文章编号:1002-1175(2011)05-0648-11

TPM 虚拟域安全模型*

秦 宇†,兰海波

(中国科学院软件研究所信息安全国家重点实验室, 北京 100080)

(2010 年 9 月 1 日收稿;2010 年 10 月 17 日收修改稿)

Qin Y, Lan H B. TPM security model for virtual domains[J]. Journal of Graduate University of Chinese Academy of Sciences, 2011,28(5):648 – 658.

摘 要 针对 TPM 访问控制机制无法直接应用于虚拟计算、云计算等环境的问题,重点分析 TPM 内部对象间依赖关系,并结合虚拟域的安全需求,建立 TPM 虚拟域安全模型. 该模型对 TPM 对象的访问请求增加了虚拟域的完整性、机密性等安全约束,解决了多虚拟域环境下的 TPM 对象的创建、使用、销毁等问题. 还进一步对该模型的安全规则进行了相关逻辑分析,并通过实际原型系统的测试,证明了 TPM 虚拟域安全模型的实施对可信虚拟平台的性能影响非常小.

关键词 TCG,TPM 安全模型,虚拟化,虚拟域,安全级

中图分类号 TP393

国际可信计算组织 TCG 制定并发布了 TPM 规范[1]、TSS 规范[2]、TNC 规范[3]等系列标准,从硬件、软件栈、网络等各个方面确立了可信计算安全体系结构. TPM 作为可信计算的专用安全芯片,已被广泛地装配于安全主机、笔记本电脑等计算平台中. 但是可信计算的安全机制并不够完善,众多科研机构对 TCG 框架下的安全机制进行了深入研究.

TPM 芯片安全方面,Danilo 等[4]研究了 TPM 的 OIAP 会话访问机制,发现其存在重放攻击的风险. 文献[5]研究了 TPM 的 LPC 总线上实施物理攻击破坏 PCR 访问保护机制,它能够导致 PCR 寄存器被非法地重置初态. 可信系统方面,IBM 研究院在可信 Linux 客户端方案[6]中,把 Biba[7],BLP[8]强制访问控制模型和 TPM 实现的完整性度量相结合,将低水印访问控制[9]应用到系统进程的强制访问控制中,TPM 为可信 Linux 客户端提供真实的完整性度量和判定,极大地增强了原有的系统强制访问控制机制. 然后 Trent Jaeger 等提出了基于信息流的完整性度量 PRIMA 方案[10],用轻量级的 CW – Lite 完整性模型[11]克服加载时完整性级控制的缺陷,对系统的完整性信息流进行验证,现已完成 PRIMA 应用于 SELinux 的原型实现. 在这方面国内学者提出了基于可信计算的保密性和完整性统一的解决方案[12]. 可信网络方面,SandHu 等将可信计算应用到分布式 P2P,网格计算环境,用 TPM 增强可信引用监控机功能,提出了基于可信计算的分布式访问控制框架[13]. TCG 组织、微软和 Cisco 各自拥有可信网络连接解决方案,分别是 TNC、NAC(network access control)和 NAP(network access protection),目前的趋势是三者相互融合,共同发展.

随着虚拟计算、云计算的飞速发展,可信计算与之相结合的趋势越来越明显. 在这些新的计算场景下,原有的 TPM 在通用 PC 平台上的安全模型面临很大的挑战. 在虚拟计算和云计算环境中,可信计算

* 国家科技支撑计划项目(2008BAH22B06)和中国科学院知识创新工程前沿项目(ISCAS2009 – DR14,ISCAS2009 – GR03)资助

†E-mail: qin_yu@ is. iscas. ac. cn

平台被分割为若干隔离的虚拟机,或者称之为虚拟域,这就需要 TPM 芯片无冲突地为这些虚拟域提供可信计算服务,因此需要建立新型的 TPM 安全模型. IBM 和 Intel 通过在虚拟机管理域中构建全虚拟化的 VTPM[14-15]提供可信计算服务,该方法解决了虚拟环境 TPM 应用的部分问题. Microsoft 则使用 TPM 半虚拟化技术[16],构建不同虚拟域的 TPM Context 和 Virtual PCR,最大可能地为多个虚拟域提供基于硬件 TPM 保护的可信计算服务.

从以上研究背景可以看出 TPM 对于系统安全增强提供了强有力的支持,特别是在系统完整性和完整性信息流控制具有极大的优势. 但是在虚拟计算、云计算环境下,系统用户除了需要满足特定的系统安全级、拥有相应的授权数据外,还必须对不同虚拟域的 TPM 访问请求进行限制,解决 TPM 内部资源虚拟域隔离的问题. 本文将在 Microsoft 的 TPM 方案基础上,结合 TPM 的各种安全机制讨论在虚拟计算环境中 TPM 内部对象的访问控制机制,研究 TPM 的虚拟域安全模型.

1　TPM 安全机制及相关问题

1.1　TPM 安全机制分析

TPM 安全芯片物理上保护其内部的密钥、数据、PCR 等,TPM 规范以授权数据[1]、Locality 机制[17]、会话等限制外部程序的非授权访问. TPM 内部对象的访问主要基于授权数据、Locality 和委托[1](Delegation),其中授权数据是判定是否具有 TPM 内部对象操作权限最主要的依据,而 Locality 和 Delegation 是 TPM 1.2 规范引入,Locality 主要用于对 TPM 内部 PCR 寄存器的访问控制,Delegation 则是对 TPM owner 权限的委托,后二者在 TPM 访问控制中较少使用. TCG 规范中定义的 TPM 内部实体对象的访问规则为:

1)访问主体的 Locality 必须满足 TPM 内部对象所要求的 Locality 等级,且在这个 Locality 下该操作允许执行;

2)访问主体具有 TPM owner 委托的权限,验证其委托授权数据(delegated AuthData)成功后,才允许其授权操作;

3)访问主体知道 TPM 内部对象的授权数据,才允许访问相应 TPM 对象.

上述 TPM 访问规则适合于任意 TPM 操作,一个 TPM 操作过程包含 3 个要素:操作、操作者、操作对象. TPM 的操作者称之为 TPM 的访问主体(记为 s),本文中讨论的主体是访问 TPM 的进程. 主体访问的 TPM 内部实体对象称之为客体(记为 o),TPM 内部的实体对象部分是固定不变的,如 PCR 寄存器、NV 存储区、SHA1 引擎、随机数发生器等,有一部分可以创建新的实体,如密钥、单向计数器、会话等. 将所有主体的集合记为 S,所有操作的集合记为 A,所有客体的集合记为 O,那么 TPM 的操作过程是定义在 $S \times A \times O$的关系.

TPM 内部实体对象集 O 包括:密钥、PCR、Counter、Session、NV 对象、owner、公用对象(SHA-1 Engine,随机数生成等). TPM 内部对象以 Locality 和授权数据验证的方式实施访问控制,表 1 描述了 TPM 现有的访问控制方法.

表 1　TPM 对象授权方式

TPM 对象	Locality 认证	授权数据认证
密钥	×	√
PCR	√	×
单向计数器	×	√
会话	×	×
owner *	√	√
NV 存储	√	√
公用对象	×	×

* owner 对象是个复杂的客体对象,部分管理功能使用 Locality 认证,一部分使用授权数据认证.

TPM 内部实体对象的基本操作定义为下列几种:①只读操作,只读出数据但不做其他操作,记为 r,常见的读取 PCR,读取 NV 区域,读取 Counter 值都属于这类操作. ②创建操作,记为 c,例如:TPM 创建新的密钥、会话对象,关闭或销毁这些已存在的对象. ③读写操作,记为 w,例如 PCR 的扩展、Counter 的值增加等操作. ④执行操作,记为 x,这类操作比较多,如使用密钥加密、签名,封装,TPM_Reset,owner 委托权限. 因此主体对客体执行 r、c、w、x 等几类基本操作,$A = \{r,c,w,x\}$. 主体 s 对客体 o 进行了 q 操作可表示为 $q(s,o)$,

其中，$q \in A$.

为了便于描述 TPM 访问规则，定义进程主体 s 拥有 TPM 对象 o，即 s 知道 o 的授权数据，记为$s \ni o$. 对于 PCR，TPM 公用对象（SHA - 1 Engine，随机数等）等无授权数据 TPM 对象（记为 po），满足 $\forall (s \in S) s \ni po$. TPM 的访问规则可描述为：

1）$\mathrm{LOC}(s) \geqslant \mathrm{LOC}(o)$：LOC 表示对象的 Locality 级别，要求主体 s 的 Locality 级不小于客体 o 的 Locality.

2）$\lambda(q(s,o)), q \in A = \{r,c,w,x\}$：TPM 所有者对 Owner 权限的委托，$\lambda$ 为委托指派，所有者赋予 s 对 TPM 客体 o 拥有执行操作 q 的权限.

3）$s \ni o$：主体 s 知道客体 o 的授权数据，拥有客体 o.

$$\forall (s \in S) \forall (o \in O) \forall (q \in A)(\mathrm{LOC}(s) \geqslant \mathrm{LOC}(o)) \wedge ((s \ni o) \vee \lambda(q(s,o))) \Rightarrow q(s,o). \tag{1}$$

1.2 TPM 虚拟域相关问题

按照现有的 TCG 规范，TPM 将外部用户区分为 2 类：TPM 所有者（TPM owner）和 TPM 用户（TPM user），TPM 所有者是拥有 TPM 所有权的实体，一个平台只能拥有一个所有者；而 TPM 用户能够加载和使用 TPM 密钥等实体对象. TPM 没有相应的用户管理，不维护 TPM 用户列表，判定一个用户是否具有 TPM 对象访问权限的方法是验证 TPM 对象的 AuthData（授权数据，或称认证数据）. 由于 TPM 不标识、不限制任何主体访问，因而对于系统平台任何程序只要满足 TPM 访问规则，都能访问 TPM 内部对象. TPM 内部对象之间存在严格的依赖关系，子密钥是由父密钥创建，子密钥的授权数据受父密钥保护. 一旦父密钥的完整性遭受破坏，子密钥将无法使用.

无论是 BLP，Biba 访问控制模型，还是 NGSCB[19] 划分为保护分区、普通分区，Terra[20]、Xen[21] 等将系统隔离为多个不同等级的安全区域，其实质都是将系统分隔为多个安全域，限制安全域之间的访问. 如果将上述 TPM 访问规则直接应用于虚拟计算环境，那么将会出现虚拟域之间 TPM 资源访问不受限制的问题. 在虚拟计算、云计算等多安全域应用中，TPM 访问规则和安全域访问规则之间必须协同一致，否则 TPM 对象在多安全域的创建、使用、释放方面存在着一定的安全问题. 例如：高安全域 P_1 拥有 TPM 密钥 K，P_1 的完整级、机密级分别为 $I(P_1) = 2$，$C(P_1) = 2$，密钥为 $I(K) = 2$，$C(K) = 2$. 低安全域 P_2（$I(P_2) = 1$，$C(P_2) = 1$）的用户知道密钥授权数据，存在如下问题：

- 多安全域 TPM 对象使用问题：尽管按照系统安全规则禁止低等级域访问高等级域对象，但是根据普通 PC 平台上 TPM 的访问规则，域 P_2 显然能够使用域 P_1 的密钥 K 进行加密、签名、创建子密钥等操作.
- TPM 子对象创建问题：安全域 P_2 创建子密钥 K'，按照安全域的完整性、机密性等级，新建密钥 $I(K') = 1$，$C(K') = 1$，而 TPM 的对象保护原则则要求密钥 K' 的机密级不小于密钥 K，这存在严重的冲突.
- TPM 对象销毁问题：如果安全域 P_1 销毁 TPM 密钥 K，将致使安全域 P_2 无法使用新创建的密钥 K'.

导致这些问题的原因是 TPM 对象的对象创建、使用、销毁不进行安全域访问限制，只要满足 Locality 和授权数据认证，TPM 允许任何安全级的主体访问其内部对象，且允许动态创建密钥、会话等新的 TPM 对象. 本文将针对这一问题，在 Microsoft 的半虚拟化 TPM 体系结构基础上，建立 TPM 虚拟域安全模型.

2 TPM 虚拟域安全模型

2.1 TPM 模型基础

2.1.1 基本概念

TPM 对象的访问遵循 Denning[18] 的流模型（flow model，FM）定义，即 FM = $(O, S, SC, \oplus, \rightarrow)$，其中

O 是 TPM 内部实体对象的集合,这些对象都是 TPM 信息载体(包括密钥、PCR、计数器等);有限集 S 是 TPM 对象访问的主体,包括用户、进程等;有限集 SC 定义所有可能的安全类别,二元关系"→"指定了 2 个安全类之间的信息流动的关系,因此定义了 SC 上的一个偏序关系,该关系满足自反性(即信息可以在自身或者同级别对象中流动)和传递性;二元运算符"⊕"定义 SC 集合的最小上界操作(上确界),同时满足结合律和交换律,实质上 FM 暗含定义了 SC 上的二元运算关系"⊗",SC 集合上的下确界操作. 基于以上对信息流模型的定义,TPM 信息流安全可简单表述为:当且仅当一个操作序列的执行不会产生违反关系"→"所规定的信息流向,那么 TPM 信息流模型 FM 是安全的. 为了更准确地描述 TPM 安全模型,定义下列基本概念.

可信主体:拥有 TPM Owner 权限,该可信进程可以委托 TPM Owner 权限,灵活地配置不同域的 TPM 访问主体的安全级,可信主体记为 π. 只有可信主体 π 才能赋予 s 对 TPM 客体 o 执行操作 q 的权限 $(\lambda(q(s,o)))$. 可信主体可以经过扩展的 TSS 的 TCS 服务进程实现.

最小权限主体:TPM 中的公共对象允许任何进程进行访问,公共对象 po 的操作权限是主体操作 TPM 的最小权限,仅拥有 TPM 最小权限的主体称为最小权限主体,记为 θ.

主体间定义拥有关系,主体 s 创建了一个新主体 s',那么便称 s 拥有 s',或 s' 隶属于 s,表示为 $s = \tau(s')$, 或者 $s \ni s'$, s 为父进程主体,s' 为子进程主体.

安全级定义:主体和客体的机密性,完整性用机密级 C 和完整级 I 表示,主体和客体的 TPM 特权级(Locality)用 LOC 表示,C、I、LOC 为非负整数表示, $C \in [*,0,1,2,3]$,$I \in [*,0,1,2,3]$,$\mathrm{LOC} \in [*,0,1,2,3,4]$, 值越大表示安全级越高. * 表示最低的安全等级,LOC 中 * 是为了兼容 TPM 1.1 规范定义的遗留 Locality(Legacy Locality),表 2 描述了 Locality 等级分类.

标记(label):标记是一组用来表示 TPM 主体或客体的各种属性的参数,每个 TPM 客体的标记由扩展的 TSS Context Manager 按照一定的规则自动生成,主体的标记则是由操作系统自动生成,或是由用户自己定义. 对于 TPM 的客体可标记为 $(C(o),I(o),\mathrm{LOC}(o))$, TPM 的主体标记表示为 $(\tau(s),C(s),I(s),\mathrm{LOC}(s))$. C,I,LOC 三者共同构成 TPM 主客体的安全级 $SC = \{C,I,\mathrm{LOC}\}$.

表 2　Locality 等级

Locality 等级	使用范围
4	特定的初始化硬件
3	特定可信初始化软件
2	可信操作系统
1	保留
0	普通应用软件

2.1.2　虚拟域、辖域

TPM 内部对象之间存在一种保护关系,例如密钥树中子密钥受父密钥保护,TPM 会话受密钥或 Owner 对象保护,图 1 描述了在虚拟域环境中 TPM 对象间的相互关系,其中 CHIP 对象是 TPM 物理芯片,保护 TPM 内部的全部部件对象和数据对象,引入虚拟的 Φ 对象,设定它受 TPM 所有对象保护. 客体 o 直接保护客体 o' 记为 $\varphi(o') = o$. 根据客体间的保护关系,显然可定义顺序关系">", $o = \varphi(o') \Leftrightarrow o > o'$, 再进一步定义偏序关系"≥": $o \geqslant o' \Leftrightarrow (o = o' \vee (o > o') \vee (o > o_1 > \cdots > o_i > o'))$, 其中 $i \geqslant 1$. 客体间的保护关系满足:自反性: $\forall o, o \geqslant o$; 对称性: $\forall o_1, o_2, o_1 \geqslant o_2$, 且 $o_2 \geqslant o_1$, 则 $o_1 = o_2$; 传递性: $\forall o_1, o_2, o_3, o_1 \geqslant o_2, o_2 \geqslant o_3$, 则 $o_1 \geqslant o_3$.

这种偏序关系确立了偏序集 $G(o) = \{o' \mid o \geqslant o'\}$,$G(o)$ 表示了 o 所保护的客体所构成的虚拟域(对象 o 直接保护、间接保护的客体的总体). 图 1 不仅描述了在虚拟域环境下 TPM 对象间的依赖关系,还反映了对象间安全级的关系. 对象 o 保护 o',如果 o 的完整性遭受破坏,将导致 o 的虚拟域 $G(o)$ 中的对象(包括 o')无法使用,因而随着保护关系往下,TPM 对象完整性级安全要求越低;如果攻击者知道了 o 的授权数据,便能够访问 o,但是对于不可迁移对象而言攻击者仍无法访问 o',攻击者要访问 o' 还得知道 o' 的授权数据,因此受保护对象 o' 的机密性不低于父对象 o.

$$o = \varphi(o') \Rightarrow o \geqslant o', o \geqslant o' \Leftrightarrow o' \in G(o), o' \in G(o) \Rightarrow C(o') \geqslant C(o) \wedge I(o) \geqslant I(o'). \quad (2)$$

由主体和客体之间的拥有关系,可以定义集合 $D(s) = \{o \mid s \ni o\} \cup \{po\}$, 它是主体 s 拥有 TPM 客

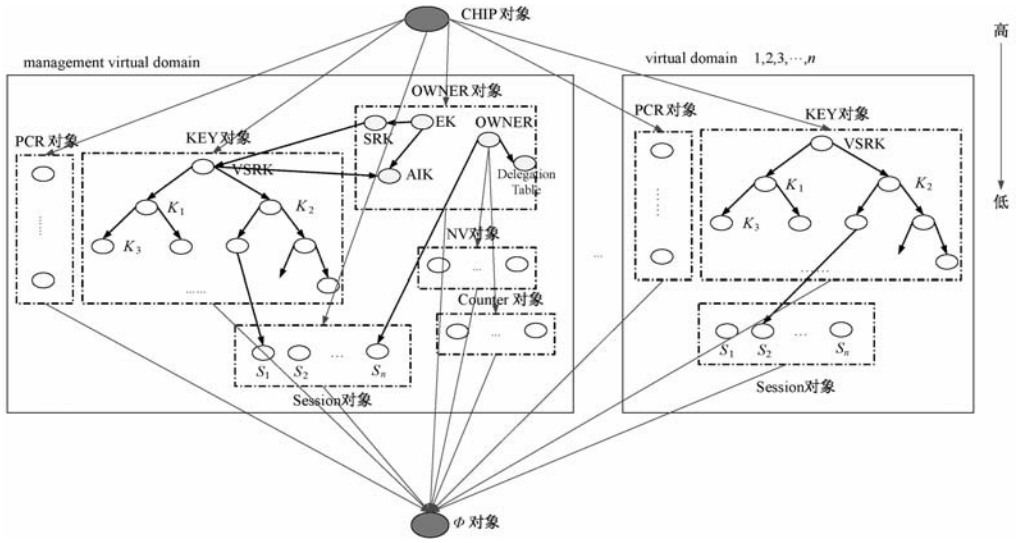

图 1 虚拟域中 TPM 对象依赖关系

体所组成的集合,也即是主体 s 能够访问 $D(s)$ 中的全部客体, 显然公共对象 $po \in D(s)$. 可以把 $D(s)$ 看成主体所能操作的客体范围, $D(s)$ 称之为主体 s 的辖域. 定义客体使用域 $U(o) = \{s \mid s \ni o\}$, 即能够操作对象 o 的全部主体的集合. 显然下列关系成立:

$$o \in D(s) \Leftrightarrow s \in U(o),\ \forall s(po \in D(s) \wedge s \in U(po)).$$

进程主体可以根据需要新创建 TPM 密钥、会话、计数器等对象,新建对象与父对象的安全级关系如何确定,我们引入了 TPM 对象安全级掩码 $SC_{\text{mask}} = \{C_{\text{mask}}, I_{\text{mask}}, \text{LOC}_{\text{mask}}\}$, 安全级掩码由可信主体 π 确定. 创建新对象 o 的安全级为

$$SC(o) = \{C(\varphi(o)) \oplus C_{\text{mask}}, I(\varphi(o)) \otimes I_{\text{mask}}, \text{LOC}(\varphi(o)) \oplus \text{LOC}_{\text{mask}}\}. \tag{3}$$

TPM 主体的辖域包含一组可访问对象的集合,TPM 主体 s 可访问对象的最大安全级称之为主体 s 辖域的安全级. $SC(D(s)) = \{\oplus\ C(D(s)), \otimes\ I(D(s)), \oplus\ \text{LOC}(D(s))\}$, 其中, $\oplus\ C(D(s))$、$\otimes\ I(D(s)$、$\oplus \text{LOC}(D(s))$ 分别表示主体辖域集合机密级的上确界、完整级的下确界、Locality 级的上确界.

2.2 安全规则

第 1 节描述的 TPM 安全机制以 TPM 物理芯片保障其完整性,以授权数据保证其机密性,这种简单的规则无法适应高安全应用系统需求. 因此在 TPM 虚拟域、辖域和使用域的基础上,定义 TPM 虚拟域安全规则.

规则 1 如果 s 创建了 TPM 客体 o,s 自然是 o 的拥有者,知道其授权数据.

$$c(s,o) \Rightarrow s \ni o, \text{且 } c(s,o) \Rightarrow o \in D(s), c(s,o) \Rightarrow s \in U(o).$$

规则 2 任何主体的 TPM 辖域都包括最小权限主体的辖域,可信主体 π 管理主体、客体的完整级和机密级,委托 TPM owner 权限,因此拥有 TPM 最大辖域,任何主体的辖域都是其子集.

$$\forall s(D(\theta) \subseteq D(s) \wedge D(s) \subseteq D(\pi)).$$

规则 3 子进程继承了其父进程的机密级、完整级和 Locality 级.

$$s \ni s' \Rightarrow (I(s') = I(s) \wedge C(s') = C(s) \wedge \text{LOC}(s') = \text{LOC}(s)).$$

规则 4 TPM 对象 o 属于 s 的辖域,其父对象 $\varphi(o)$ 也属于 s 的辖域. s 如果不能访问其父对象 $\varphi(o)$, 显然无法访问受 $\varphi(o)$ 保护的子对象.

$$o \in D(s) \Rightarrow \varphi(o) \in D(s).$$

为了限制 TPM 的不可信信息流,定义下列完整性、机密性安全规则.

规则 5 当且仅当 TPM 对象属于主体进程辖域,或者存在可信主体 π 委托读、执行 TPM 对象,同时进程

主体的密级大于等于客体密级,完整级小于等于客体完整级,Locality 级大于等于客体 Locality,主体可以对 TPM 客体进行读和执行操作.

$$(o \in D(s) \vee \lambda(r(s,o))) \wedge (C(s) \geqslant C(o) \wedge I(s) \leqslant I(o) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(o)) \Leftrightarrow r(s,o),$$

$$(o \in D(s) \vee \lambda(x(s,o))) \wedge (C(s) \geqslant C(o) \wedge I(s) \leqslant I(o) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(o)) \Leftrightarrow x(s,o).$$

规则 6　当且仅当 TPM 对象属于主体进程辖域,或者存在可信主体 π 委托允许 TPM 对象的写操作,同时进程主体的机密级等于客体机密级,完整级等于客体完整级,Locality 级大于等于客体 Locality,主体可以对 TPM 客体进行写操作.

$$(o \in D(s) \vee \lambda(w(s,o))) \wedge C(s) = C(o) \wedge I(s) = I(o) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(o) \Leftrightarrow w(s,o).$$

BLP,Biba 模型中,创建的新客体的安全级由主体所有者的安全级所确定,TPM 的对象虚拟域使 TPM 创建新对象的安全级规则截然不同,TPM 新建对象的安全级由父对象和安全级掩码确定. 不仅如此,创建新对象,还要用新对象安全级更新主体安全级,否则新对象创建主体达不到新客体访问安全级要求,无法访问该对象.

规则 7　TPM 的密钥、会话等对象支持动态创建,动态创建会话、密钥等子对象,要求主体满足父对象的安全级约束. 可信主体 π 无法对不存在的子对象作委托,所以该规则无委托指定.

$$\varphi(o) \in D(s) \wedge C(s) \geqslant C(\varphi(o)) \wedge I(s) \leqslant I(\varphi(o)) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(\varphi(o)) \Leftrightarrow c(s,o).$$

安全规则充分性分析:令 $\tilde{o} = \varphi(o)$,由于 $\tilde{o} \in D(s)$, s 拥有 $\tilde{o}$ 的访问权限,s 只要拥有 $\tilde{o}$ 的读、执行权限就允许创建子对象. 由规则 6 得到 s 和 $\tilde{o}$ 满足 $(C(s) \geqslant C(\tilde{o}) \wedge I(s) \leqslant I(\tilde{o}) \vee \lambda(r(s,\tilde{o})) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(\tilde{o})$, 因而

$$(\varphi(o) \in D(s) \wedge C(s) \geqslant C(\varphi(o)) \wedge I(s) \leqslant I(\varphi(o)) \vee \lambda(c(s,o)) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(\varphi(o)) \Rightarrow c(s,o).$$

必要性分析:如果 $c(s,o)$,由规则 1、4 得到 $I(o) = I(s) \wedge C(o) = C(s) \wedge \mathrm{LOC}(o) = \mathrm{LOC}(s)$,由虚拟域的安全级约束(式(2))得到 $\tilde{o} = \varphi(o) \Rightarrow C(\tilde{o}) \leqslant C(o) \wedge I(\tilde{o}) \geqslant I(o)$,所以 $c(s,o) \rightarrow \tilde{o} = \varphi(o) \Rightarrow C(\tilde{o}) \leqslant C(s) \wedge I(\tilde{o}) \geqslant I(s) \wedge \mathrm{LOC}(\tilde{o}) \geqslant \mathrm{LOC}(s)$, 因此

$$c(s,o) \Rightarrow (\varphi(o) \in D(s) \wedge C(s) \geqslant C(\varphi(o)) \wedge I(s) \leqslant I(\varphi(o)) \vee \lambda(c(s,o)) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(\varphi(o))$$

规则 8　主体 s 创建新对象后,可信主体 π 首先用安全级掩码确定新对象安全级,然后根据公式(3)用新对象安全级更新主体的安全级.

$$c(s,o) \wedge s \ni o \Rightarrow (C(o) = C(\varphi(o)) \oplus C_{\mathrm{mask}}) \wedge (I(o) = I(\varphi(o)) \otimes I_{\mathrm{mask}}) \wedge (\mathrm{LOC}(o) = \mathrm{LOC}(\varphi(o))),$$

$$c(s,o) \wedge s \ni o \Rightarrow (C(s) = C(s) \oplus C(o)) \wedge (I(s) = I(s) \otimes I(o)).$$

3　安全模型安全分析

3.1　安全模型相关引理

引理 1　$\forall o \exists s(s \ni o)$, 即任意一客体都属于某个主体.

证明　在 TPM 初始化之后(用户获取 TPM 所有权),TPM 所有静态对象属于可信主体 π,TPM 公用对象属于任何主体,其后由规则 1 保证了每创建 TPM 的一个新对象,都存在一个该客体的创建者,因而每个客体都有其拥有者.

引理 2　$o' \in G(o) \Leftrightarrow G(o') \subseteq G(o)$.

证明　$o' \in G(o)$, 由式(2)得到 $o \geqslant o'$, 设任意 $o'' \in G(o')$, 则 $o' \geqslant o''$, 由偏序关系的传递性得到 $o \geqslant o''$, 再根据式(2), $o'' \in G(o)$, 故 $G(o') \subseteq G(o)$.

若 $G(o') \subseteq G(o)$, 由虚拟域定义知 $o' \in G(o')$, 所以 $o' \in G(o)$.

引理 3　$D(s) \cap G(o) \neq \phi \Leftrightarrow o \in D(s)$

证明　令 $o' \in D(s) \cap G(o) \neq \phi$, 则 $o' \in D(s)$, $o' \in G(o)$, 由式(2)得到 $o \geqslant o'$, 若 $o' = o$, 得

证;若 $o' \neq o$, 令 $o_1 = o$, $o_i = o'$, 则存在 $o_1, \cdots, o_i$ ($i \geqslant 2$),满足 $o_1 > \cdots > o_i$, $o_i \in D(s)$, 因此 $o_{i-1} = \varphi(o_i)$, 根据规则 4, $\varphi(o_i) \in D(s)$, 递归的应用规则 4, $o_1 = \varphi^{(i-1)}(o_i) = \underbrace{\varphi(\varphi(\cdots}_{i-1}\varphi(o_i))) \in D(s)$, 所以 $o \in D(s)$. 反过来, $o \in D(s) \Rightarrow D(s) \cap G(o) \neq \phi$ 显然成立.

说明 该引理表明主体 s 的辖域与 TPM 对象 o 的虚拟域有交集,且不为空,则 TPM 对象 o 必定属于 s 的辖域. 由引理 3 的证明过程,可以得到如下推论.

引理 4 $o \geqslant o' \wedge o' \in D(s) \Rightarrow o \in D(s)$.

引理 5 $G(o') \subseteq G(o) \Rightarrow U(o') \subseteq U(o)$.

证明 令任意 $s \in U(o')$, 则 $o' \in D(s)$, 因为 $G(o') \subseteq G(o)$, 由引理 2 知, $o' \in G(o)$, 按照式(2),得到 $o \geqslant o'$, 根据引理 4,所以 $o \in D(s)$, 故 $s \in U(o)$, $U(o') \subseteq U(o)$. 该引理表明了虚拟域与使用域的子集关系,任意对象的虚拟域是对象 o 虚拟域的子集,能够操作该对象的用户集合也是对象 o 的用户集合的子集.

引理 6 $o \geqslant o' \Rightarrow U(o) \supseteq U(o')$.

证明 令任意 $s \in U(o')$, 则 $o' \in D(s)$, 因为 $o \geqslant o'$, 由引理 4 得到 $o \in D(s)$, 所以 $s \in U(o)$, $U(o') \subseteq U(o)$.

引理 6 和 7 表明了对象保护与使用域之间的关系,任意对象受到对象 o 保护,或者其虚拟域是对象 o 虚拟域的子集,能够操作该对象的用户集合也是对象 o 的用户集合的子集.

引理 7 $U(o) \cap U(o') = \phi \Rightarrow G(o) \cap G(o') = \phi$.

证明 假设 $G(o) \cap G(o') \neq \phi$, 用反证法证明其结论,令 $o'' \in G(o) \cap G(o')$, 由式(2)得到 $o \geqslant o''$ 和 $o' \geqslant o''$, 根据引理 6 知道, $U(o'') \subseteq U(o)$, $U(o'') \subseteq U(o')$, 所以 $U(o) \cap U(o') \neq \varphi$, 与条件矛盾,所以结论成立.

引理 8 $o \geqslant o' \Rightarrow \oplus C(G(o)) \geqslant \oplus C(G(o')) \wedge \otimes I(G(o)) \geqslant \otimes I(G(o'))$.

证明 $o \geqslant o'$, 由式(2)得到 $o' \in G(o)$, 根据引理 2, $G(o') \subseteq G(o)$, 设任意 $o'' \in G(o')$, 则 $o'' \in G(o)$, $\oplus C(G(o)) \geqslant C(o'')$, 所以 $\oplus C(G(o)) \geqslant \oplus C(G(o'))$, 同理可证 $\otimes I(G(o)) \geqslant \otimes I(G(o'))$.

3.2 安全模型分析

基于上述的安全规则和引理对 TPM 安全模型进行安全分析,可以得到以下 TPM 访问的安全性定理.

定理 1 TPM 机密性定理

$$\forall s \forall o((\lambda(r(s,o) \vee o \in D(s)) \wedge C(s) < \otimes C(G(o))) \Rightarrow \neg\, r(s,G(o)),$$
$$\forall s \forall o((\lambda(x(s,o) \vee o \in D(s)) \wedge C(s) < \otimes C(G(o))) \Rightarrow \neg\, x(s,G(o)),$$
$$\forall s \forall o((\varphi(o) \in D(s)) \wedge C(s) < C(\varphi(o))) \Rightarrow \neg\, c(s,o),$$
$$\forall s \forall o((\lambda(w(s,o) \vee o \in D(s)) \wedge C(s) \neq C(o)) \Rightarrow \neg\, w(s,o).$$

该定理说明 TPM 的主体在机密级上满足不能向上读、执行,不能向下写,不能向上创建新 TPM 对象,其中读和执行操作是适用于 TPM 对象的虚拟域.

证明 假设 $\lambda(r(s,o) \vee o \in D(s), C(s) < \otimes C(G(o)), r(s,G(o))$ 同时成立,设 $\forall o' \in G(o)$, 由规则 5 得到 $r(s,o') \Rightarrow (o' \in D(s) \vee \lambda(r(s,o'))) \wedge C(s) \geqslant C(o')$, 从而 $C(s) \geqslant C(o')$, 由于 $o' \in G(o)$, 由式(2)得到 $C(o') \geqslant C(o)$, 从而 $C(s) \geqslant C(o)$. $C(o) \geqslant \otimes C(G(o)) > C(s)$, 则 $C(s) < C(o)$, 矛盾,所以 TPM 主体不能向上读. 同理可以证明执行、创建、写操作的机密性定理.

定理 2 TPM 完整性定理

$$\forall s \forall o((\lambda(r(s,o) \vee o \in D(s)) \wedge I(s) > \oplus I(G(o))) \Rightarrow \neg\, r(s,G(o)),$$
$$\forall s \forall o((\lambda(x(s,o) \vee o \in D(s)) \wedge I(s) > \oplus I(G(o))) \Rightarrow \neg\, x(s,G(o)),$$
$$\forall s \forall o((\varphi(o) \in D(s)) \wedge I(s) > I(\varphi(o))) \Rightarrow \neg\, c(s,o),$$
$$\forall s \forall o((\lambda(w(s,o) \vee o \in D(s)) \wedge I(s) \neq I(o)) \Rightarrow \neg\, w(s,o).$$

TPM 的主体在完整级上满足不能向下读、执行,不能向上写,不能向下创建新对象.

证明　假设 $\lambda(r(s,o) \vee o \in D(s), I(s) > \oplus I(G(o)), r(s,G(o))$ 同时成立,设 $\forall o' \in G(o)$,由规则 5 得到 $r(s,o') \Rightarrow (o' \in D(s) \vee \lambda(r(s,o'))) \wedge I(s) \leqslant I(o')$,从而 $I(s) \leqslant I(o')$,由于 $o' \in G(o)$,由式(2)得到 $I(o) \geqslant I(o')$,从而 $I(o) \geqslant I(s)$. $I(s) \geqslant \oplus I(G(o)) > I(o)$,则 $I(s) < I(o)$,矛盾,所以 TPM 主体不能向下读. 同理可以证明执行、创建、写操作的完整性定理.

定理 3　TPM Locality 定理

$\forall s \forall o((\lambda(r(s,o) \vee o \in D(s)) \wedge \mathrm{LOC}(s) < \mathrm{LOC}(o)) \Rightarrow \neg q(s,o)$,其中,操作 q 为 r、c、w、x.

TPM Locality 用于对 TPM 特殊对象、特殊操作的限制访问,一般常用于远程证明的平台完整性度量,限制不同硬件、软件安全级的进程对 PCR 寄存器的操作,保证度量结果的真实、正确. TPM 的主体在 Locality 上满足不能向上执行任何操作.

证明　假设 $\lambda(r(s,o) \vee o \in D(s), \mathrm{LOC}(s) < \mathrm{LOC}(o), q(s,o)$ 同时成立,由规则 5,6,7 可以得到 $q(s,o) \Rightarrow (o \in D(s) \vee \lambda(q(s,o))) \wedge \mathrm{LOC}(s) \geqslant \mathrm{LOC}(o)$,从而 $\mathrm{LOC}(s) \geqslant \mathrm{LOC}(o)$,矛盾,因而 TPM 主体不能对高 Locality 的 TPM 对象执行任何操作.

定理 4　TPM 虚拟域定理

$$\forall s \forall o(U(o_1) \cap U(o_2) = \phi \wedge s \in U(o_1) \wedge o \in G(o_2)) \Rightarrow \neg q(s,o).$$

证明　假设 $U(o_1) \cap U(o_2) = \phi, s \in U(o_1), o \in G(o_2), q(s,o)$ 同时成立,$q(s,o)$ 根据规则 5,6,7 得到 $o \in D(s), o \in G(o_2)$ 根据式(2)得到 $o_2 \geqslant o$. $o \in D(s)$ 和 $o_2 \geqslant o$,由引理 4 得到 $o_2 \in D(s)$,即 $s \in U(o_2)$,且 $s \in U(o_1)$,从而 $s \in U(o_1) \cap U(o_2)$,与 $U(o_1) \cap U(o_2) = \phi$ 矛盾,所以 TPM 虚拟域定理成立.

TPM 虚拟域定理说明了虚拟域操作相互隔离的条件. 如果 TPM 的 2 个虚拟域父对象无公共操作主体(进程、用户),则这 2 个虚拟域是完全隔离的,即能够操作虚拟域 A 的主体,必然不能操作虚拟域 B.

上述 4 个定理从机密性、完整性、Locality、虚拟域 4 个方面确定了 TPM 主体对 TPM 对象的操作规则,可信操作系统中,对于 TPM 的访问除了要满足一般机密性、完整性强制限制外,还必须遵循 TPM 固有的 Locality、虚拟域限制. 本文所构建的 TPM 的虚拟域模型规范了 TPM 对象的安全级的定义,将 TPM 原有的授权数据、Locality、对象保护机制和可信操作系统强制访问控制相结合,形成一种新型的 TPM 安全模型.

TPM 安全模型能够灵活地应用于可信操作系统,便于配置管理 TPM 对象的权限. 可信操作系统任何进程主体安全级(C,I,LOC)初始化为(*,*,*),TPM 固有的静态对象按照虚拟域规则确立其安全级,一些常见对象的安全级初始化为 SRK(*,3,*),EK(1,3,*),Counter(1,3,*),PCR(*,3,?). 可信主体 π 不能改变这些 TPM 静态对象的安全级,但可以用安全级掩码控制 TPM 动态对象的安全级. 父对象 o 创建新子对象 o',可信主体管理员定义 TPM 创建新对象策略为 $C_{\mathrm{mask}} = C(o) + h$, $I_{\mathrm{mask}} = I(o) - h$,新对象的安全级为:$C(o) = C(\varphi(o)) \oplus C_{\mathrm{mask}}$, $I(o) = I(\varphi(o)) \otimes I_{\mathrm{mask}}$. 如果管理员定义 $h = 1$,那么从 SRK 根密钥开始计算,管理员可以严格地将创建的密钥层次树限制为 4 层,而密钥使用层次为 3 层(密钥的使用必须创建受密钥授权数据保护的会话对象). 如果管理员定义 $h = 0$,则不做密钥创建层次限制. 因此 TPM 静态对象的安全级由系统初始化时固定,动态对象的安全级由安全级掩码控制.

TPM 的主体安全级的管理受到多种因素的影响. 主体的安全级由父主体决定,但可信主体 π 可以对主体的安全级(Locality 除外)进行指派,该主体如果创建新 TPM 对象,则其安全级将随创建的新对象的安全级变化. 父主体对子主体的安全级影响按照规则 3 进行,新 TPM 客体对象对主体的安全级影响按照规则 8 进行. 可信主体除了不能改变主体 Locality 级以外,能控制主体的机密级和完整级,指派主体的操作权限等,但是可信主体除了设定安全级掩码管理动态创建的对象安全级外,对于 TPM 对象的安全级没有任何影响.

4　系统实现

我们将 TPM 虚拟域安全模型应用于可信虚拟平台环境,在虚拟机监控层中构建 TPM 访问控制模

块保证 TPM 对象在各个虚拟机内的无冲突的创建、使用、销毁等,实现 TPM 虚拟域在虚拟机环境下的管理. 可信虚拟平台上每个虚拟机都定义了一定的安全等级标识. 各个虚拟机对 TPM 对象的操作都通过位于虚拟机监控层的访问控制模块来完成,具体的系统体系结构如图 2 所示.

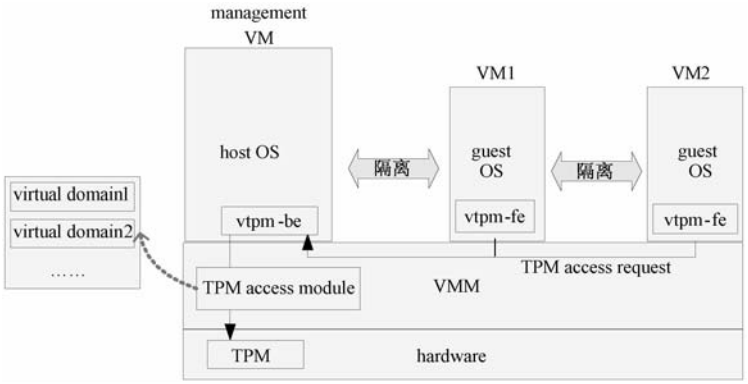

图 2 TPM 虚拟域体系结构

TPM 访问控制模块负责管理各个虚拟机的访问控制请求,为每个虚拟机维护一个 TPM 对象虚拟域环境,TPM 对象标记为隶属于各个虚拟机安全域,一些 TPM 基本对象如 PCR、EK 对象被各个虚拟域所共享. 虚拟机对每个 TPM 对象的访问除了必须遵循授权数据、Locality、Delegation 等基本规则外,还必须遵循 TPM 虚拟域规则. 除此之外还定义了更细粒度的虚拟域 TPM 命令访问控制策略,例如不允许普通虚拟域(VM1 等)执行 AIK 操作的相关命令,只允许 VM2 执行标记为 VM2 的密钥对象的加密和签名操作等规则. 为了评估 TPM 虚拟域访问控制对可信虚拟平台的性能影响,我们对系统的 TPM 命令访问请求性能进行了测试,表 3 为 TPM 普通正常执行的命令处理时间与含有 TPM 虚拟域规则控制的执行时间对比. 从测试结果可以看出,尽管增加了 TPM 虚拟域访问控制规则,但是这并不影响各个虚拟域执行 TPM 操作的效率,因此可见 TPM 虚拟域安全模型在实施中是可行的、有效的.

表 3 TPM 虚拟域测试结果对比 (s)

	TPM_Create WrapKey	TPM_LoadKey	TPM_UnBind	TPM_Sign	TPM_Quote
TPM 正常执行	0.5098	0.8604	0.6388	0.5952	0.5911
TPM 虚拟域执行	0.5861	0.8917	0.6442	0.6199	0.6315

5 总结

TPM 虚拟域安全模型从 TCG 的 TPM 访问控制机制出发,引入 TPM 虚拟域访问控制规则,建立 TPM 对象虚拟域模型,并进一步结合系统强制访问控制,增强 TPM 访问在虚拟计算环境的安全性. 原有 TPM 的 PC 平台安全模型中 TPM 对象和主体是独立的,这不适用于虚拟域计算环境. 本文依据 TPM 对象之间的依赖关系,虚拟域对 TPM 对象访问的约束关系,定义了在可信虚拟平台上 TPM 对象的访问规则,建立了 TPM 虚拟域安全模型,并且还通过实现原型系统证明 TPM 虚拟域安全模型的可行性和有效性. 因为虚拟域和安全级的引入,TPM 在虚拟计算环境的访问权限被更细粒度地划分,针对 TPM 对象的权限管理我们将进一步深入研究.

参考文献

[1] Trusted Computing Group. TPM main part 1, design principles specification, version 1.2 revision 62[EB/OL](2003-10)[2010-08-15] https://www.trustedcomputinggroup.org/home.

[2] Trusted Computing Group. TCG software stack (TSS) specification, version 1.10[EB/OL]. (2003)[2010-08-20] https://www.trustedcomputinggroup.org.

[3] Trusted Computing Group. TCG trusted network connect, TNC architecture for interoperability specification version 1.0 revision 4,3[EB/OL]. (2005-05)[2010-08-20] https://www.trustedcomputinggroup.org/home.

[4] Danilo B, Lorenzo C, Andrea L, et al. Replay attack in TCG specification and solution[C]//21st Annual Computer Security Applications Conference. 2005.

[5] Evan R S. A security assessment of trusted platform modules, TR2007-597[R]. Department of Computer Science, Dartmouth College, 2007.

[6] Safford D, Zohar M. A trusted linux client(TLC)[EB/OL]. [2010-08-20]http://www.research.ibm.com/gsal/tcpa/tlc.pdf.

[7] Biba K J. Integrity considerations for secure computer systems, ESD-TR-76-372[R]. USAF Electronic Systems Division, Hanscom Air Force Base, Bedford, Massachusetts, 1977.

[8] Bell D E, LaPadula L J. Secure computer systems: A refinement of the mathematical model[R]. Electronic Systems Division, Air Force Systems Command, Hanscom Aidr Force Base, Bedford, MA, 1974.

[9] Fraser T. LOMAC: low water-mark integrity protection for COTS environments[C]// the 2000 IEEE Symposium on Security and Privacy. USA: Oakland, California, 2000.

[10] Jaeger T, Sailer R, Shankar U. PRIMA: policy-reduced integrity measurement architecture[C]// ACM Symposium on Access Control Models and Technologies (SACMAT). California, 2006.

[11] Shankar U, Jaeger T, Sailer R. Toward automated information-flow integrity for security-critical applications[C]// the 13th Annual Network and Distributed Systems Security Symposium. 2006.

[12] Huang Q, Shen C X, Chen Y L, et al. Secrecy/integrity union MLS policy based on trusting computing[J]. Computer Engineering and Applications, 2006, 42(10):15-18 (in Chinese).
黄强,沈昌祥,陈幼雷,等.基于可信计算的保密和完整性统一安全策略[J].计算机工程与应用,2006,42(10):15-18.

[13] Sandhu R, Zhang X W. Access management for distributed systems: Peer-to-peer access control architecture using trusted computing technology[C]//the 10th ACM Symposium on Access Control Models and Technologies. 2005.

[14] Berger S, Cáceres R, Goldman K A, et al. vTPM: virtualizing the trusted platform module, RC23879[R]. IBM Research Division Thomas J. Tech Rep: Watson Research Center, 2006.

[15] Sadeghi A, Stüble C, Winandy M. Property-based TPM virtualization[C]// the 11th International Conference on information Security. Lecture Notes in Computer Science, 5222. Berlin, Heidelberg: Springer-Verlag, 1-16.

[16] England P, Loeser J. Para-virtualized TPM sharing[C]//the 1st International Conference on Trusted Computing and Trust in Information Technologies: Trusted Computing-Challenges and Applications (Villach, Austria, March 11 - 12, 2008). Lecture Notes in Computer Science. Berlin, Heidelberg: Springer-Verlag, 2009,4968:119-132.

[17] Trusted Computing Group. TCG PC client specific implementation specification for conventional BIOS, version 1.2 final, revision 1.00 [EB/OL]. July 2005, https://www.trustedcomputinggroup.org/home.

[18] Denning D E, A lattice model of secure information flow[J]. Communications of the ACM, 1976, 19(5): 236-243.

[19] Microsoft, Microsoft NGSCB home page[EB/OL]. (2003) [2010-08-15]http://www.microsoft.com/resources/ngscb.

[20] Garfinkel T, Pfaff B, Chow J, et al. Terra: A virtual machine-based platform for trusted computing[C]// the Symposium on Operating System Principles (SOSP). 2003.

[21] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization[C]// the 19th ACM Symposium on Operating Systems Principles. Bolton Landing, NY, 2003.

TPM security model for virtual domains

QIN Yu, LAN Hai-Bo

(*State Key Laboratory of Information Security*, *Institute of Software*, *Chinese Academy of Sciences*, *Beijing* 100080, *China*)

Abstract Considering that TPM access control mechanism can not be directly applied in virtualization computing, we build the security model for virtual domains based on the dependent relationships of TPM objects and the security requirements of the virtual domains. We add the security constraints of virtual domain, integrity and confidentiality, for the TPM objects' access requests in the model and solve the problems about TPM objects creation, usage, and destroy in multiple virtual domains. The logic analysis for the security rules in the model are further given in this paper. Through the tests on the prototype system, we show that the model has very small performance impact on trust virtualization platform.

Key words TCG, TPM security model, virtualization, virtual domain, security level