**Computer Science and Application** 计算机科学与应用, 2018, 8(5), 738-747
Published Online May 2018 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2018.85082

Hans汉斯

# Research on Trusted System Architecture and Virtual Certificate Chain in Cloud Environment

**Guan Wang[1,2], Yiqing Guo[1,2], Jianzhong Chen[1,2]**

[1]Department of Computer, Beijing University of Technology, Beijing
[2]Beijing Key Laboratory of Trusted Computing, Beijing
Email: wangguan@bjut.edu.cn, guoyiqing@emails.bjut.edu.cn, chenjianzhong@emails.bjut.edu.cn

Received: May 2nd, 2018; accepted: May 22nd, 2018; published: May 29th, 2018

## Abstract

**This paper proposed a trusted virtual machine system architecture based on independent Domain T, reduced the TCB size of existing systems, and increased vTPM computing performance with Xen's Credit scheduling algorithm unchanged. On this basis, by introducing the identity key of Domain T, it generated virtual trusted certificate under TCG main specifications, and provided a trusted root for vTPM. Finally, the test results show that the system reduces TCB size of Domain 0, improves vTPM computing performance, and provides client virtual machines with the capability of certificate generation and identity authentication.**

## Keywords

**Trusted Platform Module, Trusted Domain, TPM Credential, Virtual Certificate Chain**

# 云环境下可信系统架构与虚拟证书链生成研究

**王　冠[1,2]，郭一清[1,2]，陈建中[1,2]**

[1]北京工业大学计算机学院，北京
[2]可信计算北京市重点实验室，北京
Email: wangguan@bjut.edu.cn, guoyiqing@emails.bjut.edu.cn, chenjianzhong@emails.bjut.edu.cn

收稿日期：2018年5月2日；录用日期：2018年5月22日；发布日期：2018年5月29日

## 摘　要

**本文提出了基于独立可信虚拟域(Domain Trusted, Domain T)的可信虚拟机系统架构，降低了现有系统**

文章引用: 王冠, 郭一清, 陈建中. 云环境下可信系统架构与虚拟证书链生成研究[J]. 计算机科学与应用, 2018, 8(5): 738-747. DOI: 10.12677/csa.2018.85082

**中可信计算基(Trusted Computing Base, TCB)的大小，在保持Xen的Credit调度算法不变的情况下提高了vTPM (virtual TPM)计算性能。在此基础上，本文引入Domain T域身份密钥tEK，相比现有方法，在符合可信计算组织(Trusted Computing Group, TCG)主要规范的条件下生成了虚拟可信证书，为vTPM提供了信任根。最终测试结果表明，本系统在降低特权域TCB大小、提高vTPM计算性能的同时，提供了有效的客户虚拟机证书生成和身份认证的能力。**

## 关键词

**可信平台模块，可信虚拟域，TPM证书，虚拟证书链**



## 1. 引言

云计算是互联网时代一种备受关注的计算服务方式[1]，它通过虚拟化系统将计算资源虚拟化后供用户按需调用。然而，不同用户的虚拟资源可能被绑定到相同的物理资源上，不同的虚拟机可能会访问相同的物理设备[2]。此外，用户将计算任务委托给云服务商，也就失去了对计算环境和隐私数据的控制权。因此，云计算必须提供有效的安全措施，增强平台可靠性，降低安全风险。

可信计算作为保证系统安全的一项重要技术，通过可信平台模块(Trusted Platform Module, TPM)来保障计算机系统的完整性，极大地提高了操作系统的抗攻击能力[3]。目前，应用于云环境下的 TPM 的虚拟化方案主要包括 TPM 硬件环境扩展[4]，TPM 的半虚拟化[5]，以及软件式的 TPM 虚拟化。其中，TPM 硬件环境扩展方案通过扩展 TPM 上下文为客户虚拟机提供专用的 TPM 环境，但需要 TPM 硬件支持；TPM 半虚拟化方案通过一个软件 TPM 访问中间层来控制 TPM 调度及 TPM 对虚拟机的认证，但需要更改部分设备接口。相比而言，软件式 TPM 虚拟化方案以软件的形式为虚拟机提供了接近于物理 TPM 的接口，被大多数应用所采用[6]-[12]。

以 Xen 为代表的云平台虚拟机监控系统通过软件式方案实现了虚拟 TPM，为客户虚拟机提供了可信计算能力。TCB 的大小是一个可信平台最重要的安全因素之一，然而 Xen 将 vTPM 及其管理器设计在特权域 Domain 0 当中，一方面特权域会由于集成过多功能而导致 TCB 过大，一旦遭受攻击将使得客户虚拟机系统无法正常工作；另一方面，可信计算的计算需求也会和特权域中的其他请求一起抢占 vCPU 时间片，影响 vTPM 运算性能。

此外，以软件式方案实现的虚拟 TPM 没有受硬件保护的信任根(Core Root of Trust for Measurement, CRTM)，无法证明其自身的可信性。现有的虚拟可信证书生成方法大都通过硬件 TPM 的 AIK 密钥签发 vAIK 证书，以 TPM 担保 vTPM 的可信性。然而根据 TCG 规范，AIK 密钥不能对 TPM 外部数据进行加密。同时 AIK 密钥生命周期短，vAIK 证书随时有可能随 AIK 密钥的失效而突然失效。

因此，针对云环境下虚拟机监控系统存在的问题，本文对 Xen 的虚拟可信平台系统架构进行了改进，独立出一个可信计算专用虚拟域 Domain T。在此基础上，为进一步解决 vTPM 没有信任根无法生成 vAIK 证书链的问题，引入了 Domain T 的 tEK 域身份证书，使虚拟证书链的生成过程符合 TCG 相关规范要求。经过对上述方案的实现，本文可以构建起可信的虚拟云计算平台，并充分验证了系统的有效性。

## 2. 相关技术研究

### 2.1. 可信计算技术

可信平台模块 TPM 是可信计算平台的可信核心。根据 TCG 的主要规格概范，TPM 中包含 7 种类型的密钥[13]。其中，背书密钥(Endorsement Key, EK)是 TPM 的唯一标识，为避免平台身份暴露，仅被用于数据解密，不可进行其他直接操作。身份认证密钥(Attestation Identity Key, AIK)由 EK 生成，是 EK 的代替，负责对 TPM 内部数据与状态信息进行签名。AIK 是一种 RSA 非对称密钥，其私钥受 TPM 保护而不可迁移，公钥则作为证书通过可信第三方(Trusted Third Party, TTP)发布。

在 TPM 的虚拟化解决方案上，Strasser [14]提出的软件式 TPM 模拟器实现了物理 TPM 的绝大部分功能，但无法虚拟出多个 TPM 供每个虚拟机专用，只能面向单个平台环境。Perez [9]提出将一个物理 TPM 映射成多个 vTPM，为每个虚拟机提供一个单独的信任根，同时为了使 vTPM 具备远程证明能力还进一步设计了 vTPM 证书链的构建方法，但没有做进一步实现。Stumpf [10]提出的可信虚拟平台方案实现了 vTPM 和物理 TPM 的绑定，并利用 AIK 生成了 vAIK 证书链，但其构建过程不符合 TCG 对 AIK 的签名规范。

### 2.2. Xen 虚拟化技术

Xen 是一个基于开源 Linux 内核代码的虚拟化系统，它首先将 Hypervisor 载入到 Ring 0，然后授权一个特权域 Domain 0 协助 Hypervisor 参与对其它非特权域 Domain U 的管理。Xen 还采用了前后端分离的设备驱动结构，前端驱动位于 Domain U，接收操作系统发出的 I/O 请求，经由设备通道转发到后端；后端驱动位于 Domain 0，通过本地驱动访问真实设备，将前端请求交给物理硬件处理。

为实现虚拟域的可信启动，使不同虚拟域各自进行独立的可信计算，Xen 为每一个虚拟域提供了单独的 vTPM，由管理器 vTPM manager 统一分配管理。在 Xen 的 vTPM 架构中，vTPM 实例与 Domain U 一一对应，提供与物理 TPM 相同的功能；vTPM manager 是一个运行在 Domain 0 里的守护进程，通过执行 vTPM 的扩展命令实现 vTPM 实例的创建、迁移、删除；vTPM 设备驱动对为虚拟域应用程序使用 TPM 功能提供接口，为 vTPM manager 与 Xen 及其他虚拟域的数据传输提供支持。

Xen 中的 vCPU 调度基于优先级的信用值(credit)。Credit 调度算法是一种公平共享的非抢占式调度算法[15]，各个客户虚拟机按比例决定各自占用的 CPU 时间片，并控制单客户虚拟机的最大占比上限。因此，包括 Domain 0 在内各虚拟域的 vCPU 时间片都是有限的。vTPM manager 和 vTPM 隶属于 Domain 0，在域内计算需求互相竞争的情况下，可信计算相关的计算资源将无法得到公平保障。

## 3. 基于 Domain T 的虚拟机系统架构

### 3.1. 可信系统架构设计

本文从降低特权域 TCB 大小以及提升 vTPM 运行效率角度出发，将 vTPM manager 和 vTPM 实例迁移到独立的可信虚拟域 Domain T 中，提出基于 Domain T 的可信虚拟机系统架构如图 1 所示。

其中，可信虚拟域 Domain T 在结构上主要包含 vTPM manager 和 vTPM 实例两部分，是负责 vTPM 实例创建与管理的专用域。这一架构设计的优势性在于，一方面 Domain T 只支持 vTPM 相关功能，代码精简，有效地降低了 TCB 大小；另一方面，在 Credit 调度算法下能够提供更加公平的 vCPU 时间片保障，显著地提升了 vTPM 的计算性能。此外，Domain T 的出现也为下文 vTPM 信任根的构建提供了基础。

在可信虚拟域 Domain T 中，vTPM manager 模块负责监听客户机系统对 vTPM 的热插拔请求，持久化地保存 vTPM 运行状态，与硬件 TPM 通信，以及创建和管理 vTPM 实例。当客户虚拟机在创建配置中

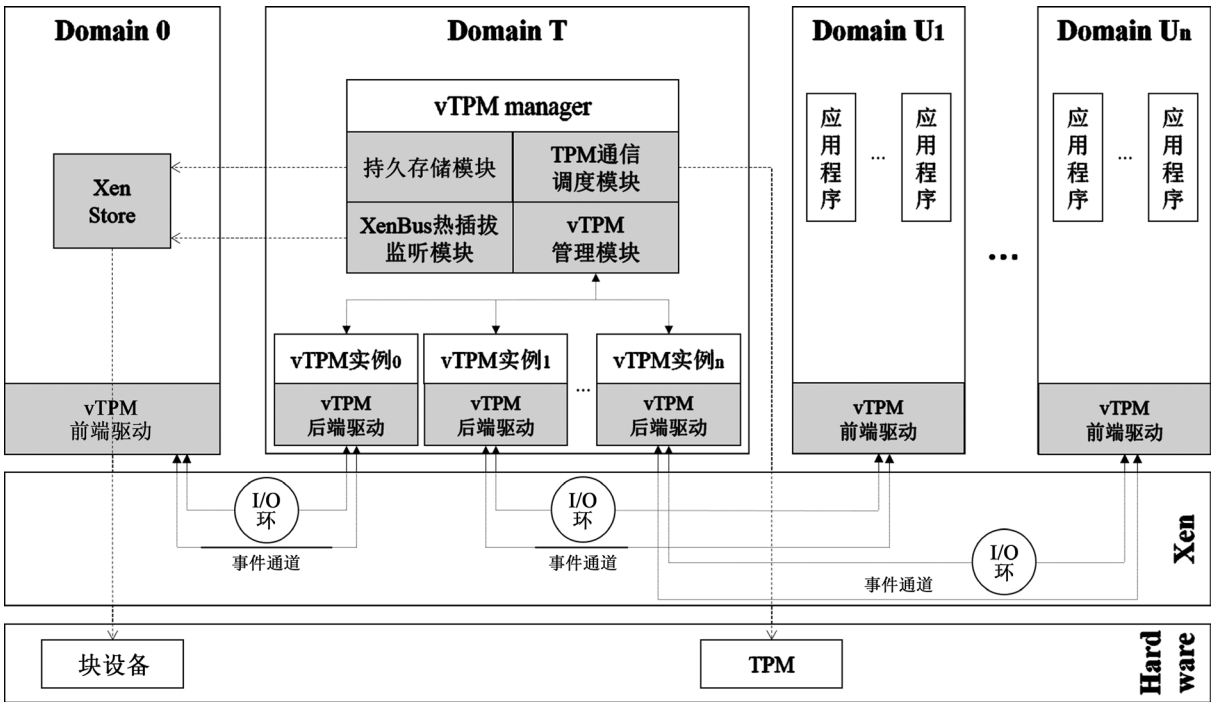


**Figure 1.** Xen trusted virtual machine system architecture based on Domain T
**图 1.** 基于 Domain T 的 Xen 可信虚拟机系统架构

声明 TPM 参数时，vTPM manager 会创建一个新的 vTPM 实例与客户虚拟机相对应。

vTPM 实例模块负责处理客户机发来的可信计算请求。其中，部分需要与硬件 TPM 通信的请求会转发给 vTPM manager 统一处理，其余请求则通过仿真逻辑在 vTPM 内部计算后直接返回给客户机系统。

### 3.2. 关键技术实现

Xen 的原有机制中，Domain U 的 vTPM 前端驱动发出的命令由 vTPM manager 负责转发给与之关联的 vTPM 实例。当请求过多时，vTPM manager 的处理能力就会成为限制 vTPM 和 Domain U 性能的瓶颈。本方案将这一流程优化，使 Domain U 中的 vTPM 前端驱动直接同 Domain T 中的 vTPM 后端驱动连接，省去了 vTPM manager 的转发环节。这一设计不仅保障了数据安全，也减轻了对 vTPM manager 的性能压力。

vTPM 前后端驱动建立连接的基础是 xend 向 XenStore 分别写 frontend 和 backend 信息。XenStore 中保存的 frontend 和 backend 信息如图 2 所示，vTPM manager 监听着/local/domain/0/backend/vbd 中的 frontend 键，vTPM 前端驱动则监听着/local/domain/U/device/vbd 中的 backend 键。当设备内容被写入到这两个位置后，两个监听的相关事件就会被触发。

由于每一个 vTPM 实例在监听 vTPM 前端驱动及与硬件 TPM 通信时都处于阻塞状态，为防止一个 vTPM 实例阻塞导致整个 vTPM manager 受到影响，本方案将 vTPM 实例独立地放在一个线程中运行，并将 vTPM 后端驱动放入 vTPM 实例线程中。在监听到 XenBus 热插入事件时，vTPM 管理模块创建 vTPM 实例与前端驱动一一对应。vTPM 线程创建完成后，初始化 vTPM 后端驱动模块并与 Domain U 中的 vTPM 前端驱动建立连接。vTPM 后端驱动模块初始化流程伪代码如图 3 所示。

如图中伪代码所示，vTPM 实例线程创建后，首先将 XenStore 中对应 Domain U 的后端驱动状态标记为 XenbusStateInitWait，等待获取更多信息以连接前端。然后监听 XenStore 中 backend/vtpm/domid/

```
/local/domain/0/backend/vbd/U/<deviceID>/...
        frontend            /local/domain/U/device/vbd/0/<deviceID>
frontend-id       U
        state               XenbusStateInitialising
        ...                 <device-specific details>
/local/domain/U/device/vbd/0/<deviceID>/...
        backend             /local/domain/0/backend/vbd/U/<deviceID>
        backend-id          0
        state               XenbusStateInitialising
        ...                 <device-specific details>
```

**Figure 2.** "KEY-VALUE" structure preserved in XenStore
**图 2.** XenStore 中保存的"键-值"结构

```
Procedure of init_vtpmback ()
1.  xenbus_switch_state ("backend/vtpm/domid/handle/state", XenbusStateInitWait);
2.  char* vtpmfstate = "backend/vtpm/domid/handle/frontend/state";
3.  While (state != XenbusStateConnected )do
4.          xenbus_wait_for_state_change(vtpmfstate, &state, &events);
5.  xenbus_read (XBT_NIL, "backend/vtpm/domid/handle/frontend/ ring-ref", &
    tpmif->fe_path);
6.  xenbus_read (XBT_NIL, "backend/vtpm/domid/handle/frontend/ event-channel", &
    evtchn);
7.  tpmif->page = gntmap_map_grant_refs(&gtpmdev.map, 1, &domid, 0, &ringref);
8.  BACK_RING_INIT(ring, s, PAGE_SIZE);
9.  evtchn_bind_interdomain(tpmif->domid, evtchn, tpmback_handler, tpmif,
    &tpmif->evtchn);
10. xenbus_switch_state ("backend/vtpm/domid/handle/state", XenbusStateConnected);
```

**Figure 3.** Pseudocode of vTPM backend driver initializing process
**图 3.** vTPM 后端驱动初始化流程的伪代码

handle/frontend/state 键值的变化，当前端驱动初始化完成时，state 字段会变为 XenbusStateConnected。接着，初始化后端驱动的配置信息，包括共享内存地址，授权引用信息等。最后将 XenStore 中后端驱动的状态改为 XenbusStateConnected，标志着 vTPM 前后端驱动连接成功。

## 4. 基于 tEK 证书的虚拟可信证书生成方法

### 4.1. vTPM 虚拟证书链扩展

由于 vTPM 中没有受硬件保护的 CRTM，因此虚拟环境下也就不会有 EK 证书。当客户虚拟机进行

远程证明时，vTPM 所产生的 vAIK 无法证明自己与一个可信的 vEK 绑定，也无法证明自身的可信性。因此，必须建立从物理 TPM 到 vTPM 的证书链，将 TPM 的信任关系传递到 vTPM 上。

本文针对上文提出的可信系统架构引入 Domain T 域身份密钥 tEK，建立起 tEK-vEK-vAIK 形式的证书链。在介绍证书链扩展方案前，首先对信任关系和可信证明的性质定义进行介绍：

**定义 1 信任关系**：实体 $A$ 信任 $B$，记做 $A\ Trust\ B$。信任关系具有以下性质：

1) 自信任性：$A\ Trust\ A$。

2) 传递性：$A\ Trust\ B, B\ Trust\ C \Rightarrow A\ Trust\ C$。

**定义 2 可信证明**：当实体 $B$ 向挑战者提供的证明实体 $A$ 信任 $B$ 的证据 $Evi_B$，能够满足 $A$ 信任 $B$ 的标准要求 $TrustS_{A,b}$ 时，可得 $A\ Trust\ B$，记做 $Evi_B\ Equ\ TrustS_{A,b} \Rightarrow A\ Trust\ B$。

对于证书链的构建方法，目前已有一些研究成果。文献[10]提出 $AIK-vAIK$ 形式的证书链方案，绕过 $vEK$ 直接由 $Cert_{AIK}$ 签发 $Cert_{vAIK}$。该方案首先利用 $AIK$ 对 $vAIK$ 公钥、物理 TPM 的 PCR 值、随机数 $nonce$ 及时间戳进行签名，在此本文用 $Sign^{AIK_{pri}}_{\{vAIK_{pub},PCR,nonce,time\}}$ 表示，然后将签名数据与 $Cert_{AIK}$ 一同构成 $Cert_{vAIK}$。由于 $Cert_{vAIK}$ 是由 $Cert_{AIK}$ 签发，因此实现了 $CA\ TrustCert_{AIK}, Cert_{AIK} TrustCert_{vAIK} \Rightarrow CA\ TrustCert_{vAIK}$ 的信任传递。然而根据 TCG 规范，$AIK_{pri}$ 不能用于对 TPM 外部数据签名。此外由于 $Cert_{AIK}$ 生命周期短，每次验证后都应当重新生成 $Cert_{AIK}$，导致 $Cert_{vAIK}$ 随 $Cert_{AIK}$ 的失效而失效。

针对 $AIK$ 不能对外部数据签名的规定，文献[16]提出一种新密钥 $SK\left(Signature\ key\right)$，通过 $Sign^{AIK_{pub}}_{SK_{pub}}$，$Sign^{SK_{pub}}_{vAIK_{pub}}$ 的方式形成 $AIK$ 与 $vAIK$ 的间接签名。然而，该方案依然存在 $Cert_{AIK}$ 时效性差的问题。

为解决 $Cert_{AIK}$ 时效性问题，文献[17]引入私钥生成中心(Private Key Generation, PKG)，提出利用环签名证明 VM 的可信性。首先云平台利用隐私认证中心(Privacy Certification Authority, CA)向 PKG 证明可信性，PKG 产生部分环签名密钥发给云平台，云平台再将部分环签名发给 vTPM manager，最后 vTPM manager 将签名分发给 vTPM。然而该方案的问题在于 vTPM 中的 vPCR 不完全等同于 TPM 中的 PCR，当挑战者发起验证时会因为 PKG 中存储的是云平台 PCR 值，与 vPCR 值不匹配而导致验证失败。

因此，为了规避 $Cert_{AIK}$ 时生命周期短的特性，同时满足 $AIK_{pri}$ 对 TPM 内部数据签名的合规性要求，本文引入 Domain T 域身份密钥 $tEK$。CA 首先通过验证 Domain T 的完整性向 Domain T 签发 $Cert_{tEK}$，之后由 Domain T 签发 $Cert_{vEK}$，进而保证 $Cert_{vAIK}$ 的合法性。

CA 签发 $Cert_{tEK}$ 必须建立在 $CA\ Trust\ DomT$ 的基础上。根据定义 1 的信任传递性特点，需要先证明 $CA\ Trust\ TPM, TPM\ Trust\ DomT$。

$Evi_{TPM}\ Equ\ TrustS_{CA,TPM} \Rightarrow CA\ Trust\ TPM$ 的过程是标准的 $Cert_{EK}$ 签发过程，在此不再赘述。接下来证明 $TPM\ Trust\ DomT$ 成立的条件是 $TrustS_{TPM,DomT} = \left\{DomT \in Xen, PCR[13] = SHA1\left(DomT\right)\right\}$。由于 Domain T 运行在 Xen Hypervisor 之上，本文将 Domain T 的度量包含在 Xen 的启动过程内，并将其摘要值扩展地写入 PCR[13]中，因此 $Evi_{DomT}\ EquT\ rustS_{TPM,DomT}$。

至此，$CA\ Trust\ TPM, TPMT\ rust\ DomT$ 的信任传递建立成立。

vTPM 中 $Cert_{vEK}, Cert_{vAIK}$ 生成的具体流程如下：

1) Domain T 生成 tEK 密钥对，并向 CA 发送 $Cert_{tEK}$ 请求。

2) CA 向 Domain T 发送 $CA_{pub}$ 以及一个随机数 $nonce$。

3) Domain T 利用该随机数 $nonce$ 向平台 TPM 发起远程挑战。

4) TPM 使用 $AIK_{pri}$ 对物理 PCR 值、$nonce$ 进行签名，将签名结果 $Sign^{AIK_{pri}}_{\{PCR,nonce\}}$ 和 $Cert_{AIK}$、度量日志一起返回给 Domain T。

5) Domain T 使用 $CA_{pub}$ 对 $tEK_{pub}$ 以及 TPM 发来的签名结果、$Cert_{AIK}$、度量日志进行签名，一起发给 CA。

6) CA 通过解密并比对 TPM 中 PCR [13]的值判断 Domain T 是否完整。若比对通过，则 $CA$ 向 Domain T 签发 $Cert_{tEK}$。

7) 当 Domain T 产生新的 vTPM 时，vTPM 随机生成 vEK 密钥对，Domain T 通过 $tEK_{pub}$ 签名 $vEK_{pub}$，配合 $Cert_{tEK}$ 一起形成 $Cert_{vEK}$。

8) 当 vTPM 申请 $Cert_{vAIK}$ 时首先向 Domain T 发送 $Cert_{vAIK}$ 请求，由 Domain T 返回一个 $tEK_{pub}$ 以及一个随机数 $nonce$。

9) vTPM 随机生成 vAIK 密钥对，然后将 $vAIK_{pub}$、$Cert_{vEK}$ 和 $nonce$ 经 $tEK_{pub}$ 签名后发送给 Domain T。

10) Domain T 通过解密比对确认 $Cert_{vEK}$ 有效后签发 $Cert_{vAIK}$。

CA 通过验证平台完整性即可验证 Domain T 的完整性，通过挑战平台完整性即可挑战 $Cert_{tEK}$ 的有效性。这一过程不涉及 AIK 密钥对 TPM 外部数据的签名。$Cert_{vEK}$ 与 $Cert_{vAIK}$ 由 Domain T 直接颁发，解决了 $Cert_{vAIK}$ 与某一个 $Cert_{AIK}$ 强绑定所导致的时效性差的问题。围绕 $Cert_{vAIK}$ 的生成与签发，本文所提虚拟证书链构建方案与其他现有方法的优缺点对比如表 1 所示。

### 4.2. vTPM 虚拟证书链实现

本文在 Domain T 的 tEK 证书申请过程中，采用 Intel 的 OpenAttestation 开源框架作为 CA，通过在 Domain T 中新增 GetCert 命令人工地触发 tEK 证书申请流程。当 Xen 获得 nonce 随机数后，利用 Tspi_Context_CreateObject 函数构建一个 PCR 对象，利用 Tspi_PcrComposite_SelectPcrIndex 函数获取 PCR [13]的值，并利用 Tspi_TPM_Quote (AIK, PCRIndex, nonce)函数生成平台状态信息，最后将平台信息发送给 Domain T。同时，OpenAttestation 通过修改源码实现了 PCR 值比对和 tEK 证书签发的功能。

此外，本文对 vTPM 实例中 TPM Emulator 的工作流程进行了修改，在 TPM Emulator 创建完成 vEK 公钥后，改为向 Domain T 中的 CertListener 进程发送 vEK 证书申请，vAIK 证书申请也被重定向到 CertListener 中。CertListener 向 vTPM 返回 vAIK 证书后，vTPM 利用 Tspi_Key_LoadKey 函数加载 vAIK 证书进入 vTPM，并利用 Tspi_TPM_ActivateIdentity 函数激活 vAIK 证书。

## 5. 基于 Domain T 的虚拟机系统测试

为验证 vTPM 虚拟证书链扩展的可用性，本文从虚拟平台的远程证明以及 vTPM 的计算性能两个维

**Table 1.** Comparison of approaches to generate vAIK certificate
**表 1.** vAIK 证书生成方法比较

| 需求对比 | AIK 签名 vAIK 证书法 | SK 签名 vAIK 证书法 | 环签名法 | tEK 签名 vEK 证书法 |
|---|---|---|---|---|
| 是否需要 TTP 参与 | 不需要 | 不需要 | 需要 CA 和 PKG 参与 | 需要 CA 和 Domain T 参与 |
| CA 如何验证 vAIK 证书 | 需要知道 AIK 公钥，并通过 AIK 公钥解密加密值后比对内容合法性 | 需要知道 SK 公钥，并通过 AIK 证书验证 SK 合法性后再验证内容合法性 | 通过环签名验证等式进行验证 | 向 Domain T 验证 AIK 证书合法性 |
| 能否满足 VM 远程证明需求 | 可以 | 可以 | 不可以，vPCR 值与 PCR 值不完全相同 | 可以 |
| 是否依赖 AIK 证书 | 依赖 | 依赖 | 不依赖 | 不依赖 |
| vAIK 证书请求是否依赖 TPM 支持 | 依赖 | 依赖 | 不依赖 | 不依赖 |

度出发，进行了系统有效性测试。本章实验采用 Ubuntu 12.4 作为宿主机的操作系统，宿主机中安装了 4.9 版本的 Xen 系统，其中 Domain 0 内核是 Linux 3.7.1 版本，Domain U 内核是 3.7.9 版本。此外，宿主机主板上搭载了一块 Infineon v1.2TPM 芯片，TSS 软件栈采用 Trousers。

通过将 vTPM manager 和 vTPM 实例迁出特权域，特权域源码大小降低了 1.73 MB，占 Xen 源代码大小的 10.9%。经过迁移后的 Domain T 架构能够满足可信虚拟机的正常运转要求。

## 5.1. 客户机系统远程证明测试

本节测试虚拟平台远程证明能力的有效性，主要包括 tEK 证书和 vAIK 证书的生成能力。

首先对 tEK 证书的生成时间进行测试。生成 tEK 证书的时间由两部分组成：CA 验证 AIK 证书及 Domain T 完整性时间，以及验证完成后签发 tEK 证书时间。如表 2 所示，生成 tEK 证书平均用时约 17.24 秒。由于证书签发流程较长，所以 tEK 证书生成时间也比较长，但考虑到该证书只在每次平台启动时申请一次，因此还是具备一定的实用性。

其次对 vAIK 证书的生成时间进行测试。如 4.1 节所述，vAIK 签发流程如下：

1) vTPM 生成 vAIK 密钥对，并向 Domain T 发送 vAIK 证书请求。

2) Domain T 返回 tEK 公钥以及随机数 nonce。

3) vTPM 向 Domain T 发送 tEK 公钥签名后的 vAIK 公钥、vEK 证书和 nonce 随机数。

4) Domain T 使用 tEK 私钥解密，在检查 nonce 新鲜度和 vEK 证书有效性后签发 vAIK 证书。

如表 3 所示，生成 vAIK 证书平均用时约 5.47 s。这一时间相比 tEK 证书生成时间更短，这是因为该过程涉及证书都在本地签发，省去了和硬件 TPM 的通信过程。

## 5.2. vTPM 性能测试

以上实验证明，本文所提 Domain T 架构以及 tEK 证书与 vAIK 证书的生成方案具备一定实用性。本节测试独立后的 Domain T 是否会对 vTPM 的计算性能产生影响。

在此本节对 AIK-vAIK 的证书生成方法进行了实现，在同等实验环境下测试在部署了 Domain T 和未

**Table 2.** tEK certificate generating time
**表 2.** tEK 证书生成时间

| 实验组 | 验证 AIK 证书及 Domain T 完整性 | 签发 tEK 证书 | 合计 |
|---|---|---|---|
| 1 | 9.13 | 8.36 | 17.49 |
| 2 | 9.07 | 8.35 | 17.42 |
| 3 | 8.59 | 8.10 | 16.69 |
| 4 | 9.03 | 8.31 | 17.34 |

**Table 3.** vAIK certificate generating time
**表 3.** vAIK 证书生成时间

| 实验组 | 生成 AIK 密钥并发送请求 | 获取 nonce | 返回 tEK 签名数据 | 签发 tEK 证书 | 合计 |
|---|---|---|---|---|---|
| 1 | 0.87 | 0.15 | 3.54 | 1.02 | 5.58 |
| 2 | 0.83 | 0.11 | 3.28 | 1.15 | 5.37 |
| 3 | 0.87 | 0.13 | 3.38 | 1.07 | 5.45 |
| 4 | 0.85 | 0.13 | 3.40 | 1.11 | 5.49 |

**Table 4.** vAIK certificate comparative generating time
**表 4.** vAIK 证书对比生成时间

| 实验组 | 基于 Domain T 架构的 Xen 系统 | 原始 Xen 系统 |
|---|---|---|
| 1 | 3.17 | 3.56 |
| 2 | 3.34 | 3.62 |
| 3 | 3.19 | 3.51 |
| 4 | 3.05 | 3.42 |

部署 Domain T 的 Xen 环境中，客户虚拟机生成 vAIK 证书所用的时长。

如表 4 所示，对比实验表明，在基于 Domain T 架构的 Xen 系统中，生成 vAIK 证书平均用时约 3.19 s；未采用 Domain T 架构的 Xen 系统中，生成 vAIK 证书平均用时约 3.53 s。相比改进前，独立后的 Domain T 架构在只安装一个客户虚拟机 Domain U 的情况下，vAIK 证书的请求速度可以提高约 9.63%。

## 6. 结束语

本文以 Xen 系统为基础，提出了基于 Domain T 的可信虚拟机系统架构。通过迁出 vTPM manager 和 vTPM 实例降低了 Domain 0 的 TCB 大小，理论上可以保障 Domain T 的计算资源不被其他虚拟机的计算请求抢占，提高了 vTPM 性能。同时，本文的虚拟证书链生成方案通过引入 Domain T 域身份密钥 tEK，解决了现有方法中用 AIK 签名 TPM 外部数据为 vAIK 证书背书的问题。最后，本文实现了相关原型系统，并验证了系统的有效性。

## 参考文献


[1] 冯登国, 张敏, 张妍, 等. 云计算安全研究[J]. 软件学报, 2011, 22(1).

[2] 罗东俊. 基于可信计算的云计算安全若干关键问题研究[D]. 广州: 华南理工大学, 2014.

[3] 沈昌祥, 张焕国, 王怀民, 等. 可信计算的研究与发展[J]. 中国科学: 信息科学, 2010, 40(2): 139-166.

[4] Goldman, K.A. and Berger, S. TPM Main Part 3 IBM Commands. http://www.research.ibm.com/secure_systems_department/projects/vtpm/mainP3IBMCommandsrev10.pdf

[5] England, P. and Loeser, J. (2008) Para-Virtualized TPM Sharing. International Conference on Trusted Computing. Springer, Berlin, Heidelberg, 119-132.

[6] Anderson, M.J., Moffie, M. and Dalton, C.I. (2007) Towards Trustworthy Virtualisation Environments: Xen Library os Security Service Infrastructure. *HP Technical Report*, 88-111.

[7] Stumpf, F., Eckert, C. and Balfe, S. (2008) Towards Secure e-Commerce Based on Virtualization and Attestation Techniques. *Third International Conference on Availability, Reliability and Security*, 2008. *ARES* 08. IEEE, 376-382.

[8] Scarlata, V., Rozas, C., Wiseman, M., *et al*. (2008) TPM Virtualization: Building a General Framework. Trusted Computing. Vieweg + Teubner, 43-56.

[9] Perez, R., Sailer, R. and van Doorn, L. (2006) vTPM: Virtualizing the Trusted Platform Module. *Proceedings of the* 15*th Conference on USENIX Security Symposium*, 305-320.

[10] Stumpf, F., Benz, M., Hermanowski, M., *et al*. (2007) An Approach to a Trustworthy System Architecture Using Virtualization. *International Conference on Autonomic and Trusted Computing*, Springer, Berlin, Heidelberg, 191-202.

[11] Jansen, B., Ramasamy, H. and Schunter, M. (2006) Flexible Integrity Protection and Verification Architecture for Virtual Machine Monitors. *Second Workshop on Advances in Trusted Computing*.

[12] Sadeghi, A.R., Stüble, C. and Winandy, M. (2008) Property-Based TPM Virtualization. *International Conference on Information Security*, Springer, Berlin, Heidelberg, 1-16.

[13] Strasser, M. and Stamer, H. (2008) A Software-Based Trusted Platform Module Emulator. *International Conference on Trusted Computing*, Springer, Berlin, Heidelberg, 33-47.

[14] TPM Main Specification Level 2 Version 1.2, Part 2, TPM Structures. 2011-1.

https://trustedcomputinggroup.org/wp-content/uploads/TPM-Main-Part-2-TPM-Structures_v1.2_rev116_01032011.pdf
[15] 顾振宇, 张申生, 李晓勇. Xen 中 Credit 调度算法的优化[J]. 微型电脑应用, 2009, 25(2): 1-3.
[16] 王丽娜, 高汉军, 余荣威, 等. 基于信任扩展的可信虚拟执行环境构建方法研究[J]. 通信学报, 2011.
[17] 荣星, 赵勇. 基于无证书环签名的虚拟机可信证明方案[J]. 计算机应用, 2017, 37(2): 378-382.