基于标签的 vTPM 私密信息保护方案

陈兴蜀[1,2]，王伟[1,3]，金鑫[1,3]

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

摘 要：虚拟可信平台模块是可信计算技术虚拟化的重要组件。vTPM 的私密信息存在被窃取、滥用的风险，为此，提出一种基于标签的安全保护方案。首先，为每个虚拟机建立 vTPM 标签，标签包括签名信息、加密信息、度量信息和状态信息。然后，基于 vTPM 标签的状态信息设计安全增强的 vTPM 动态迁移协议，保障迁移前后 vTPM 私密信息的机密性、完整性以及虚拟机与 vTPM 实例关联关系的一致性。实验表明，所提方案能够有效保护 vTPM 的私密信息，并且给虚拟机动态迁移带来的性能开销只有 19.36%。

关键词：可信计算；虚拟可信平台模块；TPM2.0；动态迁移

中图分类号：TP309.2

文献标识码：A

doi: 10.11959/j.issn.1000-436x.2018242

Label-based protection scheme of vTPM secret

CHEN Xingshu[1,2], WANG Wei[1,3], JIN Xin[1,3]

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

Abstract: The virtual trusted platform module (vTPM) played an important role in virtualization of trusted computing. According to security problems of existed vTPM, a protection scheme based on vTPM label was proposed. Firstly, a vTPM label was created for each virtual machine. This label had four main components, signature information, encryption information, measurement information and status information. Then, the security-enhanced vTPM dynamic migration protocol based on vTPM label status information was designed, to ensure the security of vTPM during live migration based on status information of vTPM label. Experiments show that the proposed scheme can protect vTPM secrets effectively and the increased performance cost during live migration is only 19.36%.

Key words: trusted computing, virtual trusted platform module, TPM2.0, live migration

1 引言

虚拟化平台 Xen、开源的系统虚拟化模块（KVM, kernel-based virtual machine）都有虚拟可信平台模块（vTPM, virtual trusted platform module）的实现方案，两者均涉及 TPM（trusted platform module）非易失性信息的模拟。由于 TPM 的非易失性信息存储了背书密钥（EK, endorsement key）等非对称密钥、访问口令等私密信息，因此本文将 vTPM 的非易失性信息称为 vTPM 的私密信息。

vTPM 私密信息面临的威胁主要表现在其机密性、完整性以及 vTPM 和虚拟机（VM, virtual machine）之间的关联关系上。文献[1]在 Xen 上采用设备驱动分离的方式实现 vTPM，每个 vTPM 实例的私密信息存储于根 vTPM 实例中，利用根 vTPM 实例的存储密钥对其进行加密存储。

收稿日期：2018−03−13；**修回日期**：2018−05−17
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, No.61802271）
Foundation Item: The National Natural Science Foundation of China (No.61802270, No.61802271)

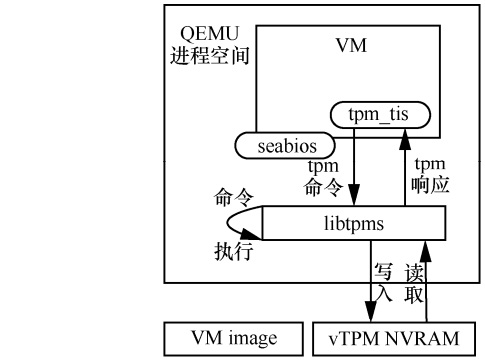
Hypervisor 维护了 vTPM 和虚拟机的关联列表，该 vTPM-VM 列表最终存放于 domain0 中，需要进一步的安全防护。文献[2]利用事务性同步扩展技术（TSX, transactional synchronization extension）对 vTPM 的私密信息进行安全防护，利用硬件 CPU 相关的 AES 主密钥进行加解密，由于该密钥不能被迁移，因此该方法不支持 vTPM 迁移。文献[3]借助于 Intel 的软件防护扩展技术（SGX, software guard extension）将 vTPM 的代码和数据存放到安全内存 enclave 中，利用基于 enclave 身份的密封机制加密存储 vTPM 私密信息，然而密封的数据不能被迁移到较新版本的应用或安全区中，不存在一种 vTPM 版本更新机制。文献[4]使用物理 TPM2.0 的可迁移密钥加密存储 vTPM 私密信息，基于可迁移密钥的复制机制实现了 vTPM 的安全迁移，但无法识别虚拟机配置为“vm_1_uuid，vm_2_镜像，vm_1_vTPM”下虚拟机 1 的 vTPM 私密信息被虚拟机 2 使用的情况。文献[5]建立管理中心维护 vTPM 和虚拟机的关联关系，建立 vTPM 实例文件的度量列表保证虚拟机关闭时 vTPM 私密信息的完整性，但无法保证虚拟机运行时 vTPM 私密信息的完整性。以上文献都没有提及虚拟机动态迁移中如何保护 vTPM 私密信息的安全。文献[6]提出一种基于 KVM 的 vTPM 动态迁移方案，但该方案只适用于配置共享存储的情况，而且没有关注迁移过程中 vTPM 的安全问题。文献[7]利用 SGX 技术的远程认证机制建立安全迁移信道，实现虚拟机（包括 vTPM）的安全动态迁移，但是需要修改 vTPM 的源码结构，改变了原本迁移双方的信息交互方式。文献[8-9]在迁移协议中加入新的安全验证与密钥生成机制，增强 vTPM 动态迁移的安全。文献[10]提出一种位于 TPM 和 vTPM 之间的密钥层次来解决 Xen 上虚拟机迁移造成的 vTPM 密钥失效的问题，该方案仅适用于暂停−迁移−重启的迁移模式，这种迁移模式不是真正意义上的动态迁移。

本文针对本地存储和动态迁移两个阶段提出一种新的 vTPM 私密信息保护方案，主要贡献如下：1) 提出 vTPM 标签结构，利用物理 TPM2.0 保护 vTPM 私密信息的机密性、完整性以及 vTPM 和虚拟机的对应关系；2) 基于 vTPM 标签的状态属性，在不改变迁移双方信息交互方式的情况下，设计了安全增强的 vTPM 动态迁移协议；3) 实现了原型系统，并进行了功能和性能测试。

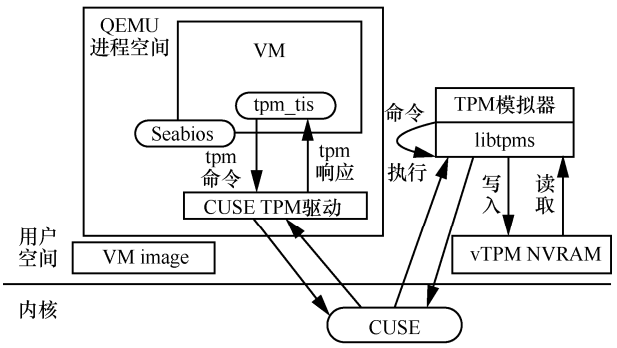
2　背景知识

2.1　KVM 虚拟化平台的全虚拟化 TPM

KVM 虚拟化平台的 TPM 虚拟化技术主要有 3 类：TPM passthrough 实现方式、基于 libtpms[11]的全虚拟化 TPM 实现方式以及 CUSE（character device in userspace）TPM[12]实现方式，其中，后两者属于全虚拟化 TPM 的实现方式。TPM passthrough 实现方式允许虚拟机直接使用物理 TPM，同一时刻物理 TPM 被单个虚拟机独占。基于 libtpms 的全虚拟化 TPM（如图 1(a)所示）在 QEMU（quick emulator）内部使用 libtpms 函数库模拟物理 TPM，可同时为每个虚拟机提供单独的 vTPM 设备，与物理 TPM 完全脱离关系。CUSE TPM（如图 1(b)所示）是基于 libtpms 的全虚拟化 TPM 的变种，在 QEMU 外部使用 libtpms 函数库来模拟物理 TPM，在宿主机上创建用户空间字符设备 /dev/vtpm0，提供 IOCTL 访问接口，在 QEMU 内部的 CUSE TPM 驱动利用该 IOCTL 访问 vTPM，与物理 TPM 完全脱离关系。



(a) 基于libtpms的全虚拟化TPM



(b) CUSE TPM

图 1　全虚拟化 TPM

KVM 虚拟化平台的全虚拟化 TPM 有如下不足：1) 采用 libtpms 函数库模拟物理 TPM 的全部功

能，与物理 TPM 完全脱离关系；2) 将 vTPM 私密信息存储在宿主机文件非易失存储器（NVRAM, non-volatile RAM）中，没有添加任何安全措施；3) 通过 QEMU 的命令行参数加载对应的 vTPM 实例，vTPM 和 VM 之间的关联关系薄弱。

2.2 TPM2.0 的密钥结构

TPM2.0 中包含 3 个持久性密钥层次[13]，即背书层次（endorsement hierarchy）、存储层次（storage hierarchy）和平台层次（platform hierarchy），与一个临时性密钥，即层次−空层次（NULL hierarchy）。每个密钥层次对应一个私有种子，即大随机数。背书私有种子（EPS, endorsement primary seed）产生背书密钥（EK, endorsement keys），存储私有种子（SPS, storage primary seed）产生存储根密钥（SRK, storage root keys），平台私有种子（PPS, platform primary seed）由平台固件使用，空私有种子（NS, NULL seed）是每次系统重启时产生的随机数。

本文将 TPM2.0 存储根密钥的两个非对称的不可迁移子密钥——RSA_local 和 RSA_mig，分别用于本地存储和动态迁移过程中的 vTPM 私密信息保护，如图 2 所示。另外，RSA_mig 需要被 CA 签名，形成对应的数字证书，用于 vTPM 动态迁移安全增强。

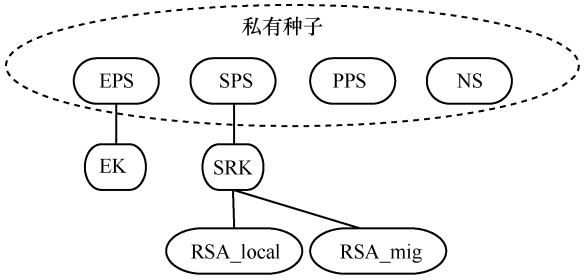


图 2 TPM2.0 密钥层次结构

3 方案设计

3.1 设计原则

文献[2,14-16]在 vTPM 与虚拟机的关联关系、vTPM 与其可信计算基（TCB, trusted computing base）[17]的依赖关系、vTPM 私密信息存储、基于证书的信任链扩展、vTPM 的密钥管理、vTPM 的迁移等 6 个方面对 vTPM 的实现提出了安全要求。

结合上述文献中的安全要求，本文认为 KVM 下的 vTPM 私密信息保护应该满足以下 3 点设计原则需求：1) vTPM 私密信息在本地存储时的机密性、完整性由硬件 TPM 保证；2) vTPM 和虚拟机保持一一对应的强关联关系；3) vTPM 支持动态迁移，并且迁移不会破坏以上两种安全需求。

3.2 vTPM 标签

vTPM 标签（vTPM_label）是本文提出的一种数据结构，是 vTPM 私密信息本地存储保护方案和动态迁移保护方案的核心元素。vTPM 标签的结构类似于数字证书，其完整性受硬件 TPM2.0 的签名保护。vTPM 标签结构及大小如图 3 所示。

vTPM标签（vTPM_label)365 B		
状态(status)		1 B
有效期(time)	开始时间(start)	32 B
	结束时间(end)	
加密信息(secret)		32 B
UUID的hash值(uuid_hash)		20 B
QEMU度量值(qemu_digest)		20 B
签名算法(alg)		4 B
标签签名值(sig)		256 B

图 3 vTPM 标签结构

vTPM 标签主要内容描述如下。

1) 状态（status）：vTPM 标签有两种存在状态，即本地状态和迁移状态；2) 有效期（time）：在时间段[start,end]内 vTPM 标签是有效的，过期的 vTPM 标签需要被重新生成；3) 加密信息（secret）：vTPM 私密信息保护所使用 key 的密文，利用物理 TPM 的密钥 RSA_local 对 key 进行加密；4) UUID（universally unique identifier）的 hash 值（uuid_hash）：UUID 是虚拟机在云平台中的唯一标识，UUID 的 hash 值建立了 vTPM 标签与虚拟机之间的一一对应关系，同时 secret 字段建立了 vTPM 标签和 vTPM 之间的一一对应关系，最终建立起 vTPM 和虚拟机之间的一一对应关系；5) QEMU 度量值（qemu_digest）：在 KVM 虚拟化平台中，QEMU 是虚拟机监控器（VMM, virtual machine monitor）的一部分，同时也是 vTPM 实例的访问者，该字段是对 QEMU 代码段进行散列运算得到的值，作为 vTPM 实例的访问者身份认证的证据，防止被篡改劫持的 QEMU 或其他恶意程序使用 vTPM 实例；6) 签名算法（alg）：标签签名值的生成算法；7) 标签的签名值（sig）：由 TPM2.0 的密钥 RSA_local 对以上字段进行签名得到的值，防止标签被恶意篡改。

vTPM 标签存在两种状态：本地状态和迁移状

态。其中，迁移状态是一种临时状态，只出现在迁移的过程中。

本地状态是指该 vTPM 标签的 sig 字段由本地 TPM2.0 的密钥 RSA_local 生成，标签中的 secret 字段由本地 TPM2.0 密钥 RSA_local 加密，即 vTPM_label.sig=RSA_local.$priv_{sign}$(vTPM_label)，vTPM_label.secret= RSA_local.$pub_{encrypt}$(key)。

迁移状态是指该vTPM标签的sig字段由迁移源主机 TPM2.0 的密钥 RSA_mig 生成，标签中的 secret 字段由迁移目的主机 TPM2.0 的密钥 RSA _mig 加密，即 vTPM_label.sig=src.RSA_mig.$priv_{sign}$ (vTPM_label)，vTPM_label.secret=dst.RSA_mig. $pub_{encrypt}$(key)。

其中，src、dst 分别代表迁移源主机和迁移目的主机，pub 和 priv 分别表示公钥和私钥。由于 vTPM 标签两种状态之间可以转换，此处给出 vTPM 标签状态的转换函数 exchange(vTPM_label)的伪代码，如算法 1 所示。

算法 1　vTPM 标签状态的转换函数 exchange (vTPM_label)

输入　vTPM_label

输出　succ 或者 fail

if vTPM_label.status==本地状态 then:

if RSA_ local. pub_{verify} (vTPM_label.sig) == succ then：

KEY = RSA_local. $Pub_{decrypt}$ (vTPM_label. secret)

vTPM_label.secret = dst.RSA_mig.$pub_{encrypt}$ (KEY)

vTPM_label.sig = local.RSA_mig.$priv_{sign}$(vTPM _label)

vTPM_label.status = 迁移状态

return succ

else if vTPM_label.status = 迁移状态 then

if src.RSA_mig. pub_{verify}(vTPM_label.sig)== succ then

KEY = local RSA_mig.$priv_{decrypt}$ (vTPM_ label. secret)

vTPM_label.secret= RSA_local.$pub_{encrypt}$ (KEY)

vTPM_label.sig = RSA_local. $priv_{sign}$(vTPM _label)

vTPM_label.status = 本地状态

return succ

return fail

由于一台物理机可以运行多个虚拟机，如果每个虚拟机对应一个 vTPM 实例，便会有多个 vTPM 标签需要管理。为了减小由 vTPM 标签存储与管理带来的复杂性和攻击面，本文方案将 vTPM 标签存储于对应的虚拟机镜像中，在减小管理复杂度的同时也增加了 vTPM 和 VM 之间的关联强度。

QCOW2（QEMU copy on write version 2）[18] 是 QEMU 模拟器官方支持的虚拟机镜像文件格式。一个 QCOW2 文件被划分为固定大小的块，这些固定大小的块被称为 cluster，每个 cluster 的大小必须在 512 B~2 MB 之间，而且必须是 512 B 的整数倍，默认的 cluster 大小为 64 kB。其中，QCOW2 文件的第一个 cluster 用于存储头部信息和头部扩展信息，如图 4 所示。QCOW2 的头部信息占据 72 B，紧随其后的是头部扩展信息。每个头部拓展包含 3 个字段：type、length、data，其中，type 字段和 length 字段各占据 4 B，data 字段的长度由 length 字段指定，type=0x00000000 是头部扩展的结束标志。本文构造一个 QCOW2 的头部扩展{0x12345678, 368, vTPM 标签+padding}，用来存储 vTPM 标签，因为头部扩展是 8 B 对齐的，于是通过填充 0x00 扩展 vTPM 标签为 8 B 的整数倍。

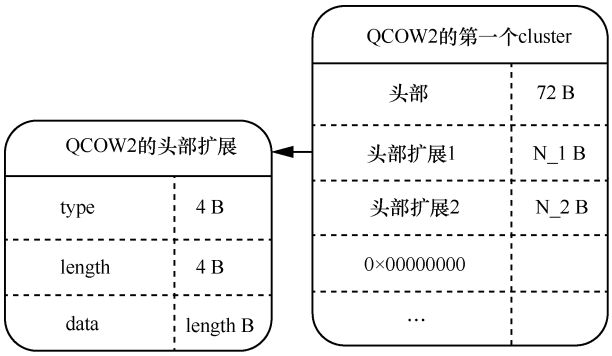


图 4　QCOW2 的第一个 cluster

3.3　本地存储保护方案

3.3.1　总体设计

针对 vTPM 在本地存储上的安全问题，本文设计了 vTPM 私密信息本地存储保护方案。方案包括 3 个组件：本地存储保护代理、vTPM 标签、本地存储保护内核部分，如图 5 所示。

本地存储保护代理位于 QEMU 进程空间内部，针对 vTPM 私密信息的机密性、完整性进行实时的保护，最终将 vTPM 私密信息加密存储在非易失性随机访问存储（NVRAM, non-volatile RAM）文件

中，所需要的 key 在虚拟机启动时向本地存储保护内核部分请求。

vTPM 标签是上文所述的一种数据结构，受控于内核中的 vTPM 标签管理子模块。

本地存储保护内核部分包括 4 个子模块：vTPM 标签管理模块、vTPM 访问控制模块、对应关系检测模块和 TPM 命令管理模块。vTPM 标签管理子模块提供 vTPM 标签的生成、完整性和有效期验证、更新和销毁等功能。vTPM 访问控制子模块根据 vTPM 标签的 qemu_digest 字段检测访问者是否合法，因此，当本地存储保护代理请求 key 时，它是第一个接收访问请求的子模块。对应关系检测子模块根据 vTPM 标签中的“UUID”字段检测访问者是否为该标签对应的虚拟机。TPM 命令管理子模块向 vTPM 标签管理子模块提供 TPM2.0 的各种服务，尤其是密钥 RSA_local 和 RSA_mig 的加解密、签名、验证等功能。

另外，tpm_crb（TPM2.0 command response buffer）[19]是 TPM2.0 在内核中的驱动程序。用户层 TCG 软件栈（TSS, TCG software stack）通过该驱动操作 TPM2.0，使 TPM 命令管理子模块对 tpm_crb 驱动向上层 TSS 提供的 IOCTL 接口进行挂钩，截取 TSS 发送给 tpm_crb 的 TPM 命令，按照一定的规则对部分 TPM 命令进行过滤，防止管理员的误操作，保护本文方案所使用的物理 TPM 密钥的安全。

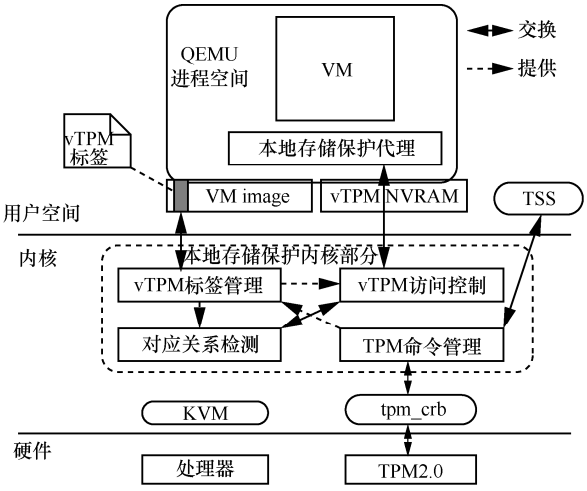


图 5 本地存储保护方案

3.3.2 本地存储保护代理

本地存储保护代理包括两个部分：完整性保护引擎和机密性保护引擎，如图 6 所示。

完整性保护引擎可以有效地识别出虚拟机加载的 vTPM 私密信息是否被恶意篡改。该引擎扩展 vTPM 私密信息的存储结构，新添加一个 hash[]数组，用于存储对 vTPM 私密信息进行散列计算得到的值，然后使用物理 TPM 的密钥 RSA_local 对该数组进行加密，防止其被恶意篡改。

机密性保护引擎将 vTPM 私密信息加密存储在 NVRAM 中，防止信息泄露。因为 vTPM 私密信息是 vTPM 的非易失性信息，所以长度是有限的（本实验平台上，其最大长度为 0x4 000 B），再加上引擎使用 AES 加密算法，因此加解密过程带来的性能消耗较小，实验证明平均一次加密耗时为 0.1 ms 左右。

在虚拟机整个生命周期内，vTPM 对 NVRAM 的读写是有规律的。虚拟机启动时初始化 vTPM，此时会读取 NVRAM 一次，vTPM 私密信息成功加载后，vTPM 便不再读取 NVRAM，然而当 vTPM 执行 TPM 命令导致加载的 vTPM 私密信息发生变化时，vTPM 私密信息便会被覆盖式地写入 NVRAM。结合对以上两种引擎的分析可知，在虚拟机运行中，频繁的操作是添加 hash 和加密，如图 6 所示，对于系统的性能影响较小。

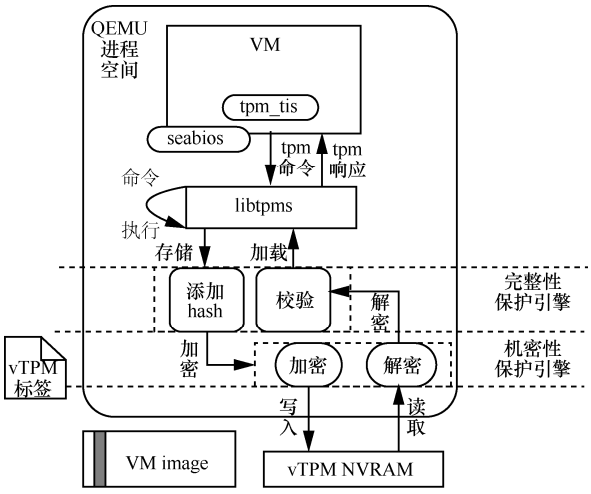


图 6 本地存储保护代理

3.3.3 vTPM 访问控制

vTPM 访问控制子模块对访问者身份进行验证，如图 7 所示。假设访问者为进程 V，则主要步骤为：1) 获取 V 的启动参数 args，按照 QEMU 的参数格式解析 V.args，找到镜像文件路径 image_path；2) 获取 V 打开的文件列表 open_f_list，检查 image_path 是否在 open_f_list 中，通过检查则表示找到了镜像文件 image_path；3) 调用 vTPM 标

签管理子模块的功能，对 image_path 中的 vTPM 标签进行完整性和有效期验证，然后提取 qemu_digest 字段；4) 度量 V 的代码段得到 V.hash；5) 对比 V.hash 和 vTPM_label.qemu_digest，验证通过则交给子模块对应关系检测，否则返回错误值。

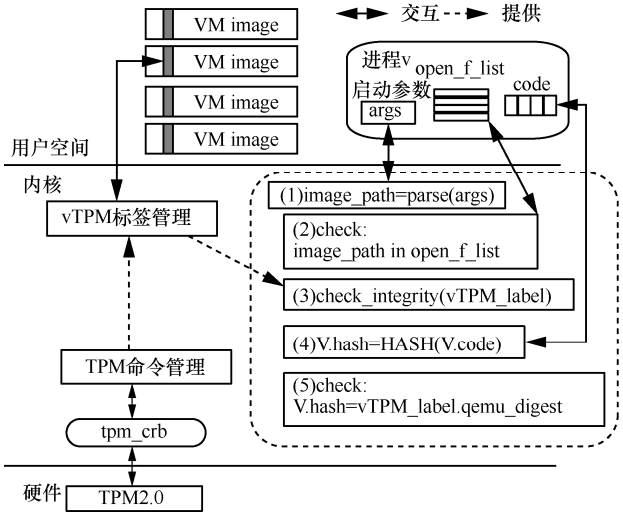


图 7　vTPM 访问控制

3.3.4　vTPM 和虚拟机的一一对应关系

对应关系检测子模块验证 vTPM 和虚拟机的一一对应关系，如图 8 所示，该检测的前提是访问者通过了 vTPM 访问控制子模块的验证。假设访问者为 V，则主要步骤为：1) 获取 V 的启动参数 args，按照 QEMU 的参数格式解析 V.args，找到虚拟机唯一标识符——UUID；2) 调用 vTPM 标签管理子模块的功能，对镜像文件中的 vTPM 标签进行完整性和有效期验证，然后提取 uuid_hash 字段；3) 对比 hash(UUID)和 vTPM_label.uuid_hash，验证通过则认为对应关系正确，否则返回错误值。

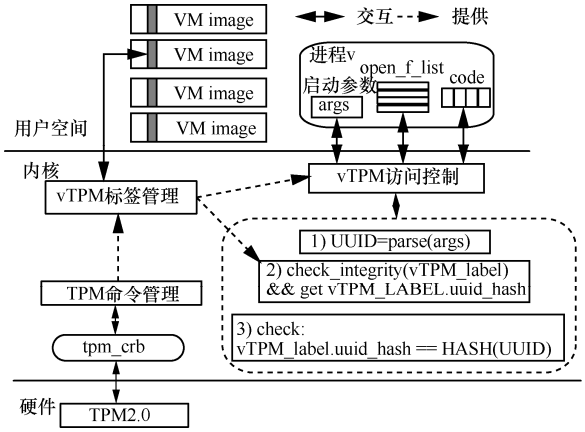


图 8　对应关系检测

3.4　动态迁移保护方案

3.4.1　迁移流程

动态迁移是虚拟化平台的一项基本功能。为了保证动态迁移不会破坏 vTPM 私密信息和易失性信息的安全需求，即机密性、完整性、vTPM 和虚拟机的一一对应关系，本文利用 vTPM 标签的状态信息设计了动态迁移保护方案，如图 9 所示。相比本地存储保护方案，本方案新增了动态迁移保护代理子模块和迁移管理子模块。

迁移管理子模块与本地存储保护内核部分进

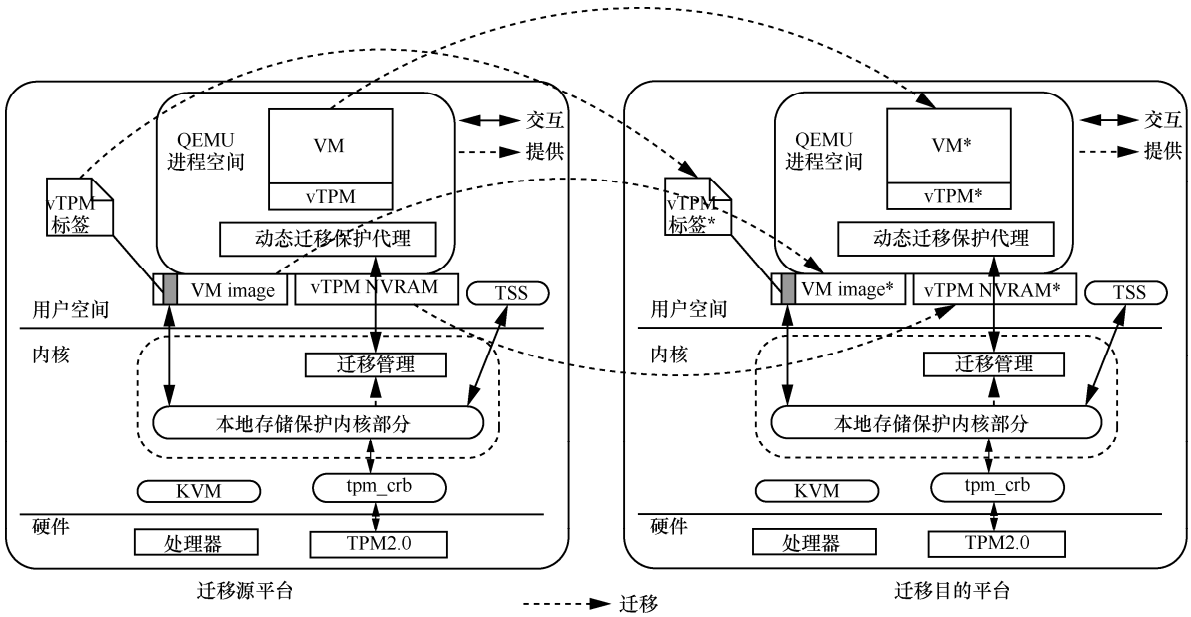


图 9　vTPM 动态迁移框架

行交互，利用算法 1 转换 vTPM 标签的状态。

动态迁移保护代理子模块为虚拟机动态迁移增加了 3 个新的阶段，分别负责安全迁移 vTPM 的易失性信息、私密信息（NVRAM）和 vTPM 标签。1) 同步 vTPM 设备状态（即 vTPM 易失性信息）阶段，动态迁移保护代理子模块注册 vTPM 设备为可迁移设备，从 QEMU 进程空间中收集 vTPM 的易失性信息，封装到数据结构 vTPM_status 中，作为虚拟机设备状态的一部分进行迁移；2) vTPM 标签拷贝阶段，动态迁移保护代理子模块从虚拟机镜像中收集 vTPM 标签，与迁移管理子模块交互进行 vTPM 标签状态的转换，然后封装 vTPM 标签到数据结构 vTPM_label 中，作为虚拟机镜像的一部分进行迁移；3) NVRAM 拷贝阶段，块设备的迁移由 QEMU 的 init_ blk_migration 函数进行初始化，该函数会检查虚拟机所有的块设备文件，包括 vTPM 的块设备文件 NVRAM，一旦 vTPM 被注册为可迁移设备，NVRAM 便和虚拟机镜像文件采用相同的脏页重传机制进行动态迁移。

3.4.2 迁移协议

利用 vTPM 标签的状态信息，本文设计了安全的 vTPM 动态迁移协议，如图 10 所示。该协议借助于普通虚拟机动态迁移的通道传输 vTPM 相关信息，不改变迁移源平台和迁移目的平台之间的信息

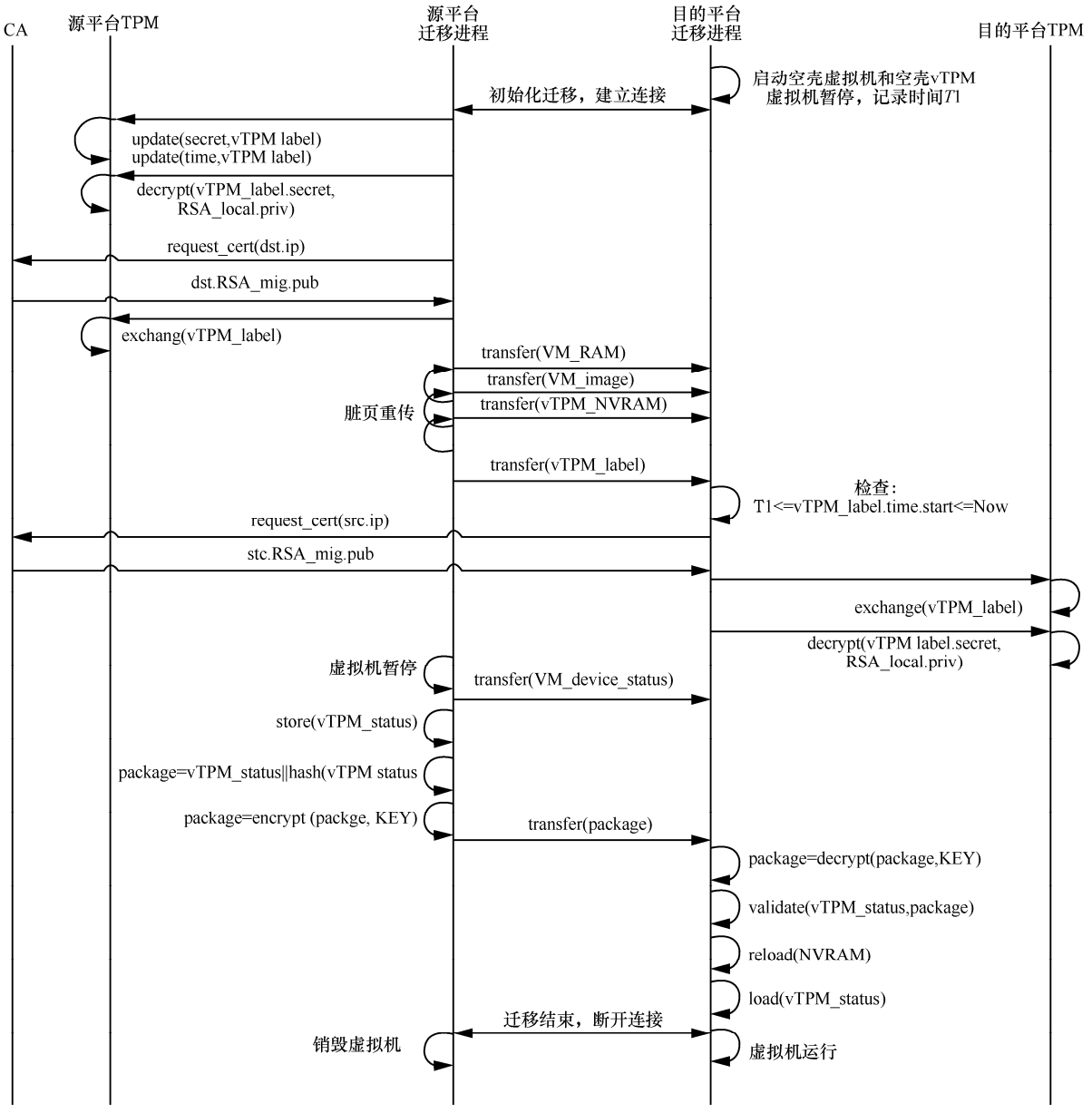


图 10 vTPM 动态迁移协议

交互方式。

本文对协议中涉及的操作原语定义如下。

update(secret, vTPM_label)：更新 vTPM 标签的 secret 字段。

update(time,vTPM_label)：更新 vTPM 标签的 time 字段，更新后的 time.start 为当前时间。

request_cert(ip)：向 CA 查询某个 IP 对应的数字证书，迁移源平台 IP 为 src.ip，迁移目的平台 IP 为 dst.ip。

exchange(vTPM_label)：转换 vTPM 标签的状态，伪代码如算法 1 所示。

transfer(content)：普通虚拟机动态迁移使用的传输函数，传输内容为 content。

encrypt(content，key)：使用 key 加密内容 content。

decrypt(package，key)：使用 key 解密内容 package。

store(vTPM_status)：收集 vTPM 的易失性信息，存储到数据结构 vTPM_status 中。

load(vTPM_status)：从数据结构 vTPM_status 中提取 vTPM 的易失性信息，并加载到 vTPM 设备状态中。

hash(content)：生成 content 的 hash 值。

validate(vTPM_status, package)：验证 package 中 vTPM_status 的 hash 值。

reload(NVRAM)：重新加载 NVRAM。

该协议运行的前提是：1) 迁移源平台和目的平台分别向 CA 申请生成密钥 RSA_mig 对应的数字证书，以便在迁移过程中，迁移源平台和目的平台能够向 CA 查询对方的数字证书，进行双方身份的认证；2) 迁移源平台和目的平台保持时间同步，用于防止旧的 vTPM 标签被重放。

协议运行过程主要步骤如下。

1) 目的平台启动空壳虚拟机和空壳 vTPM，然后暂停虚拟机运行，等待源端虚拟机的迁入，此时记录当前时间为 $T1$。

2) 源平台向目的平台发起迁移请求，双方初始化迁移，建立连接。

3) 源平台更新 vTPM 标签中的 secret 域，保证每次迁移过程中，加密使用的 key 值都是不同的，由于该 key 值被用于加密迁移过程中传输的 vTPM 相关信息，因此该操作可以防止旧的 vTPM 相关信息的重放。

4) 源平台更新 vTPM 标签中的 time 域，保证每次迁移过程中都有 time.start，这样在目的平台对 vTPM 标签的 time.start 进行检查，便可以防止旧的 vTPM 标签的重放。

5) 源平台使用 RSA_local.priv 解密 vTPM 标签的 secret 域，得到 key 值。

6) 源平台向 CA 查询目的平台的数字证书，CA 返回目的平台的数字证书，提取对应目的平台 RSA_mig 的公钥 dst.RSA_mig.pub。

7) 源平台利用 dst.RSA_mig.pub 转换 vTPM 标签为迁移状态。

8) 源平台利用脏页重传的机制，开始传输虚拟机的内存、镜像文件以及 NVRAM，此时 NVRAM 中的信息处于加密状态。

9) 源平台向目的平台传输 vTPM 标签。

10) 目的平台检查 vTPM 标签的 time.start，防止旧的 vTPM 标签的重放。

11) 目的平台向 CA 查询源平台的数字证书，CA 返回源平台的数字证书，提取对应源平台 RSA_mig 的公钥 src.RSA_mig.pub。

12) 目的平台利用 src.RSA_mig.pub 转换 vTPM 标签为本地状态。

13) 目的平台使用 RSA_local.priv 解密 vTPM 标签的 secret 域，得到 key 值。

14) 源平台暂停虚拟机运行，并开始传输虚拟机的设备状态 VM_device_status。

15) 源平台收集 vTPM 的设备状态（即 vTPM 易失性信息），存储在数据结构 vTPM_status 中，计算 vTPM _ status 的 hash 值，然后将 vTPM _ status ||hash (vTPM_status)打包成 package，使用 key 对其进行加密。

16) 源平台传输加密后的 package。

17) 目的平台使用 key 值解密 package，得到 vTPM_status，然后校验 vTPM_status 的 hash 值。

18) 目的平台重新加载 NVRAM，同步源平台虚拟机暂停前对 vTPM 私密信息的所有修改。

19) 目的平台加载 vTPM_status，恢复 vTPM 的设备状态。

20) 迁移结束，源平台销毁虚拟机，目的平台运行虚拟机。

3.4.3　RSA_mig 数字证书的生成

密钥 RSA_mig 是 TPM2.0 的不可迁移密钥，其工作原理和 AIK 密钥类似。因此，本文借助 AIK 数字证书的生成协议[20]来生成密钥 RSA_mig 的数字证书。

密钥 RSA_mig 的数字证书只需要生成一次即

可，因此，在部署整个系统之前便可以完成所有硬件 TPM2.0 芯片的 RSA_mig 数字证书的生成工作。

如图 11 所示，CA 指证书颁发机构，owner 指硬件 TPM2.0 的授权者，RSA_mig 证书的生成协议过程如下。

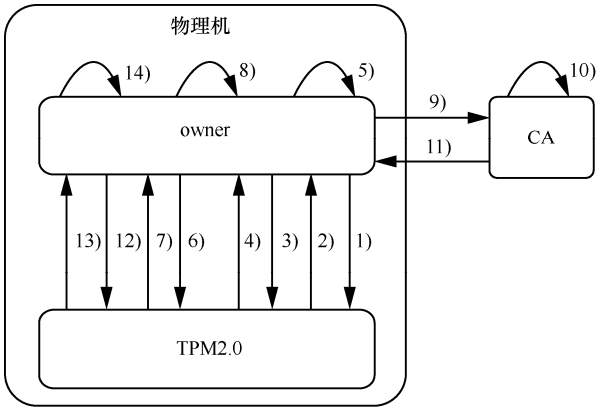


图 11 RSA_mig 数字证书的生成

1) 使用 TPM2_Create 创建 RSA_mig 密钥，使用 TPM2_EvictControl 将 RSA_mig 密钥持久化存储到 TPM2.0 芯片中。

2) 返回成功信息。

3) 使用 TPM2_ReadPublic 提取 RSA_mig 的公钥。

4) TPM2.0 芯片返回 RSA_mig 的公钥。

5) 利用 RSA_mig 的公钥信息，生成证书签名请求文件 RSA_mig.csr。

6) 使用 TPM2_Sign 生成 RSA_mig.csr 的签名值，$S_{RSA\_mig}=sign_{RSA\_mig}(RSA\_mig.csr)$。

7) 返回签名值 S_{RSA_mig}。

8) 填充 RSA_mig.csr 中的签名值 S_{RSA_mig}，收集背书证书 cred，生成对称密钥 K 和随机数 nonce。利用 K 加密需要发送给 CA 的信息，而 K 则由 CA 的公钥 pkCA 进行加密，即 $msg=(enc_K(RSA\_mig.csr,cred,Nonce),enc_{pkCA}(K))$。

9) 发送 msg。

10) 用 CA 私钥解密并验证 RSA_mig.csr 和 cred，签发 RSA_mig 证书，生成对称密钥 K′，使用 K′加密传输的信息，而 K′则由 EK 的公钥 pkEK 进行加密，$resp_{CA}=(enc_{pkEK}(K'), enc_{K'}(nonce, RSA\_mig.cert))$；然后，CA 将密钥 RSA_mig 的数字证书加入自己的证书列表中。

11) 发送 $resp_{CA}$。

12) 执行 TPM2_RSA_Decrypt，使用 EK 私钥解密 $resp_{CA}$ 得到 K′。

13) 返回 K′。

14) Owner 利用 K′解密验证 Nonce，得到证书 RSA_mig.cert。

4 实现与测试

4.1 系统实现

本文基于 KVM 虚拟化平台，通过 qemu2.3.0、模拟 TPM2.0 功能的开源 libtpms、支持 libtpms 的 qemu-patch 实现了基于 libtpms 的全虚拟化 TPM2.0。以表 1 所列组件作为基础的实验环境，设计并实现了 vTPM 私密信息保护方案，同时在控制节点上部署 CA，并对本文方案进行功能测试和性能测试。另外，动态迁移是在没有部署共享存储系统的环境下进行的。

4.2 功能测试

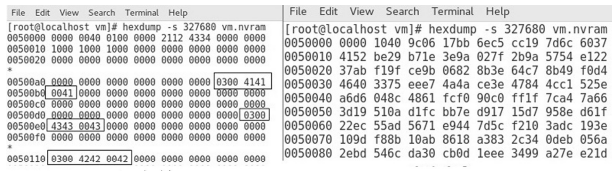
针对 vTPM 私密信息完整性保护引擎、机密性保护引擎、vTPM 和 VM 之间的一一对应关系进行功能测试。

在虚拟机内执行 TPM 命令“tpm2_ takeownership-oAAA-l BBB-e CCC”，如图 12(a)所示。在没有 vTPM 私密信息机密性保护引擎的情况下，由于 vTPM 私密信息的明文存储，通过分析 NVRAM 可以得到 vTPM2.0 的 tpm owner 授权信息。而图 12(b)显示机密性保护引擎可以有效防止 vTPM2.0 私密信息泄露。

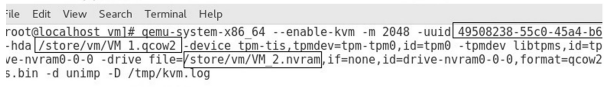
首先建立两个虚拟机 VM_1 和 VM_2 以及对应的 vTPM_1 和 vTPM_2，VM_1 和 VM_2 的 UUID 分别是 UUID_1 和 UUID_2。然后，分别使用“VM_1 的镜像、vTPM_2、UUID_1”“VM_1 的镜像、vTPM_1、UUID_2”“VM_2 的镜像、vTPM_1、UUID_1”这 3 种组合来启动虚拟机，结果如图 12(c)~图 12(e)所示。实验结果证明该方案可以有效保证 vTPM、VM 之间的一一对应关系，同时图 12(c)也证明了完整性保护引擎可以有效识别出 vTPM 私密信息的不同。

表 1 系统配置

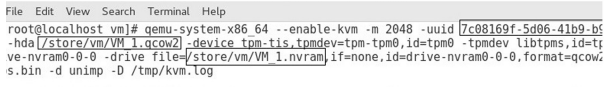
服务器	个数	CPU	内存/GB	磁盘空间/TB	TPM
控制节点服务器	1	Intel(R)Core(TM) i5-6400	8	1	无
计算节点服务器	2	Intel(R)Core(TM) i5-6400	8	1	TPM2.0



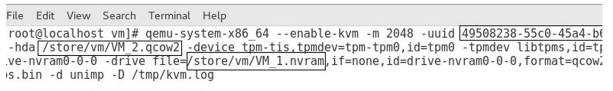
(a) 加密前　　(b) 加密后



(c) VM_1 的镜像、vTPM_2、UUID_1



(d) VM_1 的镜像、vTPM_1、UUID_2



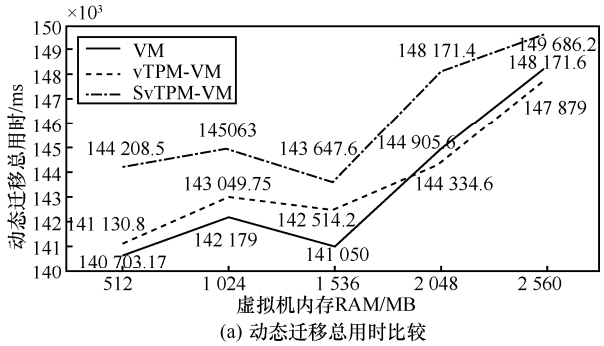
(e) VM_2 的镜像、vTPM_2、UUID_1

图 12　功能测试

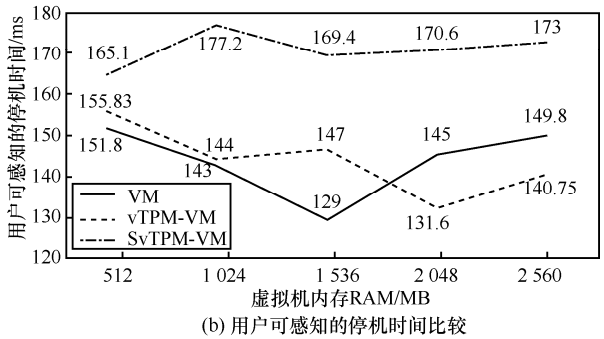
4.3　性能测试

针对虚拟机的动态迁移进行性能测试，实验环境建立在没有配置共享存储的情况下。为了减小虚拟机镜像传输带来的影响，性能测试使用大小固定为 4 GB 的虚拟机镜像。测试的虚拟机有 3 种：VM、vTPM-VM、添加 vTPM 私密信息保护方案的 vTPM-VM（即图 13 中 SvTPM-VM），分别记录动态迁移的总用时、用户可感知的停机时间的变化。本文设置 5 组不同的虚拟机内存，在每组设置中对以上 3 种虚拟机分别执行 10 次动态迁移，计算每种虚拟机动态迁移用时的平均值，结果如图 13 所示。

本文提出的 vTPM 私密信息保护方案在 VM 动态迁移过程中添加了 3 个新阶段：NVRAM 拷贝、vTPM 标签拷贝、同步 vTPM 设备状态（即 vTPM 易失性信息）。其中，NVRAM 拷贝和 vTPM 标签拷贝这两个阶段处于虚拟机运行阶段，不会对用户可感知的停机时间造成影响，但是会增加动态迁移的总体用时。由图 13(a)可知，SvTPM-VM 比 vTPM-VM 平均增加耗时 2 373.67 ms，平均增长率为 1.65%，该差异是新增 vTPM 标签拷贝阶段的平均耗时；vTPM-VM 比 VM 平均增加耗时 379.80 ms，平均增长率为 0.27%，两者总体用时非常接近，该差异是新增 NVRAM 拷贝阶段的平均耗时。另外，同步 vTPM 设备状态（即 vTPM 易失性信息）阶段会增加用户可感知的停机时间。由图 13(b)可知，SvTPM-VM 比 vTPM-VM 平均增加耗时 27.22 ms，平均增长率为 19.36%，增加的耗时不会影响正常虚拟机用户使用；vTPM-VM 比 VM 平均增加耗时 0.116 ms，平均增加率为 0.41%，两者非常接近。



(a) 动态迁移总用时比较



(b) 用户可感知的停机时间比较

图 13　性能测试

5　结束语

本文设计了一种数据结构——vTPM 标签，借助于物理 TPM2.0 的密码算法，提出一种新的 vTPM 私密信息保护方案。该方案分为本地保护和动态迁移保护两个部分。本地保护方案使用 vTPM 标签的各个字段加密存储 vTPM 的私密信息、校验其完整性、绑定虚拟机和 vTPM 的对应关系，有效实现了 3.1 节提出的设计原则 1)和设计原则 2)；动态迁移保护方案使用 vTPM 标签的状态属性，在不影响正常虚拟机动态迁移过程的情况下实现了 vTPM 的安全动态迁移，实现了 3.1 节提出的设计原则 3)。通过实验证明了所提方案能够有效保护 vTPM 私密信息在本地存储和动态迁移过程中的安全属性，同时不会带来较大的性能消耗。

参考文献：

[1] BERGER S, CÁCERES R, GOLDMAN K A, et al. vTPM: virtualizing the trusted platform module[C]//The 15th Conference on USENIX

Security Symposium. 2006: 305-320.

[2] 严飞, 龚玉凤, 于钊. 基于硬件事务内存的 vTPM 安全保护方法: CN105678173[P]. 2016-06-15.
YAN F, GONG Y F, YU Z. vTPM security protection method based on hardware transaction memory[P]. CN105678173A, 2016-06-15.

[3] 严飞, 于钊, 张立强, 等. vTSE:一种基于 SGX 的 vTPM 安全增强方案[J]. 工程科学与技术, 2017, 49(2): 133-139.
YAN F, YU Z, ZHANG L Q, et al. vTSE: a solution of SGX-based vTPM secure enhancement[J]. Advanced Engineering Sciences, 2017, 49(2): 133-139.

[4] SHI Y, ZHAO B, YU Z, et al. A Security-improved scheme for virtual TPM based on KVM[J]. Wuhan University Journal of Natural Sciences, 2015, 20(6): 505-511.

[5] JIN X, CHEN X S, ZHAO C, et al. Trusted attestation architecture on an infrastructure-as-a-service[J]. Tsinghua Science and Technology, 2017, 22(5): 469-477.

[6] 黄宇晴, 赵波, 肖钰, 等. 一种基于 KVM 的 vTPM 虚拟机动态迁移方案[J]. 山东大学学报(理学版), 2017, 52(6): 69-75.
HUANG Y Q, ZHAO B, XIAO Y, et al. A vTPM-VM live migration scheme based on KVM[J]. Journal of Shandong University (Natural Science), 2017, 52(6): 69-75.

[7] 石源, 张焕国, 赵波, 等. 基于 SGX 的虚拟机动态迁移安全增强方法[J]. 通信学报, 2017, 38(9): 65-75.
SHI Y, ZHANG H G, ZHAO B, et al. Security-enhanced live migration based on SGX for virtual machine[J]. Journal on Communications, 2017, 38(9): 65-75.

[8] FAN P R, ZHAO B, SHI Y, et al. An improved vTPM-VM live migration protocol[J]. Wuhan University Journal of Natural Sciences, 2015, 20(6): 512-520.

[9] WAN X, ZHANG X F, CHEN L, et al. An improved vTPM migration protocol based trusted channel[C]//International Conference on Systems and Informatics. 2012: 871-875.

[10] DANEV B, MASTI R J, KARAME G O, et al. Enabling secure VM-VTPM migration in private clouds[C]//The 27th Annual Computer Security Applications Conference. 2011: 187-196.

[11] CHALLENER D, YODER K, CATHERMAN R, et al. A practical guide to trusted computing[M]. Beijing: China Machine Press, 2008.

[12] BERGER S, GOLDMAN K, PENDARAKIS D, et al. Scalable attestation: a step toward secure and trusted clouds[C]//IEEE International Conference on Cloud Engineering. 2015: 185-194.

[13] ARTHUR W, CHALLENER D, GOLDMAN K. A practical guide to TPM 2.0: using the trusted platform module in the new age of security[M]. Berkeley: Apress, 2015.

[14] CUCURULL J, GUASCH S. Virtual TPM for a secure cloud: fallacy or reality?[C]//The 13th Spanish Meeting on Cryptology and Information Security. Alicante. 2014: 197-202.

[15] 杨永娇, 严飞, 毛军鹏, 等. Ng-vTPM: 新一代 TPM 虚拟化框架设计[J]. 武汉大学学报(理学版), 2015, 61(2): 103-111.
YANG Y J, YAN F, MAO J P, et al. Ng-vTPM: a next generation virtualized TPM architecture[J]. Wuhan University Journal of Natural Sciences, 2015, 61(2): 103-111.

[16] 王丽娜, 高汉军, 余荣威. 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, 32(9): 1-8.
WANG L N, GAO H J, YU R W. Research of constructing trusted virtual execution environment based on trust extension[J]. Journal on Communications, 2011, 32(9): 1-8.

[17] HOHMUTH M, PETER M, H¨ARTIG H, et al. Reducing TCB size by using untrusted components—small kernels versus virtual-machine monitors[C]//The 11th workshop on ACM SIGOPS European Workshop. 2004: 22.

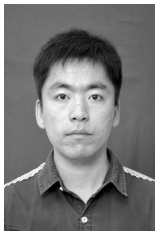
[18] RAZAVI K, KIELMANN T. Scalable virtual machine deployment using VM image caches[C]//The International Conference on High Performance Computing, Networking, Storage and Analysis. 2013: 65.

[19] MAYES K, MARKANTONAKIS K. Smart cards, tokens, security and applications[M]. New York: Springer Publishing, 2010.

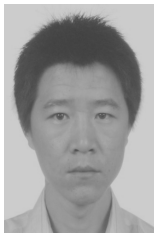
[20] TCG Infrastructure Working Group. A CMC profile for AIK certificate enrollment[M]. Beaverton, Oregon: TCG, 2011.

[作者简介]

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。



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。



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。