节上，应该借鉴可信计算的基本思想，在技术方面大胆创新。

（2）在vTPM的密钥管理方面，存在的问题和挑战包括两个方面，一方面，由于vTPM必须满足迁移需求，因此，在设定密钥类型与满足迁移需求之间始终存在两难，要么需要改变原本满足TCG规范的密钥属性，要么引入密钥冗余，均无法满足vTPM与TPM的密钥管理的一致性要求，降低了用户使用vTPM的方便性。另一方面，对于软件仿真型vTPM为虚拟机提供的可信计算服务，特别是密码运算服务均在虚拟平台的内存中（不是TPM内）完成，这容易受到一般安全威胁、特权安全威胁和共享安全威胁，这将严重影响了用户虚拟机和vTPM的安全。

（3）在证书信任扩展方面，为了将物理TPM的证书信任关系扩展到vTPM证书，无论采用何种方式，要么存在证书使用违背TCG规范的情况，要么引入密钥冗余，还未找到令人满意的解决方案。

（4）在vTPM迁移方案，目前的研究主要集中在云内相同虚拟平台之间的vTPM静态迁移协议。但云间异构虚拟平台之间的vTPM静态迁移协议、vTPM的动态迁移协议以及vTPM迁移系统等，均还需要进一步的研究和开发。

6 结束语.

可信计算与云计算技术的结合可以在一定程度上保证云计算环境的可信，促进云计算在更广的应用范围拓展和延拓，其中通过TPM虚拟化来构建虚拟机可信环境是解决此问题的有效办法之一，如何安全、高效率地对TPM进行虚拟化以及提高用户对云环境的信任是可信计算和云计算融合发展的关键。同时，TPM虚拟化不仅仅是技术问题，它还涉及标准化、监管模式、应用模式等诸多方面，因此，仅仅从技术角度探索TPM虚拟化是不够的，需要信息安全学术界、产业界和政府相关部门的共同努力才能实现。

参考文献

[1] Trusted Computing Group, <https://www.trustedcomputinggroup.org>[EB/OL],2017.

[2]Trusted Computing Group. TPM Specification, Version 2.0/[Part 1: Architecture](http://www.trustedcomputinggroup.org/wp-content/uploads/TPM%20Rev%202.0%20Part%201%20-%20Architecture%2001.35_Public_Review.pdf)[EB/OL], https://www.Trusted- computinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part1-Architecture-01.36\_public-review.pdf, 2017.

[3] Trusted Computing Group. TPM Specification, Version 2.0/ [Part 2: Structures](http://www.trustedcomputinggroup.org/wp-content/uploads/TPM%20Rev%202.0%20Part%202%20-%20Structures%2001.35_Public_Review.pdf) [EB/OL], https://www.Trusted- computinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part2-Structures-01.36\_public-review.pdf, 2017.

[4] Trusted Computing Group. TPM Specification, Version 2.0/ [Part 3: Commands](http://www.trustedcomputinggroup.org/wp-content/uploads/TPM%20Rev%202.0%20Part%203%20-%20Commands%2001.35-code_Public_Review.pdf)[ EB/OL], https://www.trusted- computinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part3-Commands-01.36-code\_public-review.pdf, 2017.

[5]Trusted Computing Group.TPM Specification, Version 2.0/[Part 4: Supporting Routines](http://www.trustedcomputinggroup.org/wp-content/uploads/TPM%20Rev%202.0%20Part%204%20-%20Supporting%20Routines%2001.35-code.pdf)[EB/OL],https://www.- trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part3-Commands-01.36-code\_public-review.pdf, 2017.

[6] 沈昌祥,张焕国,冯登国等,信息安全综述[J],中国科学E辑: 信息科学, 37(2):129-150, 2007.

[7] 沈昌祥,张焕国,王怀民,王戟,赵波,严飞,余发江,张立强,徐明迪. 可信计算的研究与发展[J]. 中国科学E辑:信息科学, 40(2):139-166, 2010.

[8] 冯登国,秦宇,汪丹,初晓博. 可信计算技术研究[J]. 计算机研究与发展, 48(8): 1332-1349, 2011.

[9] 国家密码管理局, 可信计算密码支撑平台功能与接口规范,2007.

[10] B.P.Rimal, E.Choi, I.Lumb.A Taxonomy and Survey of Cloud Computing Systems[C]. In Proceedings of the 2009 Fifth iInternational Joint Conference on INC,IMS and IDC在, IEEE Computer Society, 44–51,2009.

[11] M.Armbrust,A.Fox,R.Grith,A.D.Joseph,R.Katz,A.Konwinski,G.Lee,D.Patterson,A.Rabkin,I.Stoica,et al. A view of cloud computing[J]. Communications of the ACM, 53(4):50–58, 2010.

[12][任艳丽](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e4%bb%bb%e8%89%b3%e4%b8%bd&code=25980509;11563843;10966720;09597721;08515643;),[丁宁](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e4%b8%81%e5%ae%81&code=25980509;11563843;10966720;09597721;08515643;),[王天银](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e7%8e%8b%e5%a4%a9%e9%93%b6&code=25980509;11563843;10966720;09597721;08515643;), [陆海宁](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e9%99%86%e6%b5%b7%e5%ae%81&code=25980509;11563843;10966720;09597721;08515643;),[谷大武](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e8%b0%b7%e5%a4%a7%e6%ad%a6&code=25980509;11563843;10966720;09597721;08515643;), 可完全验证的双线性对运算外包算法[J],中国科学E辑:信息科学,46(7): 855-869, 2016.

[11] [沈尧](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e6%b2%88%e5%b0%a7&code=),[秦小麟](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e7%a7%a6%e5%b0%8f%e9%ba%9f&code=),[鲍芝峰](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e9%b2%8d%e8%8a%9d%e5%b3%b0&code=);一种云环境中数据流的高效多目标调度方法[J],软件学报,28(3): 1-19, 2017.

[12] [徐思尧](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e5%be%90%e6%80%9d%e5%b0%a7&code=35365352;07555417;35365353;),[林伟伟](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e6%9e%97%e4%bc%9f%e4%bc%9f&code=35365352;07555417;35365353;),[王子骏](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e7%8e%8b%e5%ad%90%e9%aa%8f&code=35365352;07555417;35365353;),基于负载高峰特征的虚拟机放置算法[J],软件学报,27(7): 1876-1887, 2016.

[13] [王焘](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e7%8e%8b%e7%84%98&code=),[顾泽宇](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e9%a1%be%e6%b3%bd%e5%ae%87&code=),[张文博](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e5%bc%a0%e6%96%87%e5%8d%9a&code=),[徐继伟](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e5%be%90%e7%bb%a7%e4%bc%9f&code=),[魏峻](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e9%ad%8f%e5%b3%bb&code=),[钟华](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e9%92%9f%e5%8d%8e&code=),一种基于自适应监测的云计算系统故障检测方法[J],计算机学报,39(163):1-15, 2016.

[14] [王国峰](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e7%8e%8b%e5%9b%bd%e5%b3%b0&code=),[刘川意](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e5%88%98%e5%b7%9d%e6%84%8f&code=),[潘鹤中](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e6%bd%98%e9%b9%a4%e4%b8%ad&code=),[方滨兴](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e6%96%b9%e6%bb%a8%e5%85%b4&code=),云计算模式内部威胁综述[J],计算机学报, 39(145): 1-21, 2016.

[15] [任丽芳](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e4%bb%bb%e4%b8%bd%e8%8a%b3&code=36165543;10912614;08402641;30065851;),[王文剑](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e7%8e%8b%e6%96%87%e5%89%91&code=36165543;10912614;08402641;30065851;),[许行](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e8%ae%b8%e8%a1%8c&code=36165543;10912614;08402641;30065851;),不确定感知的自适应云计算服务组合[J],计算机研究与发展,53(12):2867-2881, 2016.

[16] [梁俊杰](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e6%a2%81%e4%bf%8a%e6%9d%b0&code=07367845;32902297;06292142;31971641;33153126;),[李凤华](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e6%9d%8e%e5%87%a4%e5%8d%8e&code=07367845;32902297;06292142;31971641;33153126;),[刘琼妮](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e5%88%98%e7%90%bc%e5%a6%ae&code=07367845;32902297;06292142;31971641;33153126;),[尹利](http://202.115.193.42/rwt/42/http/P75YPLUDN3WXTLUPMW4A/KCMS/detail/%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20%20/kcms/detail/search.aspx?dbcode=CJFQ&sfield=au&skey=%e5%b0%b9%e5%88%a9&code=07367845;32902297;06292142;31971641;33153126;),MapReduce框架下的优化高维索引与KNN查询[J],电子学报,44(8): 1873-1880, 2016.

[17] Amazon Elastic Compute Cloud (EC2). [http://aws.amazon.com/ec2/[EB/OL](http://aws.amazon.com/ec2/%5bEB/OL)], 2016.

[18] Google App Engine(GAE). [https://appengine.google.com/[EB/OL](https://appengine.google.com/%5bEB/OL)], 2016.

[19] Microsoft Azure Services Platform. [http://www.microsoft.com/azure/[EB/OL](http://www.microsoft.com/azure/%5bEB/OL)], 2016.

[20]Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems (Eucalyptus). [http://www.eucalyptus.com/[EB/OL], 2016](http://www.eucalyptus.com/%5bEB/OL%5d,%202016).

[21] 林闯,苏文博,孟坤,刘渠,刘卫东.云计算安全:架构、机制与模型评价[J], 计算机学报, 36(9):1765-1784, 2013.

[22]张玉清,王晓菲,刘雪峰,刘玲. 云计算环境安全综述[J]. 软件学报, 27(6):1328-1348, 2016.

[22] [S Berger](http://xueshu.baidu.com/s?wd=author%3A%28Berger%2C%20Stefan%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[R Cáceres](http://xueshu.baidu.com/s?wd=author%3A%28C%C3%A1ceres%2C%20Ram%C3%B3n%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[KA Goldman](http://xueshu.baidu.com/s?wd=author%3A%28Goldman%2C%20Kenneth%20A.%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[R Perez](http://xueshu.baidu.com/s?wd=author%3A%28Perez%2C%20Ronald%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[R Sailer](http://xueshu.baidu.com/s?wd=author%3A%28Sailer%2C%20Reiner%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson),etal. vTPM :Virtualizaing the trusted platform modual[C], Proceedings of the 15th USENIX security Symposium. Vancouver, Canada, 305-320,2006.

[23] Paul England,Jork Loeser,Para-Virtualized TPM Sharing[C],TRUST 2008,119-132, 2008.

[24] STUMPF F,ECKERT C. Enhancing trusted platform modules with hardware-based virtualization techniques[C]. Proceedings of the 2008 Second International Conference on Emerging Security Information, 1-9, 2008.

[25][B. AlBelooshi](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.B.%20AlBelooshi.QT.&newsearch=true); [K. Salah](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.K.%20Salah.QT.&newsearch=true); [T. Martin](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.T.%20Martin.QT.&newsearch=true); [E. Damiani](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.E.%20Damiani.QT.&newsearch=true), [Securing Cryptographic Keys in the IaaS Cloud Model](http://ieeexplore.ieee.org/document/7431439/)[C], [2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7430473), IEEE Conference Publications: 397 – 401, 2015.

[26] [Zhilou Yu](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Zhilou%20Yu.QT.&newsearch=true); [Qiao Wang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Qiao%20Wang.QT.&newsearch=true); [Weipin Zhang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Weipin%20Zhang.QT.&newsearch=true); [Hongjun Dai](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Hongjun%20Dai.QT.&newsearch=true), [A Cloud Certificate Authority Architecture for Virtual Machines with Trusted Platform Module](http://ieeexplore.ieee.org/document/7336360/)[C], [2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS)](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7335977) , IEEE Conference Publications:1377-1380,2015.

[27] [Dexian Chang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Dexian%20Chang.QT.&newsearch=true); [Xiaobo Chu](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Xiaobo%20Chu.QT.&newsearch=true); [Yu Qin](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Yu%20Qin.QT.&newsearch=true); [Dengguo Feng](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Dengguo%20Feng.QT.&newsearch=true),[TSD: A Flexible Root of Trust for the Cloud](http://ieeexplore.ieee.org/document/6295966/)[C],[2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6294581) , IEEE Conference Publications: 119 - 126, 2012.

[28]  [Xin Wan](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Xin%20Wan.QT.&newsearch=true); [Zhiting Xiao](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Zhiting%20Xiao.QT.&newsearch=true); [Yi Ren](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Yi%20Ren.QT.&newsearch=true),[Building Trust into Cloud Computing Using Virtualization of TPM](http://ieeexplore.ieee.org/document/6405630/)[C],[2012 Fourth International Conference on Multimedia Information Networking and Security](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6403795),Year: 2012, IEEE Conference Publications: 59 - 63,2012.

[29]  [Xue Dongliang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Xue%20Dongliang.QT.&newsearch=true); [Wu Xiaolong](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Wu%20Xiaolong.QT.&newsearch=true); [Gao Yunwei](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Gao%20Yunwei.QT.&newsearch=true); [Song Ying](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Song%20Ying.QT.&newsearch=true); [Tian Xinhui](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Tian%20Xinhui.QT.&newsearch=true); [Li Zhaopeng](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Li%20Zhaopeng.QT.&newsearch=true),[TrustVP: Construction and Evolution of Trusted Chain onVirtualization Computing Platform](http://ieeexplore.ieee.org/document/6405916/)[C], 2012 Eighth International Conference on [Computational Intelligence and Security (CIS),](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6403635) , IEEE Conference Publications: 623 - 630,2012.

[30] Microsoft MVP, <http://anilerduran.com/vtpm-in-windows-server-2016-hyper-v/>[EB/OL],2017.

[31] Oricale, [https://www.virtualbox.org/[EB/OL](https://www.virtualbox.org/%5bEB/OL)], 2017.

[32]VMware, [http://www.vmware.com/[EB/OL](http://www.vmware.com/%5bEB/OL)], 2017.

[33] Xen Project, [http://www.xenproject.org/[EB/OL],](http://www.xenproject.org/%5bEB/OL%5d,) 2017.

[34]KVM project, [http://www.linux-kvm.org/[EB/OL], 2017](http://www.linux-kvm.org/%5bEB/OL%5d,%202017).

[35] Scarlata, Vincent and Rozas, Carlos and Wiseman, Monty and Grawrock, David and Vishik, Claire, TPM Virtualization: Building a General Framework [M], Trusted Computing, 2007.

[36] Anderson, Melvin J. and Moffie, Micha and Dalton, Chris I.: " Towards Trustworthy Virtualization Environments: Xen Library OS Security Service Infrastructure"[R] , Hewlett-Packard Laboratories, 43-51,2007.

[37] Murray D G, Milos G, Hand S. Improving Xen Security through Disaggregation [C]. VEE '08: Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, 151-160,2008.

[38]David Plaquin, Serdar Cabuk, Chris Dalton, Dirk Kuhlmann, Philipp Grete,Carsten Weinhold, Alexander Böttcher,Derek Murray, Theodore Hong,Marcel Winandy, TPM Virtualisation Architecture document[R], Open Trusted Computing, 2009.

[39] Sadeghi, Ahmad-Reza and Stuble, Christian and Winandy, Marcel, Property-Based TPM Virtualization[C], in ISC '08: Proceedings of the 11th international conference on Information Security,1-16, 2008.

[40]Xin JIN 1, Li-na WANG 1,2, Rong-wei YU 1, Peng KOU 1, Cheng-lin SHEN 1, Administrative Domain: Security Enhancement for Virtual TPM[C], 2010 International Conference on Multimedia Information Networking and Security,767-771,2010.

[41] 代炜琦，云计算执行环境可信构建关键问题研究[D],华中科技大学,2015.

[42] S. A. Bade, L. N. Betz, A. G. Kegel, M. J. Kelly, and W. L. Terrell,“Method and System for Virtualization of Trusted Platform Modules,”US Patent 7 380 119, May, 2008.

[43] [T Feller](http://xueshu.baidu.com/s?wd=author%3A%28Thomas%20Feller%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[S Malipatlolla](http://xueshu.baidu.com/s?wd=author%3A%28Sunil%20Malipatlolla%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[M Kasper](http://xueshu.baidu.com/s?wd=author%3A%28Michael%20Kasper%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson)，[SA Huss](http://xueshu.baidu.com/s?wd=author%3A%28Sorin%20A.%20Huss%29%20&tn=SE_baiduxueshu_c1gjeupa&ie=utf-8&sc_f_para=sc_hilight%3Dperson), dcTPM: A Generic Architecture for Dynamic Context Management[C]，2011 International Conference on Reconfigurable Computing and FPGAs, 211-216,2011.

[44] N. M. Smith,“Method and apparatus for virtualization of a multi-contexthardware trusted platform module (tpm),” US Patent 2009/0 055 641 A1, February, 2009.

[45] [Ramya Jayaram Masti](http://dl.acm.org/author_page.cfm?id=81504687728&CFID=700391978&CFTOKEN=37668849), [Claudio Marforio](http://dl.acm.org/author_page.cfm?id=81552074756&CFID=700391978&CFTOKEN=37668849), [Srdjan Capkun](http://dl.acm.org/author_page.cfm?id=81100396056&CFID=700391978&CFTOKEN=37668849)， An Architecture for Concurrent Execution of Secure Environments in Clouds[C]，Proceedings of the 2013 ACM workshop on Cloud computing security workshop，11-22, 2013.

[46] Jiun Yi Yap， Allan Tomlinson，Para-Virtualizing the Trusted Platform Module: An Enterprise Framework Based on Version 2.0 Specification，International Conference on Trusted Systems INTRUST 2013: [Trusted Systems](http://link.springer.com/book/10.1007/978-3-319-03491-1) pp 1-16,2013.

[47] PCI Sig, Pci-sig-single root iov. http://www.pcisig.com/specifications/iov/Single\_root/[EB/OL], 2017.

[48] 刘明达，马龙宇, 一种基于SR-IOV 技术的虚拟环境安全隔离模型[J],信息网络安全, 9:84-89, 2016.

[49] [Xinlong Liang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Xinlong%20Liang.QT.&newsearch=true); [Rui Jiang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Rui%20Jiang.QT.&newsearch=true); [Huafeng Kong](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Huafeng%20Kong.QT.&newsearch=true),  [Secure and reliable VM-vTPM migration in private cloud](http://ieeexplore.ieee.org/document/6743327/)[C],[2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA)](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6732241), 510-514, 2013.

[44] 杨永娇,严飞，毛军鹏，张焕国，Ng-vTPM: 新一代TPM虚拟化框架设计[J]，武汉大学学报（理学版）61(2):103-111,2015.

[51] R Goyette, [A review of " vTPM: Virtualizing the Trusted Platform Module"](http://richardgoyette.com/Papers/Virtual%20TPM%20Critique%20intro.pdf)[R]，Network Security and Cryptography Symposium,1-17,2007.

[52]王丽娜,高汉军,余荣威,任正伟,董永峰,[基于信任扩展的可信虚拟执行环境构建方法研究](http://222.196.199.233/kns50/detail.aspx?QueryID=20&CurRec=1)[J],通信学报,32(9):1-8, 2011.

[53] Boris Danev, Ramya Jayaram Masti, Ghassan O. Karame and Srdjan Capkun, Enabling Secure VM-vTPM Migration in Private Clouds[C], ACSAC’11, 187-196, Orlando, Florida USA, 2011.

[54]] Xin Wan, XinFang Zhang, Liang Chen, JianXin Zhu, An Improved vTPM Migration Protocol Based Trusted Channel[C], 2012 International Conference on Systems and Informatics (ICSAI 2012),870-875,2012

[55] 于颖超,刘了,陈左宁, 一种安全VM-v TPM迁移协议的设计与实现[J], 电子技术应用,38(4):130-133, 2012.

[56] 杨永娇,严飞，毛军鹏，张焕国，Ng-vTPM: 新一代TPM虚拟化框架设计[J]，武汉大学学报（理学版）61(2):103-111,2015.

[57] FAN Peiru, ZHAO Bo†, SHI Yuan, CHEN Zhihong, NI Mingtao, An Improved vTPM-VM Live Migration Protocol, Wuhan University Journal of Natural Sciences,20(6): 512-520, 2015.

[58] Trusted Computing Group. Virtual Platform Working Group:VPWG）在《Virtualized Trusted Platform Architecture Specification,[EB/OL], <https://www.trustedcomputinggroup.org/wp-content/uploads/>TCG\_ VPWG\_Architecture\_V1-0\_R0-26\_FINAL.pdf, 2017.

[59] 丁滟, 王怀民, 史佩昌, 吴庆波, 戴华东, 富弘毅, 可信云服务[J],计算机学报, 38(5):133-149,2015.

1. 一般安全威胁、特权安全威胁和共享安全威胁的相关内涵参见文献[59] [↑](#footnote-ref-1)
2. 尽管S Berger 是IBM T. J. Watson Research Center的研究员，但发表的成果属于学术论文，所以仍将之归于学术界 [↑](#footnote-ref-2)