doi:10.11959/j.issn.1000-436x.2016025

基于 TPM 联盟的可信云平台管理模型

田俊峰，常方舒
（河北大学网络技术研究所，河北 保定 071002）

摘 要：以可信计算技术为基础，针对可信云平台构建过程中可信节点动态管理存在的性能瓶颈问题，提出了基于 TPM 联盟的可信云平台体系结构及管理模型。针对 TPM 自身能力的局限性，提出了宏 TPM 和根 TPM 的概念。针对可信云中节点管理时间开销大的问题，引入时间树的概念组织 TPM 联盟，利用 TPM 和认证加密技术解决数据在 TPM 联盟内节点间的可信传输问题，提出了一种基于时间树的 TPM 联盟管理策略，包括节点配置协议、注册协议、注销协议、实时监控协议、网络管理修复协议和节点更新协议，阐述了时间树的生成算法，分析了建立可信节点管理网络的时间开销和节点状态监控的有效性。最后，通过仿真实验说明了模型具有较好的性能和有效性。
关键词：TPM 联盟；云计算；可信云平台；时间树
中图分类号：TP393 **文献标识码**：A

Trusted cloud platform management model based on TPM alliance

TIAN Jun-feng, CHANG Fang-shu
(Institute of Network Technology, Hebei University, Baoding 071002, China)

Abstract: On the basis of trusted computing technology, trusted cloud platform architecture and management model based on the TPM alliance was proposed to solve the performance bottleneck of dynamic management of trusted nodes in the building process of trusted cloud platform. Macro TPM was proposed to solve the capability limitation of TPM, the concept of time-based tree was introduced to organize TPM alliance, addressing the problem of high time cost of nodes management in trusted cloud. It used TPM and authentication encryption technology to solve the trusted transmission problem of data among nodes in TPM alliance, and a management strategy of time-based tree TPM alliance was proposed, including node configuration protocol, node registration protocol, node logout protocol, node state real-time monitor protocol, trusted nodes management network repair protocol, node update protocol. That explains the production algorithm of time-based tree, analyses the effectiveness of the time cost of building trusted node management network and monitoring of node state. The simulation result indicates that the model is efficient, and the time cost in trusted node management can be reduced.
Key words: TPM alliance, cloud computing, trusted cloud platform, time-based tree

1 引言

云计算通过将超大规模的计算和存储资源整合起来，以按需服务的方式对外提供资源。越来越多的企业开始采用云计算，据 Gartner 预计，2016 年全球数字内容的$\frac{1}{3}$将存储在云上，Forrester 预计 2020 年全球云计算市场规模将达到 2 140 亿美元。然而，云计算的发展仍然面临着很多挑战，其中云安全已成为其发展的最大障碍。Gartner 的调查结果显示，多于 70%的受访企业首席技术官表示近期不采用云计算，首要原因在于对数据安全性与隐私保护的忧虑[1]。传统的数据加密只能保护数据存储时安全，而当数据在云中以明文形式操作的时候，仍然面临着云服务提供

收稿日期：2015-03-18；**修回日期**：2015-06-23
基金项目：国家自然科学基金资助项目（No.61170254）；河北省自然科学基金资助项目（No.F2014201165）；河北省高等学校科学技术研究重点基金资助项目（No.ZH2015088）
Foundation Items: The National Natural Science Foundation of China (No.61170254), The Natural Science Foundation of Hebei Province (No.F2014201165), The University of Hebei Province Science and Technology Research Program (No.ZH2015088)

商内部员工恶意或者失误造成的泄露，这就需要一种机制来保证云平台中计算的机密性和完整性。

基于可信计算技术的可信云平台能够在用户应用加载之前验证当前平台的状态是否可信，从而保证数据的机密性和完整性。现在越来越多的基于用户验证或基于第三方验证的可信云服务方案被提出来，但随之而来的问题是，云服务提供商如何保证自己的云平台是可信的。如果云服务提供商做不到这一点，那么用户在使用上述方案时，云服务很可能通不过验证，导致双方无法进行服务交易，这对双方都是不利的。因此本文从云平台建设的视角来设计一个状态可实时监控的可信云平台。

2 相关工作

Garfinkel 等[2]提出 Terra 框架通过可信虚拟机监视器构造闭盒虚拟机环境给应用程序提供隐私和完整性保护。Shakeel 等[3]提出的自服务云计算模型通过引入新的权限模型来解决用户对自己虚拟机的安全隐私保护和灵活控制问题。Jonathan 等[4]提出的 TrustVisor 框架实现了具有特殊功能的虚拟机监视器，当安全敏感的代码运行时能够与商业操作系统和应用隔离。Hidekazu 等[5]提出的 VMCrypt 系统通过双视图内存，即正常视图和加密视图，用户虚拟机使用正常视图，管理域使用加密视图来防止用户虚拟机中的内存信息泄露给管理域。Sven 等[6]提出的加密即服务框架通过将用户虚拟机中的加密操作和证书分离到用户控制的加密域中来实施用户自己的安全解决方案。Chen 等[7]提出的 cTPM 通过 TPM 和云平台共享一个根密钥来解决 TPM 资源受限的问题。上述方案都是在单个主机上构建可信运行环境，然而云环境下存在大量的主机，这就需要可信主机之间进行协作。

吴吉义等[8]指出解决云安全问题的重要方案之一是将云计算与可信计算结合。Joshua 等[9]提出的云验证服务框架通过部署统一的云验证服务来解决云用户验证云平台完整性的问题。该方案的不足是平台的完整性度量信息不能够实时更新。Lucas 等[10]提出动态完整性度量和验证框架 DynIMA 以解决这个问题。Berger Stefan 等[11]提出可信虚拟数据中心 TVDc，该技术主要用于保证云环境中用户虚拟机的隔离。随后 Berger 等[12]对原始的 TVDc 进行功能增强，利用云资源的安全标签来控制用户对云存储的访问。Andy 等[13]提出的 Jobber 是一个高度自治的多租户网络安全框架，用于适应云数据中心的动态特性和优化租户之间的通信。Wu 等[14]提出访问控制即服务框架 AcaaS，该框架的核心思想是将用户的访问控制策略的管理外包给服务提供商。刘川意等[15]提出的 TCEE 通过给用户虚拟机绑定 vTPM，利用 TCG 软件栈 TSS 和 OpenPTS 将用户虚拟机环境信息提交给可信第三方进行完整性验证，来保证用户应用程序运行环境的完整性。Li 等[16]提出的多租户可信计算环境模型 MTCEM 将云计算服务的安全职责进行分离，云提供商负责基础设施的可信，用户负责虚拟机实例和应用程序的可信。Zhang 等[17]提出的 CloudVisor 框架利用嵌套虚拟化技术来增强云中多租户环境下用户虚拟机的安全保护。上述方案虽然对云环境下的可信问题进行了研究，但没有考虑云节点管理这个问题。

Santos 等[18]提出可信云计算平台 TCCP，该平台通过可信协调者管理云中的所有可信节点，当节点规模很大时，管理这些节点的时间开销是不能忍受的。随后 Santos 等[19]改进的 Excalibur 通过采用基于属性的密文策略加密算法 CPABE 和引入多个中心化的监视器来改进管理可信节点所造成的性能瓶颈。针对可信云平台构建过程中可信节点管理存在的性能瓶颈造成的扩展性差的问题，Santos 等改进的 Excalibur 是目前所提出的方法当中较好的。

王丽娜等[20]提出的基于可信平台模块的密钥使用次数管理方法能够安全有效地存储和保护密钥，从而保护云存储中数据的机密性并控制密钥的使用次数。田俊峰等[21]提出一种可信的云存储控制模型 TCMCS，该模型用于解决云存储服务的安全问题。张焕国等[22]对云环境下的可信网络连接进行研究，提出了统一的网络访问控制架构。王娟等[23]提出的 POSTER 利用可信网络连接为云计算提供端到端的可信保护。周振吉等[24]针对云环境下虚拟机可信度量方法存在的并发性问题，提出一种树形可信度量模型 TSTM 以提高度量模型的可扩展性。上述工作针对云平台中秘钥管理、云存储、网络访问控制和虚拟机完整性度量等问题进行了研究。

可信云平台的建设离不开对大规模可信云节点的自动化管理和实时监控。虽然 Santos 等针对节点管理这个问题提出了解决方案，但没有彻底解决可信节点管理所造成的性能瓶颈，影响可信云平台的扩展性。本文首先提出了基于 TPM 联盟的可信云平台体系结构，然后给出了基于时间树的状态可

实时监控的可信云平台管理方案。

3　基于 TPM 联盟的可信云平台体系结构

TPM（trusted platform module）联盟是 2 个或 2 个以上的独立的可信平台模块通过某种结构或协议组织起来的分布式可信基。TPM 联盟整体上作为可信云平台（TCP, trusted cloud platform）的信任根，其中 TPM 作为可信云平台中可信服务器（server）的信任根。基于 TPM 联盟的可信云平台体系结构如图 1 所示。

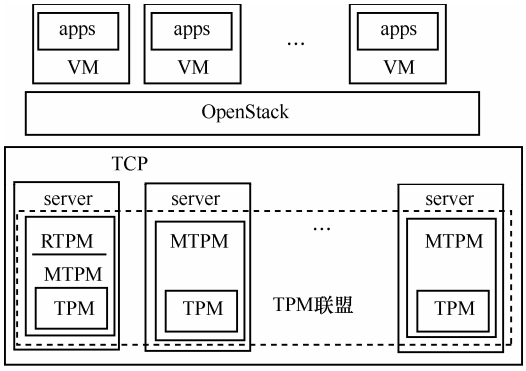


图 1　可信云平台体系结构

云资源以虚拟机（VM, virtual machine）的形式提供给云用户，OpenStack 是云用户访问云资源的接口。可信云平台由 2 类可信服务器构成，一类是管理节点，一类是服务节点。云管理员通过管理节点了解每个服务节点的可信状态。MTPM（macro TPM）是基于 TPM 的可信证据收集和验证模块，主要负责其所在服务节点的可信证据收集和验证其他服务节点的可信性，同时解决 TPM 计算资源不足的问题。RTPM（root TPM）是 TPM 联盟的可信根，主要负责维护和管理整个 TPM 联盟，TPM 联盟通过 RTPM 管理所有的 MTPM。

MTPM 逻辑结构如图 2 所示，系统中各组件功能如下。

1) 可信证据收集组件：主要包含 2 部分功能，一是系统启动时的静态完整性度量（包括 BIOS、bootloader、OS、可执行文件、配置文件）；二是系统运行时的动态完整性度量（包括虚拟机、进程的可执行文件、环境变量），结果存入证据数据库。

2) 节点状态收集组件：接收下级 MTPM 中节点状态收集组件传递的节点状态集合，并将验证组件产生的节点状态插入到节点状态集合中，将修改后的节点状态集合传递给上级 MTPM 中的节点状态收集组件。

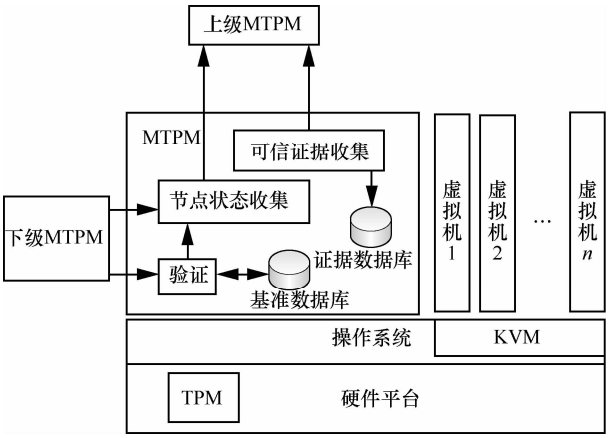


图 2　MTPM 逻辑结构

3) 验证组件：利用基准数据库中的基准值验证下级 MTPM 中可信证据收集组件传递的证据，并将结果传递到节点状态收集组件。

4) 证据数据库：是服务节点中的内存数据库文件，MTPM 初始化时由可信证据收集组件创建并管理，证据数据库的完整性由 TPM 保证，主要用来存放被度量对象的完整性度量值。

5) 基准数据库：是服务节点中的磁盘数据库文件，MTPM 初始化时由验证组件创建并管理，基准数据库的完整性由 TPM 保证。

RTPM 逻辑结构如图 3 所示，系统中各组件功能如下。

1) 节点状态收集组件：接收下级 MTPM 中节点状态收集组件传递的节点状态集合，并将验证组件产生的节点状态插入到节点状态集合中，将修改后的节点状态集合存入节点状态数据库。

2) 验证组件：利用基准数据库中的基准值验证下级 MTPM 中可信证据收集组件传递的证据，并将结果传递到节点状态收集组件。

3) 警报器：将不可信节点的状态信息报告给云管理员。

4) 节点管理组件：TPM 联盟初始化时，云管理员利用该组件创建并管理云节点数据库、可信软件仓库、节点配置数据库。

5) 基准数据库：是管理节点中的磁盘数据库文件，MTPM 初始化时由验证组件创建并管理，基准数据库的完整性由 TPM 保证，基准数据库中的内容从 RTPM 中的节点配置数据库获得，用于验证被验证服务节点的完整性。

6) 节点状态数据库：是管理节点中的内存数据库文件，TPM 联盟初始化时，由云管理员创建，用

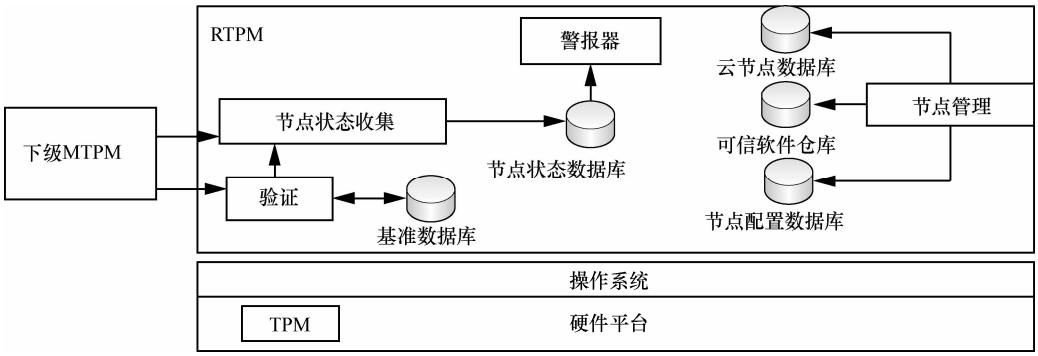


图 3 RTPM 逻辑结构

于实时记录 TPM 联盟中服务节点的状态，数据库大小只有在服务节点规模发生变化时才发生变化。

7) 云节点数据库：是管理节点中的磁盘数据库文件，存储云中所有节点（包括服务节点、管理节点、未提供服务的节点)的元数据,元数据包括节点的UUID（节点唯一标识)、公钥证书、IP 和节点配置类型。

8) 可信软件仓库：是管理节点中的磁盘数据库文件，存储软件的文件路径和签名信息，为 TPM 联盟中安装的软件提供可信源，TPM 联盟中运行的每一个可执行文件都是可验证的。

9) 节点配置数据库：是管理节点中的磁盘数据库文件，云管理员确定 TPM 联盟中的节点配置类型，每一种配置类型的详细信息包括服务器硬件信息、操作系统信息、运行的应用信息（例如软件版本、配置参数等），每一种配置类型都会对应一个指纹。

在可信云平台中，管理节点的 RTPM 负责维护所有云节点的状态信息。为保证可靠性，管理节点一般由 2 台服务器构成，组成热双工方式。MTPM 周期性对所在节点进行完整性度量，度量信息通过时间树最终汇集到 RTPM 中。

4 基于时间树的云平台可信节点管理策略

Santos 等提出的 TCCP 通过引入可信协调者（TC）来管理云平台内部的可信计算节点。TC 负责处理云平台内部可信节点的加入、退出和维护。由于 TPM 的低效，TC 执行一次可信节点加入操作大约花费 1 s 甚至更长的时间，当大量的节点同时加入时，时间上的开销是不能忍受的。同时 TC 负责管理所有的节点，当节点规模很大时，TC 容易成为瓶颈。Excalibur 虽然在监视节点与被管理节点之间采用基于属性的密文策略加密框架（CPABE, ciphertext policy attribute-based encryption），减少了大量节点同时加入可信云平台的时间成本，但是当涉及到节点的批量注销和维护等管理操作时，监视节点仍然会成为系统瓶颈。针对上述问题，本文提出了基于时间树的云平台可信节点管理策略。

定义 1 时间树 T 是一个二元组$<V, E>$，其中，V 是 $n(n\geqslant 0)$个节点的有限集合，$V=\{v_i \mid 0\leqslant i\leqslant n-1\}$，$E$ 是节点间边的集合，$E=\{<v_i, v_j> \mid v_i, v_j \in V, i\neq j\}$。

另外，$v_i=<I,t>$，其中，i 是编号，$i\in[0,n-1]$，t 是时间属性，$t\in[0, v_i]$。定义时间属性为节点加入时间树的时刻，若 v_i 与 v_j 同一时刻加入时间树，则有 $t_i=t_j$；若 v_i 先于 v_j 加入时间树，则有 $t_i<t_j$。i 和 t 满足如下关系

$$t=\begin{cases}0, i=0\\ \lfloor \mathrm{lb}\, i \rfloor+1, i\neq 0\end{cases} \tag{1}$$

$f(i)$被定义为

$$f(i)=\begin{cases}0, i=1\\ \dfrac{i}{2}, i \bmod 2=0\\ f\left(\dfrac{i-1}{2}\right), i \bmod 2\neq 0, i\neq 1\end{cases} \tag{2}$$

当且仅当 i 和 j 满足 $j=f(i)$时，$<v_i, v_j>\in E$，此时称 v_i 是 v_j 的后继节点，v_j 是 v_i 的前驱节点，v_j 的所有后继节点为兄弟节点。

当 $V=\boldsymbol{\Phi}$ 时，称 T 为空时间树；当 $V\neq\boldsymbol{\Phi}$ 时，V 中的节点随机确定编号，并依式(1)确定节点的时间属性值，此时依照各节点的编号和时间属性唯一构建时间树。

若时间树 T 非空，称<0, 0>为根节点。若 $i\geqslant\dfrac{n}{2}$，称$<i, t_i>$为叶节点，否则$<i,\ t_i>$称为分支节点。

依时间树定义中集合 V 和集合 E 必须满足的条

件，形成的时间树有 2 种表示形式。第 1 种表示形式如图 4(a)所示，时间树中的节点依时间属性值 t 分层。第 2 种表示形式如图 4(b)所示，时间树中的节点依兄弟关系进行分层。

4.1　可信云平台节点管理

TPM 联盟中的节点依时间树模型形成的可信节点管理网络如图 5 所示。

RTPM 表示管理节点，MTPM 表示服务节点。RTPM 的编号为 0，MTPM 的编号按照时间树的定义由云管理员在 TPM 联盟初始化创建节点状态数据库时指定。节点的时间属性值表示节点在第几个认证周期加入到可信节点管理网络中。认证周期表示 2 个节点之间进行远程证明所花费的时间。

在描述具体协议之前，本文做如下约定。

$UUID_i$：表示节点 i 的 UUID，用于 TPM 联盟和云节点数据库中永久标识一个节点。

ID_i：表示节点 i 的编号，用于可信节点管理网络和节点状态数据库中标识一个节点。

$MTPM_i$：表示节点编号为 ID_i 的服务节点。

$M / RTPM_{i\to j}$：表示节点 i 的前驱节点 j。

IP_i：表示节点 i 的 IP 地址。

PK_i、SK_i：分别表示节点 i 的公钥和私钥。

KEY_{ij}：表示节点 i 和 j 共享的对称密钥。

ML_i：表示节点 i 的完整性度量值，由可信证据收集组件产生。

CON_i：表示节点 i 的配置类型，对应于节点配置数据库中的某种配置类型。

$CHECKVALUE_i$：表示节点 i 的基准值，是节点配置数据库中与 CON_i 类型相对应的指纹。

$\{M\}K$：表示用密钥 K 对消息 M 进行加密。

4.2　节点配置协议

该协议使 MTPM 从 RTPM 获取自己的编号、前驱节点的 IP 和公钥证书用于建立可信节点管理网络。该协议在 TPM 联盟启动后（包括重新启动）执行。

图 6 描述了 MTPM 和 RTPM 之间的信息交换过程，具体流程如下。

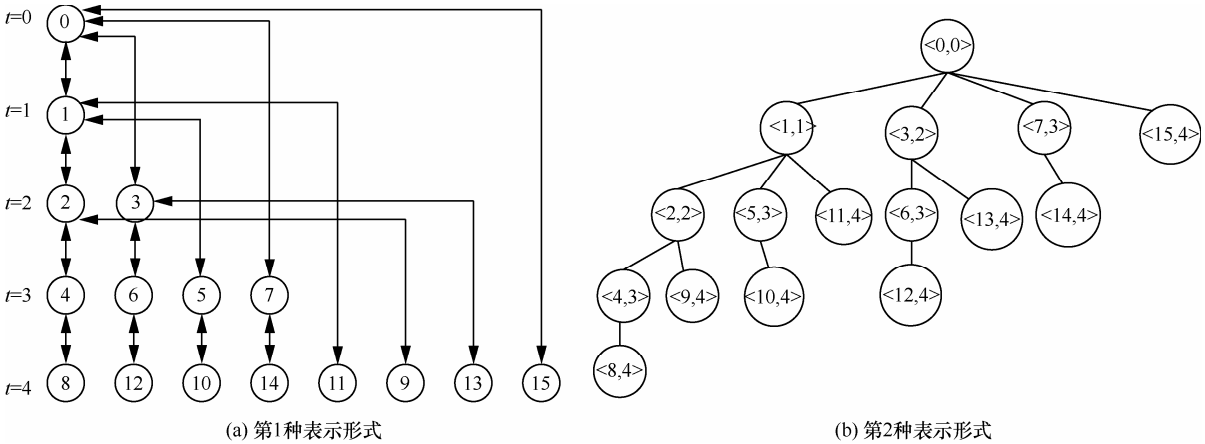


图 4　n=16 时的时间树

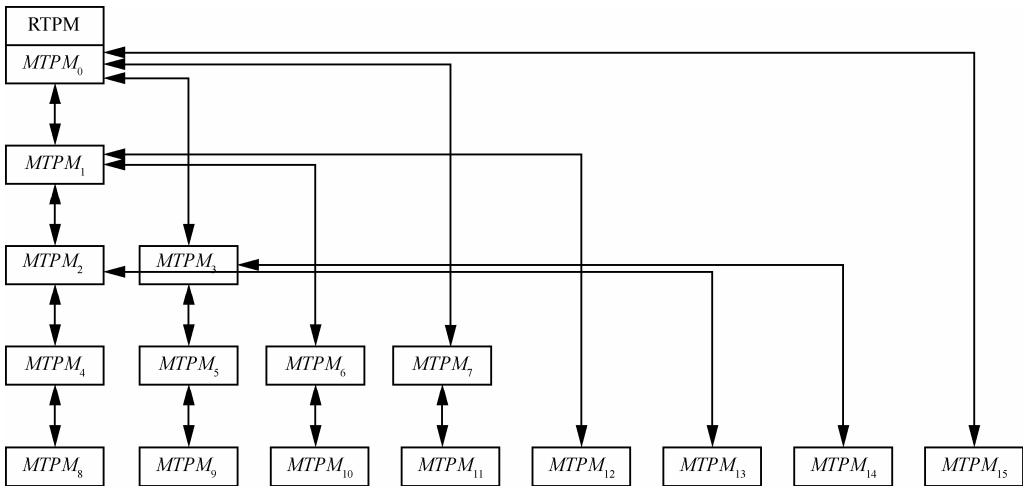


图 5　可信节点管理网络

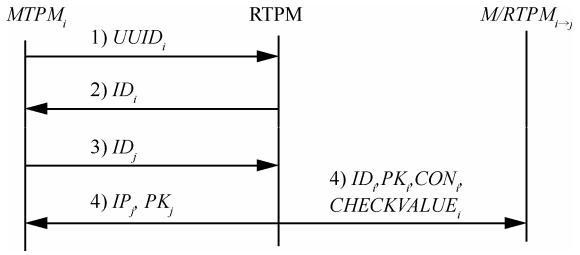


图 6 节点配置协议

1) $MTPM_i$ 向 RTPM 发送编号请求信息 $UUID_i$。

2) RTPM 通过 $UUID_i$ 查询节点状态数据库得到 ID_i，同时发送响应信息 ID_i。

3) $MTPM_i$ 将 ID_i 代入式(2)中，计算出前驱节点编号 ID_j，同时发送前驱节点信息请求 ID_j。

4) RTPM 通过 ID_j 查询节点状态数据库得到 $UUID_j$，再通过 $UUID_j$ 查询云节点数据库得到 IP_j 和 PK_j，同时向 $MTPM_i$ 发送响应信息：IP_j 和 PK_j；RTPM 通过 $UUID_i$ 查询云节点数据库得到 PK_i、CON_i 和 $CHECKVALUE_i$，同时向 $M/RTPM_{i\to j}$ 发送信息 ID_i、PK_i、CON_i 和 $CHECKVALUE_i$。

$MTPM_i$ 将 ID_i、IP_j 和 PK_j 存储到 TPM 的非易失性存储器中。$M/RTPM_{i\to j}$ 将 ID_i 和 PK_i 存储到 TPM 的非易失性存储器中，将 CON_i 和 $CHECKVALUE_i$ 存储到基准数据库。

4.3 节点注册协议

该协议使 RTPM 验证所有 MTPM 启动后的完整性。MTPM 通过验证，才能成为 TPM 联盟中的一员来提供服务。该协议与基于中心的节点管理策略不同的是，该协议的执行不一定发生在 MTPM 和 RTPM 之间，而是发生在 MTPM 和前驱节点之间，这样保证了大量的节点能够同时注册。该协议同时实现了节点间的可信传输。可信节点管理网络的建立是通过各 MTPM 执行节点注册协议完成的，其中节点注册协议利用了 TPM 的随机数生成、密钥生成、签名和验证签名等基本功能。

MTPM 每次重启之后都要主动执行节点注册协议，MTPM 的状态通过可信节点管理网络传输给 RTPM。RTPM 利用可信节点管理网络实时监控所有 MTPM 的状态。

图 7 描述了 $MTPM_i$ 与 $M/RTPM_{i\to j}$ 之间的信息交换过程，具体流程如下。

1) $MTPM_i$ 向 $M/RTPM_{i\to j}$ 发送注册请求信息 ID_i、CON_i 和随机数 n_1。

2) $M/RTPM_{i\to j}$ 向 $MTPM_i$ 发送响应信息：随机数 n_2，签名 SIG_1：$\{n_1,t\}SK_j$，其中 t 是 $M/RTPM_{i\to j}$ 加入可信节点管理网络的标识。

3) $MTPM_i$ 验证 SIG_1，若通过验证发送响应信息：签名 SIG_2：$\{n_1, n_2, ML_i\}SK_i$。否则，发送挂起信号 SUSPEND，等待下一个认证周期重新注册。

4) $M/RTPM_{i\to j}$ 验证 SIG_2，若验证通过，生成 KEY_{ij}，发送响应信息：$\{KEY_{ij}\}PK_i$，$\{KEYSET, IPSET\}KEY_{ij}$，其中，$KEYSET$ 是 $M/RTPM_{i\to j}$ 与其他节点之间的对称密钥集合，$IPSET$ 是各节点的 IP 集合。同时 $M/RTPM_{i\to j}$ 向其他后继节点发送 KEY_{ij} 和 IP_i。若验证不通过，向前驱节点发送 $MTPM_i$ 的状态不可信警报。

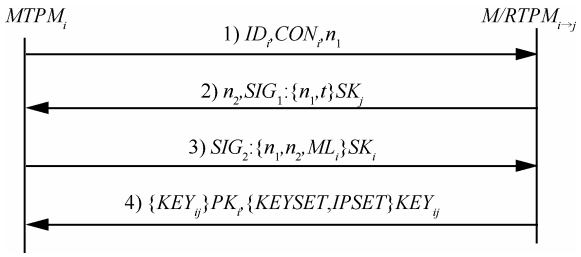


图 7 节点注册协议

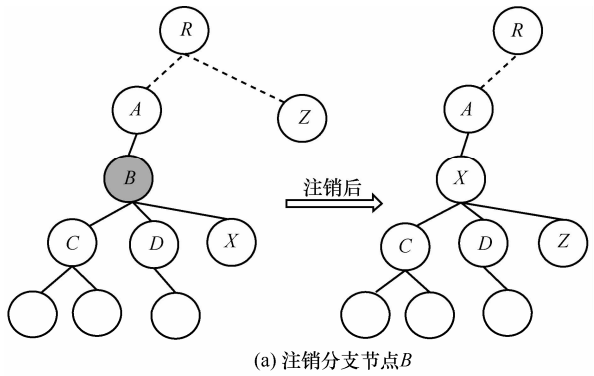
4.4 节点注销协议

当 MTPM 通过节点注册协议添加到 TPM 联盟中时，会形成如图 5 所示的时间树型拓扑的可信节点管理网络。网络中每一个 MTPM 的状态都是通过前驱节点一级一级主动汇报给 RTPM。当云管理员缩减节点规模时，可以注销一部分 MTPM，若注销的 MTPM 节点是分支节点，必然会造成后继节点的状态无法传递到 RTPM。为了使 TPM 联盟能够自由控制 MTPM 数量，同时又不破坏可信节点管理网络，需要对节点的注销采取以下策略，如图 8 所示。其中图 8(a)注销分支节点 B，图 8(b)注销叶子节点 X。虚线表示省略中间节点，R 代表 RTPM，其他字母代表 MTPM。具体流程如下。

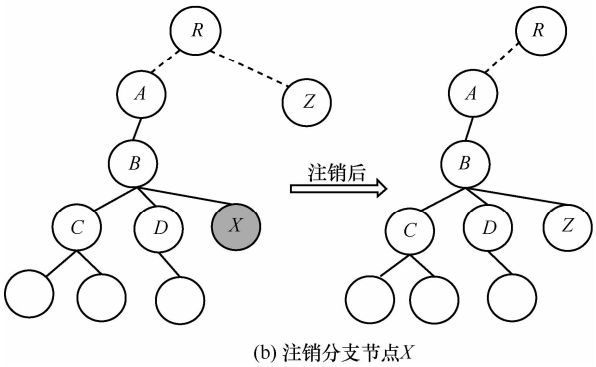
4.4.1 注销节点 B

1) R 向 B 发送注销命令和 PK_Z，同时将 B 从节点状态数据库中删除。

2) B 收到注销命令后，将 PK_A 和 PK_Z 转发给 X，将 PK_X 和 IP_X 发送给 A 和 R，将 ID_X 发送给 R，将 PK_X 发送给 C 和 D。此后 B 在可信节点管理网络中的职责由 X 来承担，B 关闭计算机。



(a) 注销分支节点B



(b) 注销分支节点X

图 8　节点注销示意

3) X 收到 PK_A 和 PK_Z 后，将 PK_Z 存储到 TPM 的非易失性存储器中。X 更新前驱节点的公钥为 PK_A，更新前驱节点的 IP 为 IP_A，将自己的编号改为 ID_B。C 和 D 收到 PK_X 后，更新前驱节点的公钥为 PK_X，更新前驱节点的 IP 为 IP_X。A 收到 PK_X 和 IP_X 后，将 B 的公钥信息修改为 PK_X，通过 IP_X 与 X 建立可信通道。R 收到 PK_X、IP_X 和 ID_X 后，将 PK_X、IP_X 和 ID_X 转发给 Z。在节点状态数据库中修改 Z 的编号为 ID_X，修改 X 的编号为 ID_B。

4) Z 收到 PK_X、IP_X 和 ID_X 后，向前驱节点发送终止命令，修改前驱节点的 IP 为 IP_X，修改前驱节点的公钥为 PK_X，修改自己的编号为 ID_X，重新执行节点注册协议。

4.4.2　注销节点 *X*

1) R 向 X 发送注销命令和 PK_Z，将 Z 的编号改为 ID_X，将 ID_X 发送给 Z，将 X 从节点状态数据库中删除。

2) B 将存储的 PK_X 修改为 PK_Z，向 R 发送 IP_B 和 PK_B。X 收到注销命令后关闭计算机。Z 收到 ID_X 后，将自己的编号改为 ID_X。

3) R 收到送 IP_B 和 PK_B 后，将 IP_B 和 PK_B 转发给 Z。

4) Z 收到 IP_B 和 PK_B 后向前驱节点发送终止命令，修改前驱节点的 IP 为 IP_B，修改前驱节点的公钥为 PK_B，重新执行节点注册协议。

当注销大量 MTPM 时，RTPM 应先注销叶子节点，其次注销编号大的分支节点，这样能保证各 MTPM 的注销同时进行。

4.5　节点状态实时监控协议

MTPM 的状态分为 3 种：可信，即系统完整性未遭破坏；不可信，即系统中某些进程的完整性遭到破坏但节点的 MTPM 完整性未遭破坏（记为类型 1）或者是节点的 MTPM 完整性遭到破坏但该节点的前驱和所有兄弟节点中至少有一个节点的 MTPM 完整性未遭破坏（记为类型 2）；状态未知，即 MTPM 的状态没有通过可信节点管理网络传递到 RTPM。节点的状态转换如图 9 所示，其中虚线椭圆表示该状态是暂态。

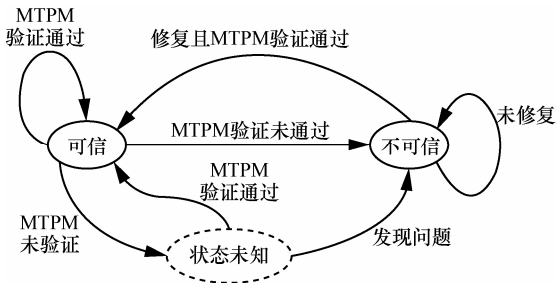


图 9　状态转换

分布式状态监控协议如下。

TPM 联盟中每个节点初始化 2 个空集合 NoTrustedList 和 LeftList，其中 NoTrustedList 中的元素为节点的编号及其类型，LeftList 中的元素为叶子节点编号。同时每个节点维护一个后继节点列表 SubsequentList，列表中元素按照节点编号降序排列。

TPM 联盟中每个节点进行如下操作。

1) 从 SubsequentList 取一个后继节点。

2) 判断与后继节点的可信通道是否断开，若是，则将后继节点编号及类型 2 插入 NoTrustedList；否则验证可信证据，若验证不通过，则将后继节点编号和类型 1 插入 NoTrustedList。

3) 判断后继节点是否为叶子，若是，将其编号插入到 LeftList。

4) 将后继节点的 NoTrustedList 插入到本节点的 NoTrustedList，将后继节点的 LeftList 插入到本节点的 LeftList。

5) 判断后继节点是否为 SubsequentList 中的最后一个元素，若否，则跳转到 1)执行。

6) 将本节点的可信证据、NoTrustedList 和

LeftList 传递给前驱节点。

RTPM 节点中的 NoTrustedList 和 LeftList 将会是关于整个 TPM 联盟中节点的状态信息。RTPM 通过 LeftList 计算出状态未知的叶子节点编号列表 NoStatusList。

1) 从 NoStatusList 中取一个叶子节点编号，并将其插入到 NoTrustedList，置类型为 2。

2) 以此编号计算前驱节点编号，若前驱节点编号未出现在 NoTrustedList 中，则将前驱节点编号插入到 NoTrustedList，类型置为 2，跳转到 2)执行。

3) 判断 NoStatusList 是否遍历完，若否，跳转到 1)执行。

4.6 可信节点管理网络修复协议

为了维护可信节点管理网络的完整性，采取以下修复策略。

1) 若是叶子节点出现故障，RTPM 通过警报器隔离该节点并向云管理员报警。云管理员修复该节点后，该节点依据节点注册协议重新加入 TPM 联盟。

2) 若是单个分支节点出现故障，修复策略如图 10 所示。

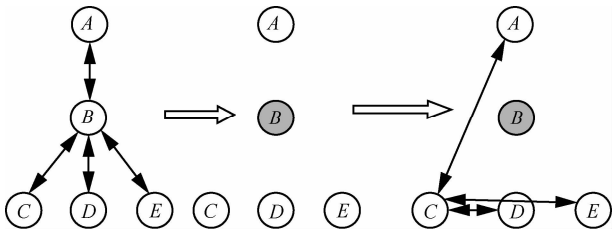


图 10 分支节点 *B* 故障修复策略示意

其中，*A*、*B*、*C*、*D*、*E* 代表 MTPM，*B* 故障造成可信节点管理网络局部连接中断，从 *B* 的后继节点 *C*、*D*、*E* 中选取一个节点代替 *B*。选取规则是选编号最大的节点。

假设图 10 中 *C* 编号最大，故选 *C* 代替 *B*，该局部网络的通信即可恢复。同时 RTPM 通过警报器隔离 *B* 并向云管理员报警，*B* 修复后依据节点注册协议重新加入 TPM 联盟，这时可信节点管理网络得到修复。

3) 若节点与前驱节点同时出现故障，修复策略如图 11 所示。

其中，*B*、*C*、*F*、*G*、*Z* 代表 MTPM，*R* 代表 RTPM。*B*、*C* 故障造成可信节点管理网络局部连接中断，此时 *B* 的后继节点依照图 10 的方式重建网络，本图中不再给出示例；从 *C* 的后继节点 *F*、*G* 中选取一个节点代替 *C*，选取规则依然是选编号最大的节点。图中假设 *F* 编号比 *G* 编号大，故选 *F* 代替 *C*。因为 *C* 的前驱 *B* 故障，所以 *F* 不再与 *B* 建立连接，而是直接与 *R* 建立临时连接，*R* 从 TPM 联盟中选一个编号最大的节点 *Z* 作为 *F* 的前驱节点，*F* 与 *Z* 建立连接后，该局部网络的通信即可恢复。同时 RTPM 通过警报器隔离 *B*、*C* 并向云管理员报警，*B*、*C* 修复后依据节点注册协议重新加入 TPM 联盟，这时可信节点管理网络得到修复。

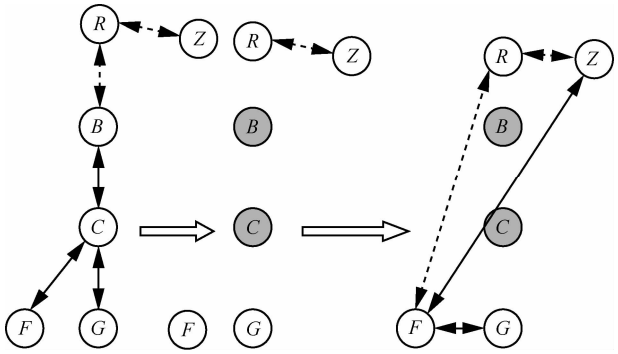


图 11 *BC* 同时故障修复策略示意

5 仿真实验及结果分析

下面针对本文提出的可信云平台管理模型从以下两方面进行实验验证及分析：1) TPM 联盟建立可信节点管理网络的时间开销分析；2) TPM 联盟节点状态监控的有效性分析。

5.1 性能分析

使用 2 台瑞达可信安全服务器，每一台配置 Intel Xeon E5620，4G DDRIII，STCM，运行 Linux 3.11.0，连接到 1 Gbit/s 网络。使用这样的环境测试节点注册协议 10 次，取平均值，执行一次节点注册协议的时间为 1.93 s。这个测量结果与文献[19]的测量结果基本吻合，文献[19]中节点证明协议花费 0.82 s，这是因为文献[19]是单向认证，TPM quote 操作只需执行一次，执行一次 TPM quote 操作大约需要 0.8 s，而本文的节点注册协议是双向认证，需要执行 2 次 TPM quote 操作。使用 Octave 模拟 100 万个节点建立 TPM 联盟，其结果如图 12 所示。

取执行节点注册协议所需时间的一半作为单位时间，因为第一个加入 TPM 联盟的 MTPM 节点不需要认证 RTPM，也就是只需执行一次 TPM quote 操作，同时注册协议中的第二次 TPM quote 操作，每个节点可以同时执行，即执行节点注册协议的一半时间隐藏在第一个节点加入 TPM 联盟的时间内。

Time tree one 表示节点一次 TPM quote 操作认证一个后继节点的方式建立 TPM 联盟，Time tree two 表示节点采用 Merkel tree[25]技术，一次 TPM quote 操作认证所有的后继节点，Monitor 表示采用文献[19]中的节点注册方法。从图中可以看出，Time tree one 方式下，100 万个节点加入 TPM 联盟大约花费 20 s，若采用 Time tree two 方式，则可以在更短的时间内提够更多的可信节点来提供云服务，例如大约 12 s 时已有 86 万个节点加入 TPM 联盟，而 Monitor 方式下，20 s 只能注册 1.3 万个节点。值得注意的一点是，当 TPM 联盟规模确定时，相应的时间树也就确定了，因此 Time tree one 和 Time tree two 这 2 种方式建立 TPM 联盟的最终时间是一致的。本文采用的时间树算法能够使节点注册的数量与时间成指数关系。文献[19]中，每秒大约注册 633 个节点，且节点注册的数量与时间成线性关系，显然当云规模巨大时，文献[19]中的方法是不能胜任的。

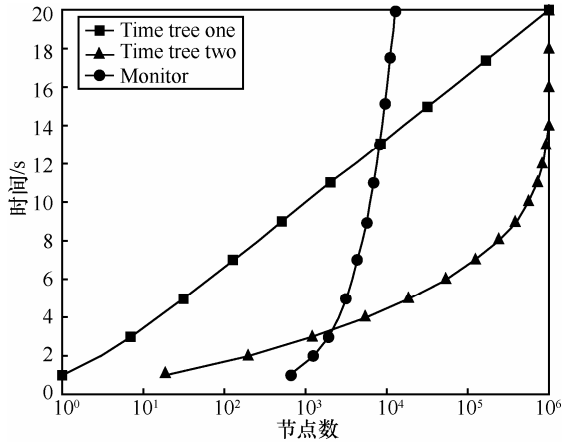


图 12　注册节点数量与时间关系

5.2　有效性分析

使用 Octave 模拟具有 100 万个节点的 TPM 联盟的节点状态，验证节点状态监控协议和可信节点管理网络修复协议能够使 RTPM 掌握所有节点的状态，同时给出每个节点通过以下哪种方式将状态传递给 RTPM。方式如表 1 所示。

表 1　　节点状态传递的不同方式

方式	描述
方式 1	节点状态通过前驱节点传递到 RTPM
方式 2	节点状态通过兄弟节点传递到 RTPM
方式 3	节点状态通过前驱的前驱传递到 RTPM
方式 4	节点状态通过临时前驱传递到 RTPM
方式 5	节点状态通过 RTPM 计算出来

其中，方式 1 为正常节点的状态传递方式，方式 2~方式 5 为故障节点的状态传递方式。实验设定节点故障率从 0.5%到 5%变化，为了模拟真实场景，其中节点的故障按照故障率随机产生，结果如图 13 所示。

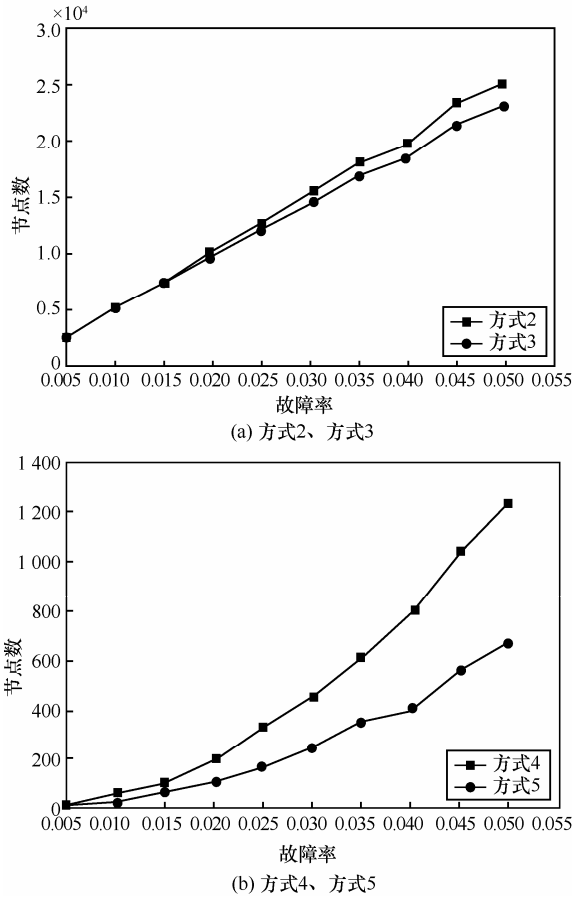


图 13　不同方式下的节点数量随着节点故障率的变化趋势

图 13 结果数据分析显示故障节点数量（即方式 2~方式 5 下的节点数量之和）占总节点数量的比重与之前设定的节点故障率一致。该结论很好地验证了此模型的有效性。其中方式 2 和方式 3 下的节点数量基本一致，且比方式 4 和方式 5 下的节点数量多，是因为前 2 种方式下的节点数量受单节点故障的影响（单节点故障是指该节点故障，而与该节点直接相连接的所有节点正常），而后 2 种方式下的节点数量受多节点同时故障（节点同时故障是指该节点与其直接相连接的部分或者所有节点同时故障）的影响。

6　结束语

构建可信云平台是云计算被广泛应用的重要前提，国内外研究者对如何构建可信云平台进行了

深入的研究，取得了重要研究成果。本文针对在可信云平台中由于 TPM 的低效性引起的可信节点管理的性能瓶颈问题，提出了基于 TPM 联盟的可信云平台的管理模型。该方案与已有方案相比，很好地解决了可信节点管理的性能瓶颈问题。

参考文献：

[1] 冯登国, 张敏, 张妍. 云计算安全研究[J]. 软件学报, 2011, 22(1): 71-83.
FENG D G, ZHANG M, ZHANG Y. Study on cloud computing security[J]. Journal of Software ,2011, 22(1): 71-83.

[2] GARFINKEL T, PFAFF B, CHOW J. Terra: a virtual machine-based platform for trusted computing[J]. ACM SIGOPS Operating Systems Review, 2003, 37(5): 193-206.

[3] BUTT S, LAGAR-CAVILLA H A, SRIVASTAVA A. Self-service cloud computing[C]//The 2012 ACM Conference on Computer and Communications Security. ACM, c2012:253-264.

[4] MCCUNE J M, LI Y, QU N. TrustVisor: efficient TCB reduction and attestation[C]//Security and Privacy (SP), 2010 IEEE Symposium. c2010:143-158.

[5] TADOKORO H, KOURAI K, CHIBA S. Preventing information leakage from virtual machines' memory in IaaS clouds[J]. Information and Media Technologies, 2012, 7(4): 1421-1431.

[6] BLEIKERTZ S, BUGIEL S, IDELER H. Client-controlled cryptography-as-a-service in the cloud[C]//Applied Cryptography and Network Security. Springer Berlin Heidelberg, c2013:19-36.

[7] CHEN C, RAJ H, SAROIU S. cTPM: a cloud TPM for cross-device trusted applications[C]//The 11th USENIX Conference on Networked Systems Design and Implementation USENIX Association. c2014: 187-201.

[8] 吴吉义, 沈千里, 章剑林. 云计算: 从云安全到可信云[J]. 计算机研究与发展, 2011, 48(l): 229-233.
WU J Y, SHEN Q L, ZHANG J L. Cloud computing: cloud security to trusted cloud[J]. Journal of Computer Research and Development, 2011, 48(l): 229-233.

[9] SCHIFFMAN J, MOYER T, VIJAYAKUMAR H. Seeding clouds with trust anchors[C]//The 2010 ACM Workshop on Cloud Computing Security Workshop. ACM, c2010: 43-46.

[10] DAVI L, SADEGHI A R, WINANDY M. Dynamic integrity measurement and attestation: towards defense against return-oriented programming attacks[C]//The 2009 ACM Workshop on Scalable Trusted Computing. ACM, c2009:49-54.

[11] BERGER S, CÁCERES R, PENDARAKIS D. TVDc: managing security in the trusted virtual datacenter[J]. ACM SIGOPS Operating Systems Review, 2008, 42(1): 40-47.

[12] BERGER S, CÁCERES S, GOLDMAN K. Security for the cloud infrastructure: trusted virtual data center implementation[J]. IBM Journal of Research and Development, 2009, 53(4): 6: 1-6: 12.

[13] SAYLER A, KELLER E, GRUNWALD D. Jobber: automating inter-tenant trust in the cloud[J/OL].http://www.usenix.org/node/174570, 2013.

[14] WU R, ZHANG X, AHN G J. Design and implementation of access control as a service for iaas cloud[J]. SCIENCE, 2013, 2(3): 115-130.

[15] 刘川意, 林杰, 唐博. 面向云计算模式的运行环境可信性动态验证机制[J]. 软件学报, 2014, 25(3): 662-674.
LIU C Y, LIN J, TANG B. Dynamic trustworthiness verification mechanism for trusted cloud execution environment[J]. Journal of Software, 2014, 25(3): 662-674.

[16] LI X Y, ZHOU L T, SHI Y. A trusted computing environment model in cloud architecture[C]//Machine Learning and Cybernetics (ICMLC), 2010 International Conference. IEEE, c2010:2843-2848.

[17] ZHANG F, CHEN J, CHEN H. CloudVisor: retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization[C]// The Twenty-Third ACM Symposium on Operating Systems Principles. ACM, c2011:203-216.

[18] SANTOS N, GUMMADI K P, RODRIGUES R. Towards trusted cloud computing[C]//The 2009 Conference on Hot Topics in Cloud Computing. c2009:3.

[19] SANTOS N, RODRIGUES R, GUMMADI K P. Policy-sealed data: a new abstraction for building trusted cloud services[C]//USENIX Security Symposium. c2012:175-188.

[20] 王丽娜, 任正伟, 董永峰. 云存储中基于可信平台模块的密钥使用次数管理方法[J]. 计算机研究与发展, 2013, 50(8): 1628-1636.
WANG L N, REN Z W, DONG Y F. A management approach to key-used times based on trusted platform module in cloud storage[J]. Journal of Computer Research and Development, 2013, 50(8): 1628-1636.

[21] 田俊峰, 吴志杰. 一种可信的云存储控制模型[J]. 小型微型计算机系统, 2013, 34(4): 789-795.
TIAN J F, WU Z J. Trusted control model of cloud storage[J]. Journal of Chinese Computer Systems, 2013, 34(4): 789-795.

[22] 张焕国, 陈璐, 张立强. 可信网络连接研究[J]. 计算机学报, 2010, 33(4): 706-717.
ZHANG H G, CHEN L, ZHANG L Q. Research on trusted network connection[J]. Chinese Journal of Computers, 2010, 33(4): 706-717.

[23] WANG J, ZHAO B, ZHANG H. POSTER: an E2E trusted cloud infrastructure[C]//The 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, c2014:1517-1519.

[24] 周振吉, 吴礼发, 洪征. 云计算环境下的虚拟机可信度量模型[J]. 东南大学学报(自然科学版), 2014, 44(1): 45-50.
ZHOU Z J, WU L F, HONG Z. Trustworthiness measurement model of virtual machine for cloud computing[J]. Journal of Southeast University (Natural Science Edition), 2014, 44(1):45-50.

[25] SZYDLO M. Merkle tree traversal in log space and time[C]//Advances in Cryptology-EUROCRYPT 2004. Springer Berlin Heidelberg, c2004:541-554.

作者简介：

田俊峰（1965-），男，河北保定人，博士，河北大学教授、博士生导师，主要研究方向为信息安全、分布式计算、网络技术、可信计算、云计算。

常方舒（1989-），男，河北邯郸人，河北大学硕士生，主要研究方向为可信计算和云计算。