

基于 VMI 的虚拟机远程证明方案

王伟^{1,2}, 陈兴蜀^{2,3}, 兰晓², 金鑫^{1,2}

(1. 四川大学计算机学院, 四川 成都 610065;
2. 四川大学网络空间安全研究院, 四川 成都 610065;
3. 四川大学网络空间安全学院, 四川 成都 610065)

摘要: 可信计算组织 (TCG, trusted computing group) 提出的虚拟机远程证明方案可以为云计算平台提供虚拟机完整性验证服务, 而直接使用 TCG 提出的方案性能较低, 并且会受到布谷鸟攻击的威胁。利用虚拟机自省技术 (VMI, virtual machine introspection) 设计了新的虚拟机远程证明方案。通过在虚拟机监视器 (VMM, virtual machine monitor) 中获取虚拟机远程验证证据的方法消除在虚拟机内执行布谷鸟攻击的路径, 利用物理可信平台模块 (TPM, trusted platform module) 保证虚拟机远程验证证据的完整性, 减少了身份证明密钥 (AIK, attestation identity key) 证书的产生数量, 降低了私有证书颁发机构的负载。实验表明, 方案可以有效验证虚拟机的完整性状态, 在虚拟机数量较多的情况下, 性能优于 TCG 提出的虚拟机远程证明方案。

关键词: 虚拟机远程证明; 布谷鸟攻击; 虚拟机自省技术; 可信平台模块; 身份证明密钥

中图分类号: TP309

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2018098

VMI-based virtual machine remote attestation scheme

WANG Wei^{1,2}, CHEN Xingshu^{2,3}, LAN Xiao², JIN Xin^{1,2}

1. College of Computer Science, Sichuan University, Chengdu 610065, China
2. Cybersecurity Research Institute, Sichuan University, Chengdu 610065, China
3. College of Cybersecurity, Sichuan University, Chengdu 610065, China

Abstract: The virtual machine attestation scheme proposed by trusted computing group (TCG) can provide attestation service of virtual machine for cloud computing. However, the service using the scheme proposed by the TCG directly would be threatened by the cuckoo attack and its performance would be lower. Therefore, a new virtual machine remote attestation scheme based on virtual machine introspection (VMI) was proposed. Firstly, it eliminated the path to perform cuckoo attacks in virtual machines via obtaining virtual machines' remote attestation evidence in virtual machine monitor (VMM). Secondly, it used physical trusted platform module (TPM) to ensure the integrity of virtual machines' remote attestation evidence and reduced the number of attestation identity key (AIK) certificates required during remote attestation to balance the load of private CA. Experiments show that the proposed scheme can verify the status of virtual machines correctly and increase the performance of bulk virtual machines' remote attestation significantly.

Key words: virtual machine remote attestation, the cuckoo attack, virtual machine introspection, TPM, attestation identity key

收稿日期: 2018-11-02; 修回日期: 2018-11-29

通信作者: 陈兴蜀, chenxsh@scu.edu.cn

基金项目: 国家自然科学基金资助项目 (No.61802270); 国家“双创”示范基地之变革性技术国际研发转化平台基金资助项目 (No.C700011)

Foundation Items: The National Natural Science Foundation of China (No.61802270), The Transformational Technology International Research Platform for National Dual Innovation Base (No.C700011)

1 引言

随着云计算的广泛应用, 虚拟机的完整性状态受到越来越多的关注。可信计算技术^[1]可以为云计算提供可靠的运行环境完整性解决方案。然而, 传统的可信计算技术无法验证虚拟机状态的完整性, 于是 Berger 等^[2]提出了虚拟可信计算技术。

可信计算组织 (TCG, trusted computing group) 提出了基于可信平台模块 (TPM, trusted platform module) 的远程证明机制, 并制定了虚拟可信平台规范, 其中包括基于虚拟可信平台模块 (vTPM, virtual trusted platform module) 的虚拟机远程证明机制。

文献[3]提出直接使用 TCG 的远程证明协议会受到伪装攻击的威胁, 研究该攻击路径并改进了远程证明协议; 文献[4]首次提出布谷鸟攻击模型, 与文献[3]提到的攻击原理类似; 文献[5]利用串空间模型对远程证明协议进行分析, 对布谷鸟攻击进行了实验验证; 文献[6]提出用于检测布谷鸟攻击的时序模型, 但每次远程证明需要重复进行 TPM_Quote 运算, 消耗物理 TPM 芯片计算资源; 文献[7]提出 5 条设计原则用于指导远程证明系统的开发; 文献[8]以用户空间字符设备 (CUSE, character device in user space) 模拟 TPM, 向虚拟机提供 vTPM 设备, 基于 TCG 的远程证明方案提出大规模虚拟机远程证明模型, 但没有考虑布谷鸟攻击的影响; 文献[9]提出基于属性的虚拟机远程证明方案, 利用虚拟机自省 (VMI, virtual machine introspection) 技术获取虚拟机的特定属性; 文献[10]针对 TPM 性能瓶颈导致的远程证明协议只支持有限数量虚拟机的问题, 提出了基于虚拟机监视器 (VMM, virtual machine monitor) 的远程证明设计框架, 但没有给出具体的实现方案, 并且设计框架中忽略了虚拟机和远程验证证据之间的关联性。

针对以上问题, 本文基于 VMI 技术设计了虚拟机远程证明方案, 主要贡献如下。

1) 利用 VMI 技术在 VMM 中获取虚拟机远程验证证据, 并与虚拟机进行有效绑定。

2) 重新设计了远程证明协议的证据收集阶段, 利用物理 TPM 保证虚拟机远程验证证据的完

整性。

3) 实现了原型系统, 证明该方案可以有效验证虚拟机的运行状态、提高虚拟机远程证明的性能、消除虚拟机远程证明过程中在虚拟机内执行布谷鸟攻击的路径。

2 相关知识

2.1 虚拟机自省

自省是在虚拟机外部对虚拟机内部软件进行分析和监控的一类技术, Garfinkel 等^[11]第一次提出虚拟机自省的概念, 并利用 VMI 技术构建了基于 Hypervisor 的入侵检测系统。

VMI 技术可以在 Hypervisor 层访问虚拟机的内存和硬件状态信息, 然而这些信息需要经过高级语义的重构才能够反映虚拟机内部软件的状态和行为, 该问题被称为语义鸿沟。根据解决语义鸿沟问题所使用方法的不同, 文献[12]将现有的 VMI 技术分为 4 类: 基于目标虚拟机的依赖型自省、基于安全虚拟机的依赖型自省、基于软件结构知识的独立型自省和基于硬件结构知识的独立型自省。KVM 下主流的 VMI 开源项目 LibVMI^[13]主要借助于虚拟机的内核符号表及内核重要结构体的成员偏移量来解决语义鸿沟问题, 属于基于软件结构知识的独立型自省方式, 其可以支持简单的虚拟机内部状态的监控, 如进程链表、内核模块链表等。通过扩展 LibVMI 拥有的软件结构知识, 增强 LibVMI 对于二进制内存的高级语义翻译能力, 可以得到更多的虚拟机内部的实时状态信息。

2.2 虚拟机远程证明

TCG 提出的虚拟机远程证明机制分为两部分: 验证虚拟机内部状态、验证虚拟机监视器的状态, 基本的运行流程如图 1 所示。

协议运行前, 宿主机和虚拟机分别向私有证书颁发机构申请身份证明密钥 (AIK, attestation identity key) 证书和虚拟机身份证明密钥 (vAIK, virtual machine's attestation identity key) 证书。

1) 挑战者向某个虚拟机请求完整性验证证据, 并传输随机值 GuestNonce, 位于虚拟机内部的远程证明代理 RA_Agent 接收该请求。

1a) RA_Agent 采集完整性度量架构 (IMA,

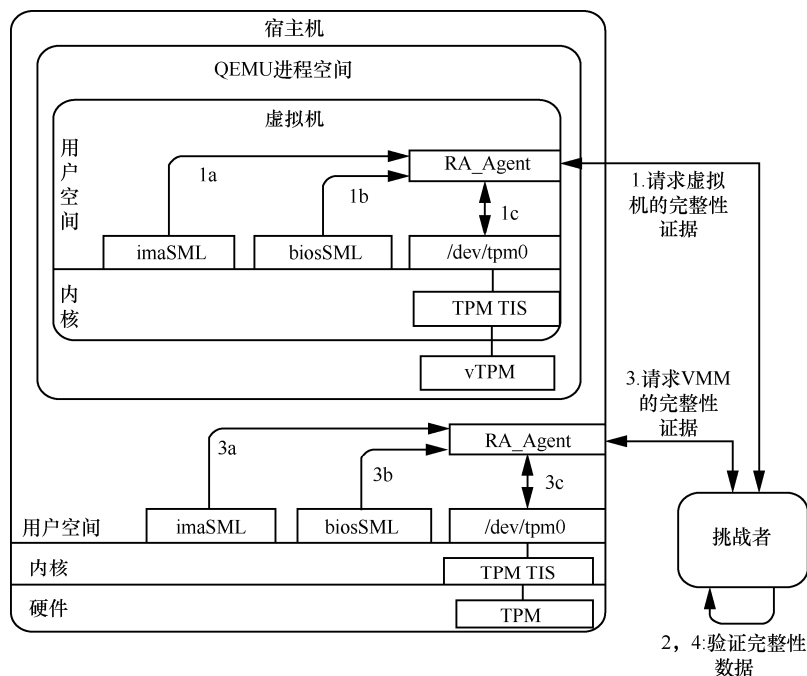


图 1 虚拟机远程证明

integrity measurement architecture) 模块的度量日志 imaSML。

1b) RA_Agent 采集 SeaBIOS 和 TrustedGrub 的度量日志 biosSML。

1c) RA_Agent 采集 vPCR 的值，并利用本地的 vTPM 执行 TPM_Quote 操作，即使用 vAIK 证书对 GuestNonce 和 vPCR 值进行签名。

2) RA_Agent 将采集的完整性证据发送给挑战者，然后挑战者依次验证 vPCR 完整性、imaSML 完整性、biosSML 完整性，进而根据具体的日志内容判定虚拟机当前状态。

3) 挑战者向虚拟机所在的宿主机请求 VMM 的完整性证据, 并传输随机值 HostNonce, 位于宿主机内部的远程证明代理 RA_Agent 接收该请求。

3a) RA_Agent 采集 IMA 的度量日志 imaSML。

3b) RA_Agent 采集 BIOS 和 TrustedGrub 的度量日志 biosSML。

3c) RA_Agent 采集相应 PCR 的值, 并利用本地的 TPM 执行 TPM_Quote 操作, 即使用 AIK 证书对 HostNonce 和 PCR 值进行签名。

4) RA_Agent 将采集的 VMM 完整性证据发送给挑战者，然后挑战者依次验证 PCR 完整性、

imaSML 完整性、biosSML 完整性, 进而根据具体的日志内容判定 VMM 当前状态。

2.3 布谷鸟攻击

文献[4]首次提出布谷鸟攻击，它是一种针对远程证明过程的中间人攻击方式，其在虚拟机远程证明过程中的利用方式如图 2 所示，流程如下：

- 1) 攻击者申请一台正常的虚拟机 C，入侵目标虚拟机 M，并劫持虚拟机 M 上的远程证明代理软件 RA_Agent；
- 2) 当挑战者对虚拟机 M 进行远程证明时，虚拟机 M 上的 RA_Agent 冒充挑战者将请求转发给正常虚拟机 C，利用正常虚拟机 C 的完整性证据来欺骗挑战者；
- 3) 挑战者把正常虚拟机 C 的完整性证据当成被劫持虚拟机 M 的完整性证据，认为虚拟机 M 的状态是正常的。

3 方案设计

3.1 总体设计

本文基于 VMI 技术对于虚拟机的远程证明过程进行了改进，主要思路是：将虚拟机内部的远程证明代理工作下移到宿主机的 VMM 中，利用 VMI 技术实现，如图 3 所示，VMI_Agent 就是在 VMM 中完成此工作的代理。

具体的运行过程如下。

1) 挑战者向宿主机请求虚拟机和 VMM 的

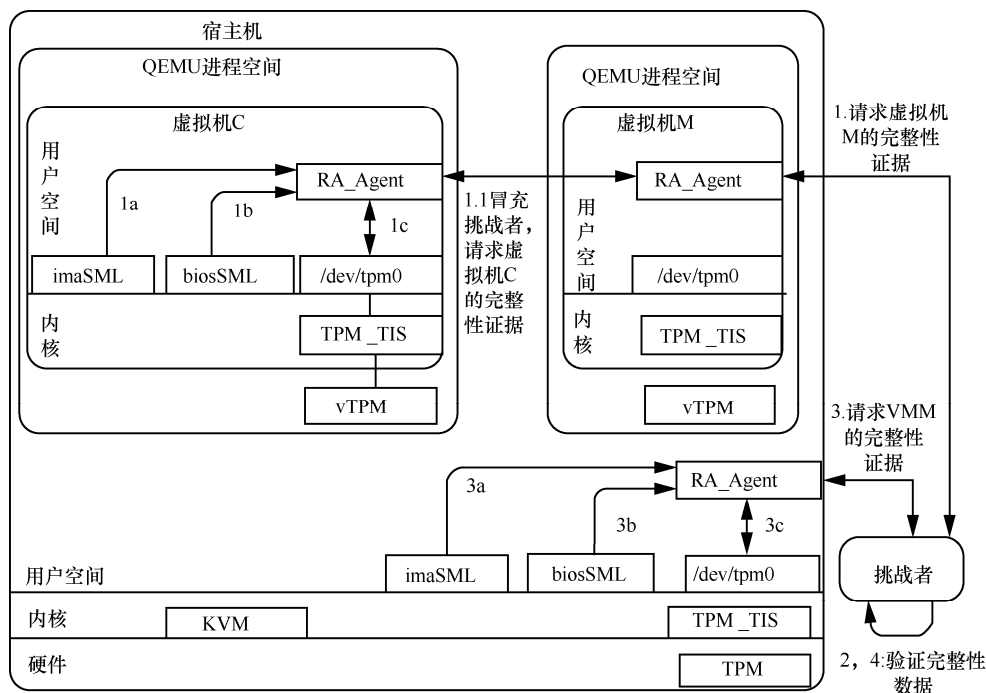


图 2 布谷鸟攻击

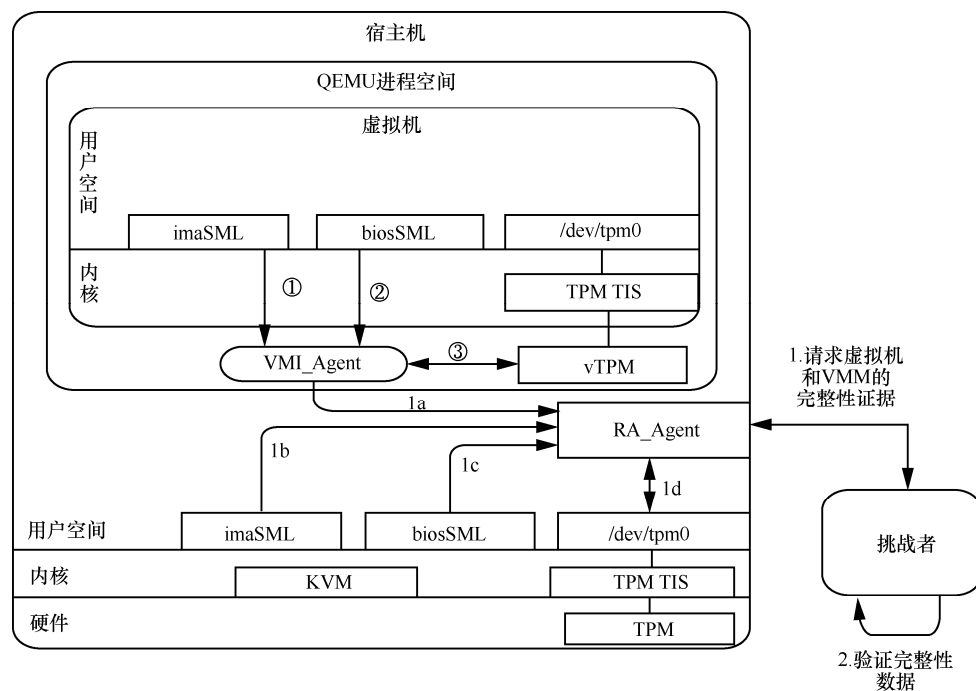


图 3 总体设计

完整性证据，并传输随机值 Nonce，位于宿主机上的 RA_Agent 接收该请求。

1a) RA_Agent 从 VMI_Agent 处获取虚拟机的完整性证据。

① VMI_Agent 通过 VMI 技术获取虚拟机内 IMA 的度量日志 imaSML，获取该日志的具体流

程将在 3.3.2 节给出。

② VMI_Agent 通过 VMI 技术获取虚拟机内 SeaBIOS 和 TrustedGrub 的度量日志 biosSML，获取该日志的具体流程将在 3.3.1 节给出。

③ VMI_Agent 截获虚拟机和 vTPM 的交互过程，采集 vPCR 值，获取 vPCR 值的具体流程

将在 3.2 节给出。

1b) RA_Agent 获取宿主机 IMA 的度量日志 imaSML。

1c) RA_Agent 获取宿主机 BIOS 和 TrustedGrub 的度量日志 biosSML。

1d) RA_Agent 采集 TPM 的 PCR 值, 并利用本地的 TPM 芯片执行 TPM_Quote 操作, 即使用 AIK 证书对 Nonce、PCR 值和 vPCR 值进行签名, 具体流程将在 3.4 节给出。

2) RA_Agent 将完整性证据发送给挑战者, 然后挑战者验证证据完整性, 并根据远程验证证据判定 VMM 状态、虚拟机状态。

3.2 获取虚拟机的 vPCR

正常虚拟机和其 vTPM 交互的过程如图 4(a) 所示, 虚拟机内核中的 TPM_TIS 驱动与 VMM 中实现的 vTPM 后端设备进行交互, 通过 vTPM 后端驱动向虚拟机提供的接口发送命令和接收响应; 同时虚拟机内核中的 TPM_TIS 驱动向虚拟机用户空间提供 /dev/tpm0 接口, 以供用户空间的应用程序使用 VMM 中实现的 vTPM 后端设备。

如图 4(b) 所示, VMI_Agent 在 VMM 层截获虚拟机内核的 TPM_TIS 驱动和 vTPM 后端设备交互的消息, 然后判断当前执行的 TPM 命令, 如果是 vPCR 相关的操作, VMI_Agent 则通过执行 TPM_ReadPCR 操作更新其维护的 vPCR 列表。具体的执行流程如下。

1) 虚拟机内应用程序向 TPM_TIS 驱动发送 TPM 命令流 C_{normal} 。

2) VMI_Agent 截获 TPM_TIS 驱动发送给 vTPM 后端驱动的 C_{normal} , 然后解析 C_{normal} , 如果该命令与 vPCR 操作相关, 则标记 ReadAgain 为 True, 否则标记为 False。

3) VMI_Agent 将 C_{normal} 发送给 vTPM 后端设备执行。

4) VMI_Agent 截获 vTPM 后端设备返回给 TPM_TIS 的响应流 R_{normal} 。

5) VMI_Agent 判断 ReadAgain 的值, 如果为 True, 则构造一个 TPM_ReadPCR 的命令 $C_{readpcr}$, 并发送给 vTPM 后端设备; 如果为 False, 则进入步骤 7)。

6) VMI_Agent 接收 vTPM 后端设备的响应流 $R_{readpcr}$, 解析 $R_{readpcr}$, 并更新其维护的 vPCR 列表。

7) VMI_Agent 将响应流 R_{normal} 返回给虚拟机内核的 TPM_TIS 驱动。

3.3 获取虚拟机的度量日志

3.3.1 获取虚拟机内的 biosSML 日志

biosSML 日志是 BIOS 用来保存度量历史记录日志, 同时 TrustedGrub 利用相应的中断向该日志中写入自身的度量历史记录, 因此, 该日志记录了 BIOS 和 TrustedGrub 这 2 个组件的度量内容。BIOS 将计算机主板上所有硬件的信息以高级配置和电源管理接口 (ACPI, advanced configuration and power management interface) 表的形式提供上层操作系统, 每个 ACPI 表都有特定的标识符。SeaBIOS 是 KVM 虚拟化平台中 QEMU 用来模拟 BIOS 的开源项目, 其集成了对于 vTPM

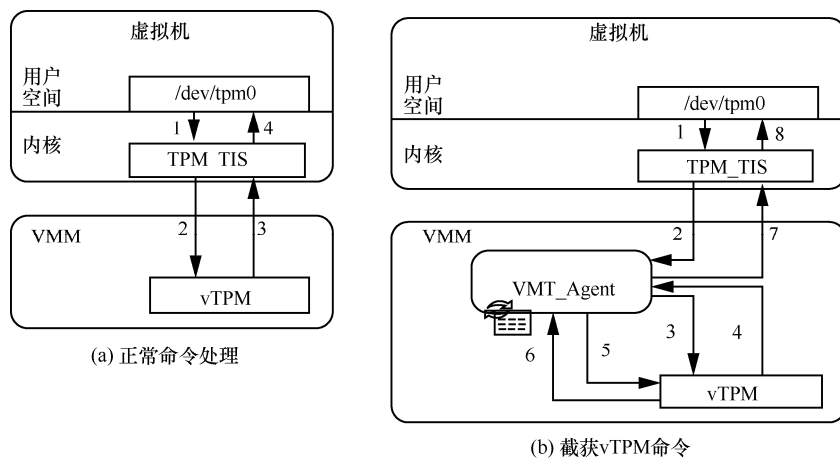


图4 虚拟机 vPCR 的获取

的支持，而且按照 TCG 规定的 ACPI 表格式^[14]向上层虚拟机提供关于 vTPM 硬件相关的信息，如厂商、版本、biosSML 度量日志起始地址和长度等。目前，SeaBIOS 使用 TPM1.2 客户端版本的 ACPI 表，其标识符为“TCPA”。

从虚拟机的角度来看，SeaBIOS 提供的 ACPI 表之间是有关联的，如图 5 所示，可以根据 ACPI 表的根来检索其他 ACPI 表，而该 ACPI 表的根则由虚拟机内核符号表中导出的符号 `acpi_gbl_root_table_list` 来指定。

biosSML 日志的获取过程如下。

1) VMI_Agent 根据内核符号 `acpi_gbl_root_table_list` 的地址读取 ACPI 表的根所在结构体 `rootACPI` 的内容。

2) VMI_Agent 解析结构体 `rootACPI` 的内容，获取 ACPI 表数组的起始地址 `SACPI_array`。

3) VMI_Agent 依次读取从地址 `SACPI_array` 开始的每个 ACPI 表的内容，假设第 i 个 ACPI 表的内容是 C_i 。

4) VMI_Agent 解析当前内容 C_i ，判断其标识符是不是“TCPA”，如果是，则按照 TCG 规定的 TPM 的 ACPI 表格式解析当前内容 C_i ，取出 biosSML 日志在虚拟机物理内存中的地址 `AddrbiosSML` 和日志最大长度 `MaxLbiosSML`，进入步骤 5)，否则，返回步骤 3)。

5) VMI_Agent 访问从虚拟机物理地址

`AddrbiosSML` 开始的长度为 `MaxLbiosSML` 的一段内存区域 `MembiosSML`，按照日志的格式解析 `MembiosSML`，得到虚拟机内部 biosSML 度量日志。

3.3.2 获取虚拟机内的 imaSML 日志

IMA 是 Linux 内核的完整性度量架构，当加载应用程序、动态链接库或者内核模块时，IMA 对加载对象的重要配置和代码段进行度量，将度量结果扩展到 TPM 的 PCR10 上，并将度量内容记录到日志 imaSML 中，用于远程证明。

imaSML 是一种特殊的日志文件，其本质上是 IMA 提供给上层应用程序的日志内容读取接口，真实的内容被存储在 IMA 维护的一段内存中。IMA 使用链表来维护度量日志的内容，日志的每一条度量记录为链表上的一个节点，整个链表可以通过内核中导出的符号 `ima_measurements` 来索引。

imaSML 日志的获取过程如下。

1) VMI_Agent 根据内核符号 `ima_measurements` 读取该符号所在结构体 `Mima` 的内容。

2) VMI_Agent 解析结构体 `Mima` 的内容，获取日志链表的表头节点地址 `HimaSML`。

3) VMI_Agent 依次读取 `HimaSML` 开始的日志链表上的每一个节点，直至链表结尾，假设当前读取的第 i 个节点的内容为 C_i 。

4) VMI_Agent 解析内容 C_i ，然后获取当前日志记录在虚拟机内核地址空间中的位置 `Addri`。

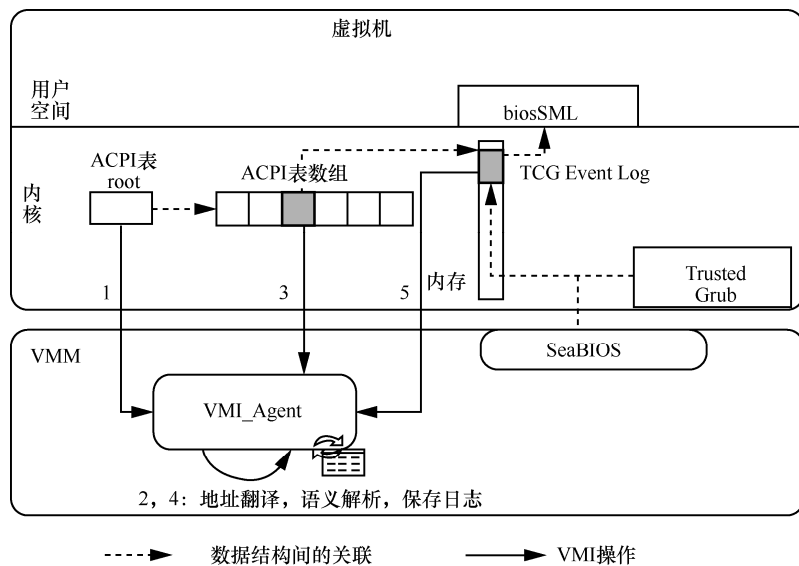


图 5 biosSML 日志的获取

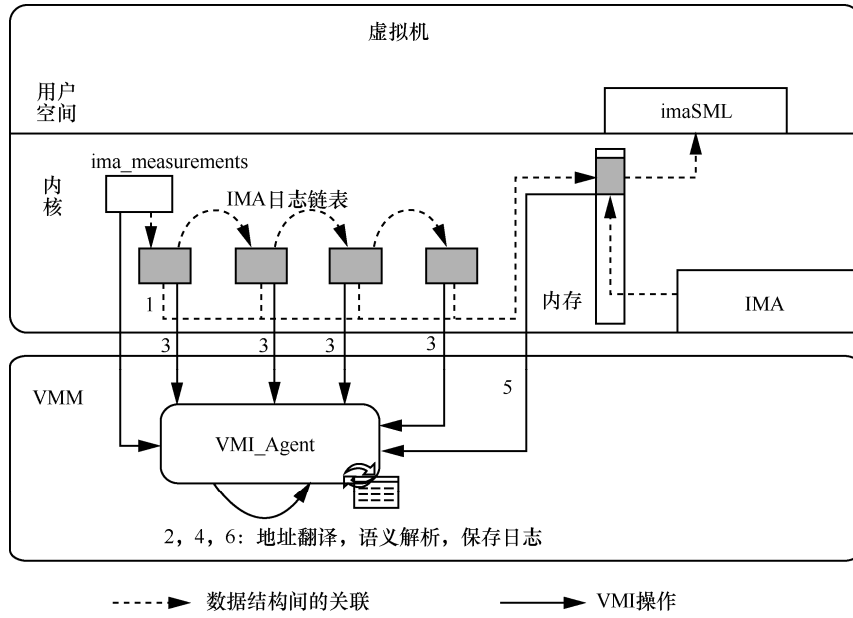


图6 imaSML 日志的获取

5) VMI_Agent 读取虚拟机内核地址空间中地址 Addr_i 开始的一段内存 Mem_i 。

6) VMI_Agent 解析获得的内存 Mem_i ，根据解析的结果更新其维护的 imaSML 日志列表，然后返回步骤 3)。

3.4 远程证明协议的证据收集阶段

远程证明协议是挑战者与远程证明代理交互所遵循的规范，可以分为两部分：证据收集阶段、证据验证阶段。由于本文方案的证据收集方法不同于原本 TCG 的虚拟机远程证明方案，所以需要修改 TCG 提出的远程证明协议的证据收集阶段（如图 7 所示）来规范本文方案各个组件的工作方式。

协议运行前，宿主机向私有证书颁发机构申请 AIK 证书。

操作原语设定：Challenger 代表挑战者，RA_Agent 代表宿主机的认证代理，VMI_Agent 代表虚拟机证据收集代理，TPM 代表宿主机上的物理可信平台模块，Hash 代表一种散列算法，Sig 代表一种签名算法，Ver 代表一种签名验证算法， Cert_{AIK} 代表宿主机申请的身份认证密钥证书， K_{AIK} 代表宿主机 TPM 生成的身份认证密钥。

具体交互过程如下：

1) Challenger->RA_Agent: Nonce;

2) RA_Agent->VMI_Agent: $\text{vm}_i \parallel \text{Nonce}$ ，其中， $i \in (0, m]$ ，若 $i > m$ ，则进入步骤 7)；

3) VMI_Agent->RA_Agent: $\text{vPCR} \parallel \text{vSML} \parallel H_{\text{vm}_i} \parallel \text{Hash}(\text{UUID}_{\text{vm}_i})$ ，其中， $H_{\text{vm}_i} = \text{Hash}(\text{vPCR}_0 \parallel \text{vPCR}_1 \parallel \dots \parallel \text{vPCR}_M \parallel \text{Hash}(\text{UUID}_{\text{vm}_i}) \parallel \text{Nonce})$ ， $M \in [0, 24]$ ；

4) RA_Agent->TPM: $\text{PCR}_{\text{index}} \parallel H_{\text{vm}_i}$ ；

5) TPM->RA_Agent: $\text{vm}_i_S_{\text{AIK}} \parallel H_{\text{vm}_i}$ ， $\text{vm}_i_S_{\text{AIK}} = \text{TPM_Quote}(\text{PCR}_{\text{index}}, H_{\text{vm}_i}) = \text{Sig}(\text{K}_{\text{AIK}}, \text{Hash}(\text{PCR}_{\text{index}} \parallel H_{\text{vm}_i}))$ ；

6) RA_Agent: $P_i = \text{vPCR} \parallel \text{vSML} \parallel \text{Hash}(\text{UUID}_{\text{vm}_i}) \parallel \text{vm}_i_S_{\text{AIK}}$ ，然后 $i++$ ，进入步骤 2)；

7) RA_Agent->TPM: $\text{PCR}_{\text{index}} \parallel \text{Nonce}$ ；

8) TPM->RA_Agent: $S_{\text{AIK}} \parallel \text{Nonce}$ ， $S_{\text{AIK}} = \text{TPM_Quote}(\text{PCR}_{\text{index}}, \text{Nonce}) = \text{Sig}(\text{K}_{\text{AIK}}, \text{Hash}(\text{PCR}_{\text{index}} \parallel \text{Nonce}))$ ；

9) RA_Agent: $P_0 = \text{PCR} \parallel \text{SML} \parallel \text{Nonce} \parallel S_{\text{AIK}}$ ， $\text{msg} = P_0 \parallel P_1 \parallel \dots \parallel P_m$ ；

10) RA_Agent->Challenger: msg;

11) Challenger: $\text{Ver}(\text{Cert}_{\text{AIK}}, P_i)$ ，其中， $i \in [0, m]$ ，然后进入证据验证阶段。

接下来对上述每一步进行详细解释，如下。

步骤 1：挑战者向宿主机上的认证代理 RA_Agent 发送随机值 Nonce。

步骤 2：RA_Agent 向 VMI_Agent 发送虚拟机编号 vm_i 和 Nonce，其中， $i \in (0, m]$ ， m 为一次远程证明可以验证的虚拟机个数。

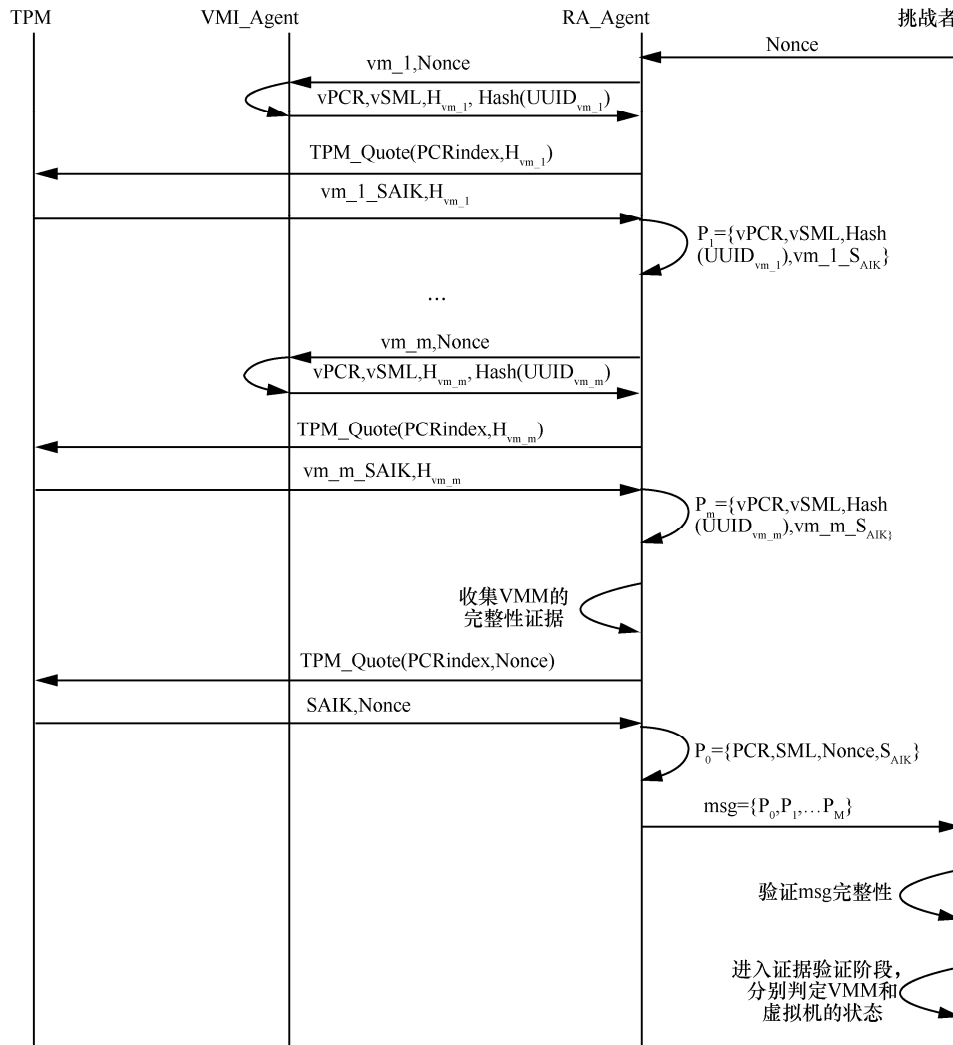


图 7 虚拟机远程证明协议证据收集阶段

步骤 3: VMI_Agent 采集编号为 vm_i 的虚拟机的 vPCR、vSML, 并计算该虚拟机在云平台上唯一标识的散列值 $\text{Hash}(\text{UUID}_{vm_i})$, 避免直接使用 UUID 泄露虚拟机身份, 然后根据 vPCR 和 Nonce 计算得到 H_{vm_i} , 用以将远程证明证据和对应虚拟机进行有效绑定, 计算方法为 $H_{vm_i} = \text{Hash}(vPCR_0 \parallel vPCR_1 \parallel \dots \parallel vPCR_M \parallel \text{Hash}(\text{UUID}_{vm_i}) \parallel \text{Nonce})$, 其中, M 为编号 vm_i 的虚拟机的 vPCR 个数, 最后将 $\{vPCR \parallel vSML \parallel H_{vm_i} \parallel \text{Hash}(\text{UUID}_{vm_i})\}$ 返回给 RA_Agent。

步骤 4: RA_Agent 给 TPM 发送命令 $\text{TPM_Quote}(\text{PCR}_{\text{index}}, H_{vm_i})$, 利用 AIK 密钥对 H_{vm_i} 和当前宿主机的 $\text{PCR}_{\text{index}}$ 值进行签名, 利用物理 TPM 保证虚拟机远程验证证据的完整性。

步骤 5: TPM 将签名结果 $\{vm_i_SAIK \parallel H_{vm_i}\}$

返回给 RA_Agent。

步骤 6: RA_Agent 打包编号 vm_i 的虚拟机的完整性证据 $P_i = \{vPCR \parallel vSML \parallel \text{Hash}(\text{UUID}_{vm_i}) \parallel vm_i_SAIK\}$, 然后 i 自增, 并判断 i 和 m 的大小关系, 如果 $i > m$, 则进入步骤 7, 否则进入步骤 2。

步骤 7: RA_Agent 向 TPM 发送命令 $\text{TPM_Quote}(\text{PCR}_{\text{index}}, \text{Nonce})$, 利用 AIK 密钥对 Nonce 和当前宿主机的 $\text{PCR}_{\text{index}}$ 值进行签名。

步骤 8: TPM 返回签名结果 $\{SAIK \parallel \text{Nonce}\}$ 。

步骤 9: RA_Agent 打包宿主机上 VMM 的完整性证据 $P_0 = \{\text{PCR} \parallel \text{SML} \parallel \text{Nonce} \parallel S_{AIK}\}$, 然后打包虚拟机和 VMM 的完整性证据 $\text{msg} = \{P_0 \parallel P_1 \parallel \dots \parallel P_m\}$ 。

步骤 10: RA_Agent 将 msg 发送给挑战者。

步骤 11: 挑战者首先根据 AIK 证书验证 msg 中每个 $P_i(i \in [0, m])$ 的完整性, 然后进入证据验证阶段, 分别根据 VMM 和每个虚拟机的度量日志和 PCR/vPCR 值判定各自的当前状态, 并生成虚拟机远程证明报告。

4 安全性分析

本文提出基于 VMI 的虚拟机远程证明方案, 首先将位于虚拟机内部的远程证明代理的工作放到 VMM 中完成, 然后改进传统的远程证明协议以保证宿主机中采集到的虚拟机和 VMM 的证据完整性, 该方案可以有效抵抗在虚拟机内部执行的布谷鸟攻击, 并且减少远程证明过程中涉及的 AIK 证书的生成数量, 降低私有证书颁发机构的负载。

4.1 消除布谷鸟攻击路径

如图 2 所示, 布谷鸟攻击之所以会成功, 是因为攻击者修改了目标虚拟机 M 上的远程证明代理程序, 将挑战者引导向攻击者自己的正常虚拟机 C, 成功欺骗挑战者。而本文方案在宿主机 VMM 中借助 VMI 技术获取虚拟机的度量日志和 vPCR 值, 不需要在虚拟机内部部署远程证明代理, 因此, 在虚拟机内部执行布谷鸟攻击的这条攻击路径是不存在的。

如果攻击者可以入侵宿主机, 那么针对宿主机远程证明过程的布谷鸟攻击是可以成功的, 本文建议使用文献[3]所提出的方法增强宿主机远程证明过程的安全。

4.2 降低私有证书颁发机构的负载

在 TCG 提出的虚拟机远程证明过程中, 需要依次对虚拟机和 VMM 进行远程证明, 而且每次证明都会由私有证书颁发机构为虚拟机和 VMM 分别颁发一个临时的 vAIK/AIK 证书, 如果同时对多个虚拟机进行远程证明, 会导致私有证书颁发机构的负载过大甚至产生拒绝服务攻击。本文方案将虚拟机的完整性证据采集代理放到 VMM 中执行, 取消了虚拟机远程证明过程中私有证书颁发机构为虚拟机颁发 vAIK 证书的步骤, 减少了远程证明过程中产生 AIK 证书的数量, 降低了虚拟机远程证明过程中私有证书颁发机构的负载以及对私有证书颁发机构实施拒绝服务攻击产生

的风险。

5 原型系统实现与测试

5.1 系统实现

本文首先在 KVM 虚拟化平台上为每一个虚拟机提供基于 Libtpms^[15] 函数库的 vTPM 设备, 然后在 VMM 中实现 VMI_Agent, 通过修改 IBM 的远程证明开源项目 TPM 2.0 Attestation Client Server 实现 RA_Agent 和挑战者, 最后以表 1 的配置为基础环境设计并实现了本文方案的原型系统。

表 1 系统配置

服务器	个数	CPU	内存	磁盘	TPM
控制节点	1	Intel(R)Core i5-6400	64 GB	1 TB	TPM2.0
计算节点	2	Intel(R)Core i5-6400	64 GB	1 TB	TPM2.0

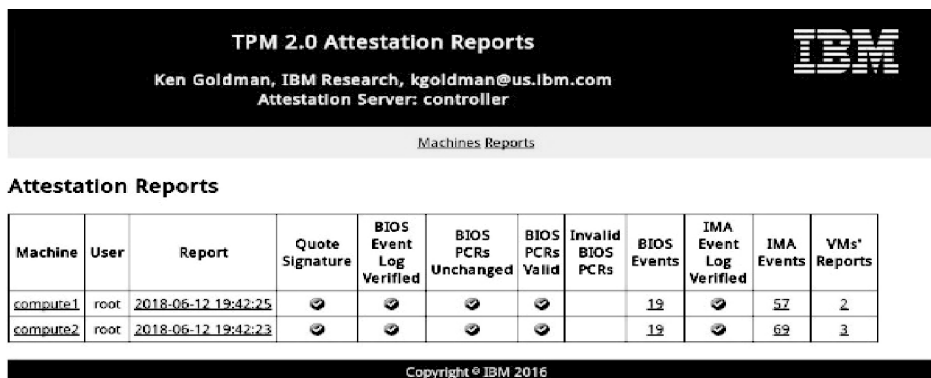
5.2 功能测试与分析

功能测试主要测试基于 VMI 的虚拟机远程证明方案的有效性。

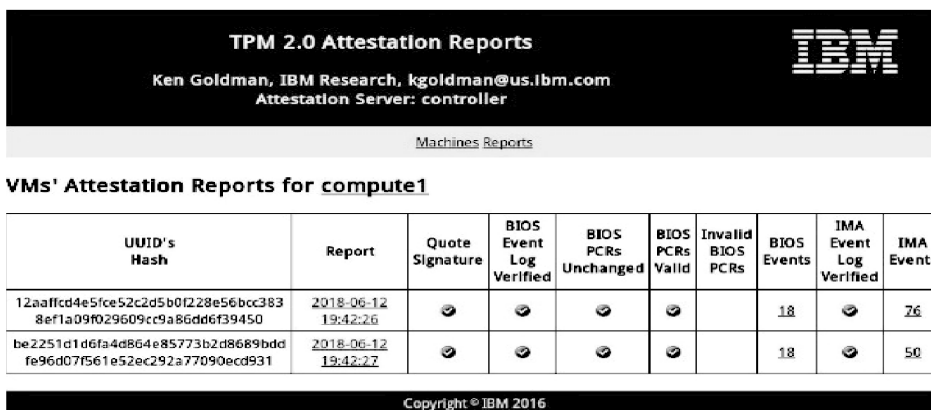
本文对 IBM 的远程证明开源项目进行了修改, 与原系统相比, 修改后的系统调整了宿主机视图 (如图 8(a)所示), 并且增加了虚拟机视图 (如图 8(b)所示), 然后通过部署在控制节点上的挑战者向计算节点发起虚拟机远程证明请求。图 8(a)宿主机视图展示了两台计算节点的验证结果都是正常的, 验证结果的最后一个表项 “VMs’ Reports” 链接了对该计算节点上运行的虚拟机的远程证明结果, 表项的值为虚拟机的个数。图 8(b)虚拟机视图展示了计算节点 computer1 上虚拟机的远程证明结果, 虚拟机视图的验证内容和宿主机视图是一致的, 不同的是虚拟机视图中验证结果的唯一标识是虚拟机 UUID 的散列值。

5.3 性能测试与分析

性能测试主要针对 2 种方案的比较: 基于 VMI 的虚拟机远程证明方案和 TCG 提出的虚拟机远程证明方案。本文在计算节点设置 10 组运行不同数量虚拟机的场景, 对于每个场景分别实施 2 种方案的远程证明, 在特定场景下对每个方案分别进行 20 次远程证明, 最后计算耗时平均值, 结果如图 9 所示。



(a) 宿主机视图



(b) 虚拟机视图

图 8 功能测试

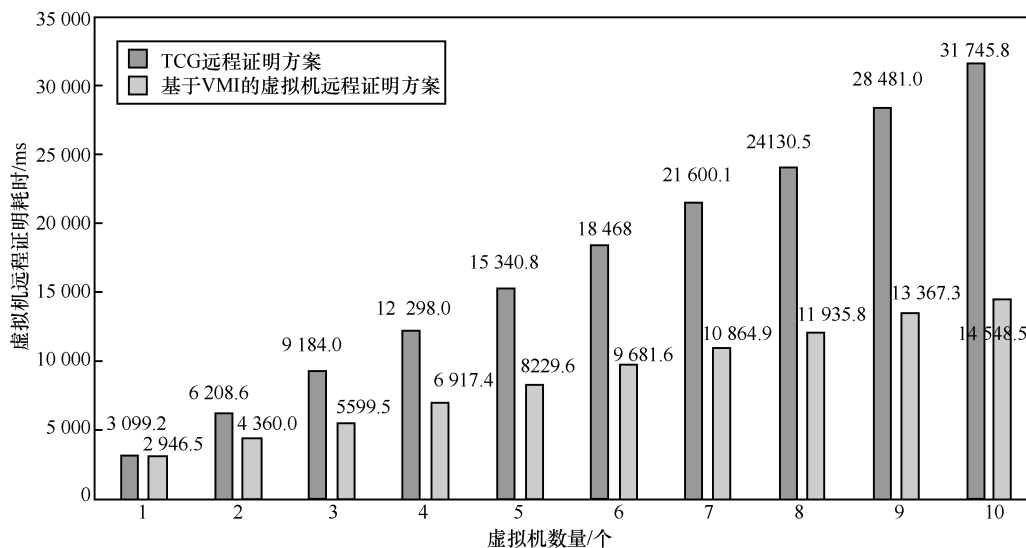


图 9 性能测试

随着虚拟机数量的增加, 2 种远程证明方案的耗时均呈线性增长的趋势。假设 2 种方案耗时均满足函数 $Y=aX+b$, 其中, X 是虚拟机个数, Y 是远程证明的耗时, a 、 b 为参数。通过线性回归

拟合可得, 基于 VMI 的虚拟机远程证明方案耗时的参数 a 为 1 287.55, TCG 虚拟机远程证明方案耗时的参数 a 为 3 148.44, 由此可见, 当 X 比较大时, 本文方案性能较好。

表2给出了一次完整的虚拟机远程证明过程中不同方案的通信开销、可以验证的虚拟机数量、TPM_Quote执行次数和产生的AIK证书数量。其中,在某个远程证明协议内部发送和接收一次消息称为一次交互(连接建立和释放时传输的消息不计算), m 是本文方案一次远程证明可以验证的虚拟机个数。TCG RA是TCG提出的虚拟机远程证明方案,RIRP RA是文献[3]提出的抵抗布谷鸟攻击顽健性的远程证明方案,VMI RA是本文提出的基于VMI的虚拟机远程证明方案。

表2 方案性能对比

方案名称	交互 轮数	验证的虚拟机 数量	TPM_Quote	AIK 证书
TCG RA	2	1	2	2
RIRP RA	4	1	2	2
VMI RA	1	$m(m \geq 1)$	$m+1(m \geq 1)$	1

TCG RA在一次虚拟机远程证明过程中,分别对虚拟机、VMM进行一次远程证明,因此需要2轮交互,每一轮都需要1个AIK证书和1次TPM_Quote操作,1次虚拟机远程证明过程只能验证1台虚拟机状态;RIRP RA为了解决布谷鸟攻击的威胁,在对虚拟机、VMM进行远程证明时增加了Diffie-Hellman密钥交换协议,因此需要4轮交互,TPM_Quote操作次数和AIK证书数量与TCG RA一样,一次虚拟机远程证明过程也只能验证1台虚拟机状态;本文方案在一次虚拟机远程证明过程中,通过VMI_Agent得到了虚拟机的远程验证证据,因此只需要1轮交互、1个AIK证书,由于一次虚拟机远程证明过程可以验证 m 台虚拟机状态,而每个虚拟机远程验证证据都需要计算一次TPM_Quote,再加上对于宿主机的远程证明过程,共需要 $m+1$ 次TPM_Quote操作。

由表2可以得出,对 K 台虚拟机进行远程证明,TCG RA和RIRP RA需要的AIK证书的数量是 $2K$ 个,而VMI RA需要的AIK证书的数量是 $\left\lceil \frac{K}{m} \right\rceil$ 个,由此可见本文方案大大减少了远程证明过程中AIK证书的产生数量,降低了私有证书颁发机构的负载。进一步分析,对 K 台虚拟机进行

远程证明,TCG RA需要 $2K$ 轮交互、 $2K$ 次TPM_Quote,RIRP RA需要 $4K$ 轮交互、 $2K$ 次TPM_Quote,VMI RA需要 $\left\lceil \frac{K}{m} \right\rceil$ 轮交互、 $(m+1)\left\lceil \frac{K}{m} \right\rceil$ 次TPM_Quote,因此,当 K 值较大时,VMI RA的性能明显优于前2种方案。

6 结束语

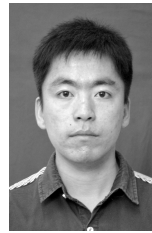
为了消除在虚拟机远程证明过程中产生的布谷鸟攻击,本文提出利用VMI技术在VMM中获取虚拟机远程验证证据的方法,重新设计了虚拟机远程证明协议的证据收集阶段。该方案可以消除虚拟机远程证明过程中在虚拟机内执行布谷鸟攻击的路径,减少远程证明过程中产生的AIK证书数量,进而降低私有证书颁发机构的负载,而且通过对原型系统的测试与分析可知,在虚拟机数量较多的情况下,本文方案的性能优于TCG提出的虚拟机远程证明方案。

参考文献:

- [1] 冯登国,秦宇,汪丹,等.可信计算技术研究[J].计算机研究与发展,2011,48(8):1332-1349.
FENG D G, QIN Y, WANG D, et al. Research on trusted computing technology[J]. Journal of Computer Research and Development, 2011, 48(8):1332-1349.
- [2] BERGER S, CACERES R, GOLDMAN K A, et al. vTPM: virtualizing the trusted platform module[C]//The 15th Conference on USENIX Security Symposium. 2006: 305-320.
- [3] STUMPF F, TAFRESCHI O, RÖDER P, et al. A robust integrity reporting protocol for remote attestation[C]//The 2nd Workshop on Advances in Trusted Computing. Tokyo, Japan, 2006.
- [4] PARNO B. Bootstrapping trust in a trusted platform[C]//The USENIX Workshop on Hot Topics in Security. 2008.
- [5] 冯伟,冯登国.基于串空间的可信计算协议分析[J].计算机学报,2015,38(4):701-706.
FENG W, FENG D G. Analyzing trusted computing protocol based on the strand space model[J]. Chinese Journal of Computers, 2015, 38(4): 701-706.
- [6] FINK R A, SHERMAN A T, MITCHELL A O, et al. Catching the cuckoo: verifying TPM proximity using a quote timing side-channel[C]//The Trust and Trustworthy Computing. 2011: 294-301.
- [7] COKER G, GUTTMAN J, LOSCOCCO P, et al. Principles of remote attestation[C]//The International Journal of Information Security. 2010: 63-81.

- [8] BERGER S, GOLDMAN K, PENDARAKIS D, et al. Scalable attestation: a step toward secure and trusted clouds[C]//The IEEE International Conference on Cloud Engineering. 2015: 185-194.
- [9] ZHANG T, B. LEE R. Cloud Monatt: an architecture for security health monitoring and attestation of virtual machines in cloud computing[C]//The IEEE 42nd Annual International Symposium on Computer Architecture (ISCA). 2015: 362-374.
- [10] LAUER H, KUNTZE N. Hypervisor-based attestation of virtual environments[C]//The Ubiquitous Intelligence Computing Advanced and Trusted Computing Scalable Computing and Communications Cloud and Big Data Computing Internet of People and Smart World Congress. 2016:333-340.
- [11] GARFINKEL T, ROSENBLUM M. A Virtual machine introspection based architecture for intrusion detection[C]//The Network and Distributed Systems Security Symposium. 2003:191-206.
- [12] 李保琿, 方滨兴, 徐克付, 等. 虚拟机自省技术研究与应用进展[J]. 软件学报, 2016, 27(6): 1384-1401.
LI B H, FANG B X, XU K F, et al. Research and application progress of virtual machine introspection technology[J]. Journal of Software, 2016, 27(6):1384-1401.
- [13] 彭春洪. 基于 KVM 虚拟机的恶意行为检测系统设计与实现[D]. 成都: 电子科技大学, 2015.
PENG C H. The design and implementation of the malicious behavior detecting system based on KVM virtual machine[D]. Chengdu: University of Electronic Science and Technology of China, 2015.
- [14] DUFLOT L, LEVILLAIN O, MORIN B. ACPI: design principles and concerns[C]//The 2nd International Conference on Trusted Computing. 2009: 14-28.
- [15] HENDERSON A, YIN H, HAN H, et al. VDF: targeted evolutionary fuzz testing of virtual devices[C]//The International Symposium on Research in Attacks, Intrusions, and Defenses. 2017: 3-25.

[作者简介]



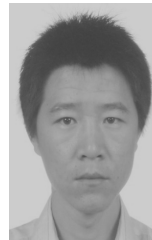
王伟 (1992-), 男, 山东聊城人, 四川大学硕士生, 主要研究方向为可信计算、虚拟化安全。



陈兴蜀 (1968-), 女, 贵州六枝人, 博士, 四川大学教授、博士生导师, 主要研究方向为云计算与大数据安全、可信计算与信息保障。



兰晓 (1990-), 女, 四川自贡人, 博士, 四川大学助理研究员, 主要研究方向为认证密钥协商协议、区块链安全。



金鑫 (1976-), 男, 辽宁营口人, 四川大学博士生, 主要研究方向为可信计算、虚拟化安全。