基于 Chameleon 哈希改进的平台配置远程证明机制

付东来[1,2] 彭新光[1]
(太原理工大学计算机科学与技术学院 太原 030024)[1]
(中北大学电子与计算机科学与技术学院 太原 030051)[2]

摘 要 为了进一步提高平台配置远程证明机制的实用性,针对 RAMT(remote attestation based on Merkle hash tree)方案的不足,基于 Chameleon 哈希算法,采用软件分组的思想,改进了 RAMT 方案,给出了实验证明。认真讨论了 RAMT 方案的特点,详细描述了改进后的 RAMT 方案的体系结构、度量及验证过程,并深入讨论了新机制的特点。实验结果表明,新机制不仅提高了远程证明机制的可伸缩性,而且进一步增强了隐私保护能力,从而进一步提高了方案的实用性。

关键词 可信计算,远程证明,Chameleon 哈希,软件分组
中图法分类号 TP309 **文献标识码** A

Improved Remote Attestation Mechanism of Platform Configuration Based on Chameleon Hashes

FU Dong-lai[1,2] PENG Xin-guang[1]
(Institute of Computer Science and Technology, Taiyuan University of Technology, Taiyuan 030024, China)[1]
(Institute of Electronics & Computer Science and Technology, North University of China, Taiyuan 030051, China)[2]

Abstract In order to obtain a practical remote attestation mechanism for platform configurations, an improved RAMT (remote attestation based on Merkle hash tree) method was proposed using chameleon hashes and software group. And the relevant proof was given. The problems of the existing methods were analyzed. And the architecture of improved scheme, its process of integrity measurement and attestation were discussed in detail. The advantages of new scheme were also discussed. Compared with RAMT, the scalability and the ability to protect privacy are enhanced, and the efficiency of the remote attestation is improved highly.

Keywords Trusted computing, Rremote attestation, Chameleon hash, Software group

远程证明是可信平台模块的主要功能之一。根据证明内容的不同,可以将远程证明分为平台身份证明和平台配置证明两种方式。本文针对平台配置证明的问题展开研究。

可信计算规范[1,2]中给出了关于平台配置证明机制的描述。该机制将可信平台模块作为可信根,采用先度量后装载的机制,即在将控制权交给后续的可信实体前,首先度量该可信实体的完整性,并将度量结果保存在平台配置寄存器和度量日志中供后续的证明过程使用。平台配置寄存器位于可信平台模块内部,是一种特殊的寄存器。这些寄存器仅当平台被重新启动时才能被重置,即平台运行过程中寄存器的内容是不能被重置的,新的度量结果只能通过连接寄存器当前值的方式被保存,所以平台配置寄存器的内容被认为可以反映平台当前的配置状态。由于平台配置寄存器不能记录度量的中间过程,因此引入了度量日志来记录每一个中间步骤的度量事件和度量值。在远程证明过程中,平台配置寄存器的内容被签名后和度量日志一起发送给挑战者。挑战者首先使用该平台的公钥验证签名,然后根据度量日志重新计算度量值,最后再依据度量参考列表检查度量日志的内容。如果所有的验证过程都通过,则挑战者可以据此做出是否信任该平台的决策。

目前相关的研究工作已经证明可信计算规范提出的平台配置证明机制在实际应用中存在较多的问题,比如:隐私泄露,可伸缩性差,通信和管理负载较重,验证层次较低等。为此,Sailer 等人提出了一种较为实用的远程证明框架 IMA[3]。IMA 通过在系统中维护一张度量列表,将信任链延伸到应用层。与 TCG 提供的方法相比,IMA 提高了验证层次,但其他问题仍然存在。所以徐梓耀等人针对 IMA 的不足之处,利用 Merkle 哈希树,设计了一种保护隐私的高效远程验证机制(简称:RAMT 方案)[4],但由于应用层可信实体完整性度量的复杂性,因此该方案仍然有很大的改进空间。一方面,应用层大量的可信实体会导致 Merkle 哈希树的结点数急剧增加,从而导致了远程证明机制的效率和实用性骤然下降。另一方

到稿日期:2012-03-05 返修日期:2012-06-10 本文受山西省留学基金项目(2009-28),山西省自然科学基金项目(2009011022-2),中北大学自然科学基金资助。

付东来(1979-)男,博士生,讲师,主要研究方向为计算机网络与安全、信息隐藏、程序理解等,E-mail:hhluci@163.com;彭新光(1955-),男,博士,教授,主要研究方向为计算机网络与安全。

面，由度量参考列表引起的可伸缩性问题没有提出相应的改进措施。

为进一步提高平台配置远程证明机制的实用性，引入软件分组[5]的概念，利用 Chameleon 哈希函数对 RAMT 方案进行了改进。与 RAMT 方案相比，新机制一方面大量减少了 Merkle 哈希树的结点个数，提高了方案的实用性，另一方面缓解了由度量参考列表引起的可伸缩性问题，并且进一步增强了方案的隐私保护能力，提高了远程证明的灵活性。

1 Chameleon 哈希

Chameleon 哈希与 SHA-1 哈希函数不同，它利用一对公、私钥来产生哈希值。一个完整的 Chameleon 哈希算法由以下几个算法构成。

1)密钥生成算法：密钥生成算法可形式化定义为以下映射：

$KG:1^{\kappa}\rightarrow(pk,sk)$

κ 就是一个安全输入参数，pk 和 sk 分别是算法输出的公钥和私钥。

2)哈希生成算法：哈希生成算法可形式化定义为以下映射：

$CH:(pk,m,r)\rightarrow h\in\{0,1\}^{\tau}$

输入给定的消息 m、公钥 pk 和随机数 r，就可以获得一个长度为 τ 的哈希值。

3)冲突发现算法：冲突发现算法可形式化定义为以下映射：

$Forge:(sk,m,r)\rightarrow(m',r')$

输入给定的消息 m、私钥 sk 和随机数 r，就可以获得消息对 $(m',r')\neq(m,r)$，且

$CH(pk,m,r)=CH(pk,m',r')$

与标准的哈希函数相比，Chameleon 哈希增加了冲突发现算法。私钥的拥有者可以利用该算法找到 $(m',r')\neq(m,r)$，但两者的哈希值却相同，而且算法具有根据给定消息 m' 获得随机数 r'，使得 $CH(pk,m',r')=CH(pk,m,r)$ 的能力。这点对本文所提的可信远程证明机制非常重要。

新机制中采用的 Chameleon 哈希算法应满足以下几条安全属性：

1)抗冲突性：抗冲突性是 Chameleon 哈希算法的一个基本安全属性，即在没有私钥的情况下，无法获得与 (m,r) 哈希值相同的 (m',r')。

2)语义安全：无法通过 $CH(pk,m,r)$ 来获得 m 的任何信息，即无法通过 $CH(pk,m',r')$ 和 $CH(pk,m,r)$ 来区分不同的消息 m' 和 m。

3)私钥的安全性：攻击者在知道 m,r,m',r' 的情况下，无法获得私钥 sk 的任何信息，即使攻击者可以多次查询 (m,r) 的冲突值 (m',r')。

4)指定消息 m'：即私钥的拥有者可以指定消息 m' 来获得满足条件的 r'，使得 (m',r') 的哈希值与 (m,r) 的哈希值相同。

本文中选取的 Chameleon 哈希算法的具体细节可参考文献[6]。

2 CRAMT：改进的 RAMT 机制

2.1 体系结构

在 RAMT 机制中，大量的可信实体会导致 Merkle 树的体积较大，从而影响远程证明的效率。在 CRAMT 机制中，通过引入软件分组的概念可以有效地减小 Merkle 树的体积及可信实体完整性参考列表的大小，从而提高远程证明的效率。即借助 Chameleon 哈希函数给一个软件组的所有成员分配同样的哈希值。图 1 展示了 CRAMT 的体系结构。

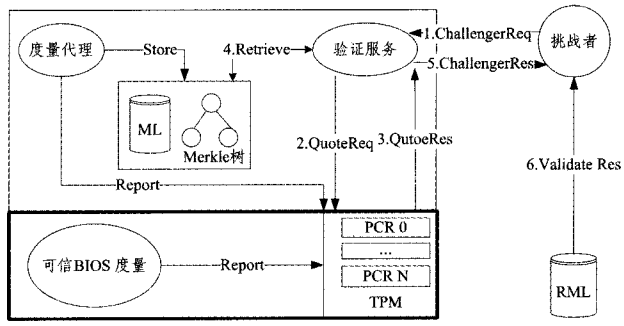

图 1 CRAMT 的体系结构

由图 1 可知，CRAMT 的体系结构与 RAMT 及 IMA 的体系结构非常类似，但由于所存储的可信实体的哈希值的变化，致使其在可信实体的完整性度量、验证及 TPM 的使用等方面存在较大的差别。

首先，CRAMT 在内核中所维护的完整性度量值 Merkle 哈希树的叶子结点所存储的数据对象是待验证计算平台上被度量的各种可信实体的 Chameleon 哈希值，而其内部结点则是由孩子结点的 Chameleon 哈希值连接后动态生成的。此外，还维护了一张度量列表，但该列表不是用作重构可信根哈希的值，而是用于检查认证路径的数据来源的正确性，且其中存储的是可信实体组的 Chameleon 哈希值。

其次，为保证认证路径的完整性而提供的可信实体完整性度量值参考列表中所存储的数据对象均为可信实体组的 Chameleon 哈希值，这有效降低了可信实体完整性参考列表的尺寸。

再次，RAMT 方案仅提供一个叶子结点的信息来保证认证路径的完整性是不够的。显然，为了验证认证路径的完整性，证明方需要提供所有叶子结点的完整性，但这同时又会陷入隐私泄露的困境。在 CRAMT 方案中，利用软件分组的思想解决了该问题，并进一步增强了隐私信息的保护能力，因为验证方无法获得一个具体的可信实体的完整性哈希值，其获得的仅仅是可信实体组的完整性 Chameleon 哈希值。

最后，CRAMT 方案提供了粒度可控的远程证明的能力，用户可根据软件分组的大小控制可信证明的层次与深度。

总之，CRAMT 尽管在软件分发、完整性度量及证明的总体思路上与 RAMT、IMA 等相同，但这些过程的细节却存在诸多的不同之处。

2.2 符号的定义

为了叙述方便，定义以下符号。

1)软件组：表示属于同一软件分组的软件集合，记为：$SG=\{st_i \mid i=1,2,3,\cdots,m\}$。

2)软件组集：由软件组 SG 构成的集合，记为：

$SGS=\{SG_i \mid i=1,2,3,\cdots,n\}$

2.3 软件分发算法

算法 1 软件分发算法

Step 1 令 i=1，对 $SG_i \in SGS$ 执行密钥生成算法 $KG:1^\kappa \rightarrow (pk,sk)$，获得一对公/私钥$(pk_i,sk_i)$；

Step 2 令 j=1，对 $st_j \in SG_i$ 执行哈希生成算法 $CH:(pk_i,m_j,r_j) \rightarrow h \in \{0,1\}^\tau$，$r_j$ 为一个指定的随机数，$m_j=SHA-1(st_j)$，同时将 h 写入可信实体完整性度量值参考列表 RML，r_j 随 st_j 发布；

Step 3 令 j++，当 $j \leqslant m$ 时，循环执行冲突发现算法 $Forge:(sk_i,m_j,r_j) \rightarrow (m_j',r_j')$，$r_j$ 由算法自动生成，仅需提供 $m_j=SHA-1(st_j)$，r_j' 随 st_j 发布；

Step 4 令 i++，当 $i \leqslant n$ 时，对 $sg_i \in SGS$ 执行密钥生成算法 $KG:1^\kappa \rightarrow (pk,sk)$，获得一对公/私钥$(pk_i,sk_i)$后，返回 Step 2，否则结束。

2.4 完整性度量算法

算法 2 完整性度量算法

Step 1 当可信实体 st 被装载前首先依据公式 m=SHA-1(st)获得 m 的值；

Step 2 执行哈希生成算法 $CH:(pk,m,r) \rightarrow h \in \{0,1\}^\tau$ 获得可信实体的 Chameleon 哈希值；

Step 3 将(st,h)写入度量列表；

Step 4 在 Merkle 哈希树中增加一个叶子结点，同时调用 TPM_HashTree 命令更新可信根哈希的值。与 RAMT 机制不同的是，在 CRAMT 机制中 TPM_HashTree 接口命令需要将哈希值的计算方法修改为 $CH:(pk,m,r) \rightarrow h \in \{0,1\}^\tau$。

2.5 完整性验证算法

算法 3 完整性验证算法

设验证方为 VF，证明方为 AS，则具体算法可描述如下(CRAMT 的远程证明过程如图 2 所示)：

Step 1 VF 产生一个 160 位的随机数 nonce，然后以挑战请求的格式向 AS 发送随机数 nonce 和所要请求的证明信息，即：$VF \rightarrow AS:ChReq(nonce,PCR_{NO.})$；

Step 2 AS 将请求转发给 TPM 芯片，即 $AS \xrightarrow{nonce,PCR_{rn}} TPM$；

Step 3 首先，由 TPM 芯片对可信根哈希值及随机数进行签名获得

$$Quote=Sig(PCR,nonce)AIK_{priv}$$

然后，获取 VF 请求的可信实体的哈希值的叶子结点 v_i 到根节点 rn 的认证路径：

$$v_i,u_j,\cdots,u_k,rn$$

最后，获取度量列表 ML；

Step 4 AS 将认证路径 $v_i,u_j,\cdots,u_k,rn$、Quote 及度量列表 ML 以挑战响应的格式发送给 VF，即：

$$AS \rightarrow VF:ChRes(Quote,(v_i,u_j,\cdots,u_k,rn),ML)$$

Step 5 VF 依次验证 AS 的签名和随机数 nonce；

Step 6 根据认证路径 $v_i,u_j,\cdots,u_k,rn$ 重新计算可信根哈希的值，并与 TPM 芯片签名的可信根哈希的值进行比较；

Step 7 检查度量列表 ML 中的项是否与可信实体完整性度量值参考列表 RML 中声明的相符；

Step 8 在 Step 5—Step 7 验证都通过后，验证方 VF 就可以据此判断是否信任 AS。

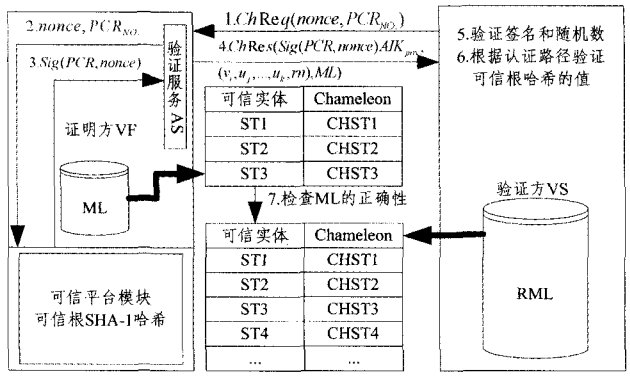

图 2 CRAMT 的远程证明过程

3 实验与分析

3.1 实验方案与设计

可信平台模块采用 TPM Emulator 0.7.1，选用 Fedora 10，内核为 2.6.27.38，TSS 采用 jTSS 0.4.1，开发工具采用 Eclipse 3.2，实现了一个基于套接字通信的客户端/服务器模式的可信远程证明程序，服务器端模拟验证方，客户端模拟证明方，可信实体完整性度量值参考列表均以文件的形式存储在磁盘上。为了实现 CRAMT 方案，做了以下几方面工作：

1)度量列表 ML 的读取问题：为了读取用于存储 Merkle 哈希树叶子结点的 Chameleon 哈希值的度量列表 ML，修改了 jTSS。

2)可信根哈希值的问题：为了实现方便，没有扩展 TPM_HashTree 接口命令，而是将可信根的 Chameleon 哈希值再次执行 SHA-1 运算后，扩展存储到 PCR 寄存器，并把更新的历史情况存储到度量列表 ML。

3)度量列表 ML 和 Merkle 哈希树存储位置：为了编程方便，将度量列表 ML 和 Merkle 哈希树均存放到了用户空间。

4)可信根哈希的验证问题：由于证明方提供的可信根哈希是经过几次迭代更新而形成的 SHA-1 哈希值，因此它在依据认证路径重新计算后，仍需要依据度量列表 ML 中可信根哈希的更新历史数据重新计算后，才能与证明方提供的 PCR 值进行比较。

5)软件的随机数问题：随软件发布的随机数 r 存储到文本文件中。

3.2 实验分析

3.2.1 可伸缩性

为了测试 Merkle 哈希树中叶子结点的变化情况，首先根据算法 1(软件分发算法)，通过包的方式对软件进行分组，并对包内的文件分配随机数 r。表 1 展示了 RAMT、IMA 方案和 CRAMT 方案下 Merkle 树中结点数的统计情况。

表 1 RAMT、IMA 方案和 CRAMT 方案结点数比较表

度量方法	叶子结点数	总结点数
CRAMT	769	1155
RAMT	6986	10480
IMA	6986	6986

由表 1 可知，在 RAMT 方案下，Merkle 树需要创建 10480 个结点，而在 IMA 方案中，需要创建 6986 个结点(由

于IMA方案采用的是线性结构，因此叶子结点数与总结点数相同)，若采用CRAMT方案(一个包一个哈希值的方法)，仅需要创建1155个结点。而且叶子结点的数量实际上就等于可信实体完整性度量值参考列表的记录数。因此，采用CRAMT方案能够有效地提高远程证明方案的可伸缩性。

3.2.2 隐私保护能力

在RAMT方案的远程证明中，采用基于认证路径的方法，即可信根哈希的值是按照从叶子结点到根结点的路径进行重新计算后，与证明方提交的根哈希值进行比较，从而推断可信根哈希的值的完整性，所以验证方仅知道一个叶子结点的信息，即一个可信实体的哈希值。而在IMA中需要发送所有可信实体的哈希值即ML，所以证明方的所有隐私信息暴露给了验证方。显然，在隐私保护方面，RAMT方案要强于IMA方案。

但是RAMT方案在远程证明的最后一个步骤，仅将一个叶子结点的信息与可信实体完整性度量值参考列表的记录进行比对，而包含在认证路径中的由其他叶子结点生成的中间结点却无法保证其有效性，所以据此做出的信任决策的正确性也是难以保证的。因此，从这个角度分析，IMA方案又占有一定的优势。

在CRAMT方案的远程证明过程中，仍然采用基于认证路径的方法验证根哈希的完整性。与RAMT方案不同的是，证明方将Merkle树中所有叶子结点的信息(即ML列表)提供给了验证方，显然验证方据此做出的信任决策的正确性要高于RAMT方案；与IMA方案相比，因为ML列表中存放的仅仅是软件组的哈希值，所以纠正了IMA方案泄露隐私的缺陷。实际上CRAMT方案不仅没有泄露任何隐私信息，而且进一步增强了隐私保护能力，因为验证方不能够获得任何一个具体的可信实体的哈希值，相比之下RAMT方案需要泄露一个可信实体的具体信息，这实际上在有些情况下会导致软件歧视、剥夺用户的软件选择权的现象。比如在数字版权管理[7]的应用领域中，流媒体提供商可能会与流媒体软件客户端的供应商合谋，强制用户使用特定的客户端软件。新机制可以避免该现象的发生。因此，采用CRAMT方案进一步增强了远程证明的隐私保护能力。

3.2.3 验证方式的灵活性

与RAMT、IMA相比，CRAMT方案的验证方式具有较强的灵活性，因为它引入的软件组可大可小，软件供应商可以将其开发的所有软件作为一个软件组，也可以将一个类别作为一个软件组，甚至还可以将一个软件的不同版本作为一个软件组，当然也可以将一个软件作为一个软件组。但软件组的不同与隐私保护能力有关，软件组越大，隐私保护能力越强，反之软件组越小，泄露的隐私信息就会越多。所以，在实际的远程证明过程中，用户可根据情况选择不同的软件分组。

3.2.4 验证效率的分析

与RAMT、IMA方案相比，CRAMT方案中Merkle树的结点数及可信实体完整性度量值参考列表的记录数的急剧减少，极大地提升了方案的整体效率。实际上可能影响CRAMT方案远程证明效率的因素主要有以下两个方面：

1)度量列表的传输对通信负载的影响：为保证认证路径的有效性，CRAMT方案向验证方提供了所有叶子结点的信息，即度量列表，方案中度量列表的记录长度为200个字节，叶子结点数为769个，所以总大小为153800个字节，约为150k个字节，因此对通信不会构成影响。

2)Chameleon哈希值的计算对效率的影响：为了提高Chameleon哈希函数的性能，方案中均先对可信实体做了SHA-1算法。文献[5]给出了SHA-1算法及Chameleon算法在不同长度的输入下所消耗的时间，如表2所列。

表2 Chameleon哈希和SHA-1哈希的时间性能数据

哈希算法	2byte	1kB	1Mb
SHA-1	2微秒	20微秒	18312微秒
SHA-1+CH	4677微秒	4694微秒	22986微秒
增长率	99.8%	99.6%	20%

由表2可知，随着文件尺寸的增大，在SHA-1算法的基础上，CH算法所消耗的时间的增长率的下降速度是比较快的，所以这在实际中是可以接受的。

结束语 在RAMT方案的实际应用过程中，一方面由于应用层存在大量的可信实体，因此Merkle树的结点数较多，会影响远程证明方案的可伸缩性。另一方面，在远程证明的最后一个关键环节，仅将一个叶子结点的信息与可信实体完整性度量值参考列表的记录进行比对，而在认证路径中由其他叶子结点生成的中间结点却无法保证其有效性，据此做出的信任决策的正确性也是难以保证的。本文基于此，采用软件分组的思想，利用Chameleon哈希算法，改进了RAMT方案。实验证明，新方案不仅提高了远程证明方案的可伸缩性，增强了隐私保护能力，避免了“软件歧视”现象的发生，而且使得验证方式更加灵活。

参考文献

[1] Trusted Computing Group. TCG specification architecture overview revision 1.4[EB/OL]. http://www.Trusted comp- utinggroup.org/,2007

[2] Trusted Computing Group. TPM main specification version 1.2 revision 103 part 1 & 2 & 3[EB/OL]. http://www.trustedcomputinggroup.org/,2007

[3] Sailer R,Zhang X L,Jaeger T,et al. Design and implementation of a TCG-based integrity measurement architecture[C]// Proceedings of the 13th USENIX Security Symp. Berkley:USENIX Association,2004:223-238

[4] 徐梓耀，贺也平，邓灵莉. 一种保护隐私的高效远程验证机制[J]. 软件学报，2011，22(2)：339-352

[5] Alsouri S, Dagdelen Ö, Katzenbeisser S. Group-based attestation: Enhancing privacy and management in remote attestation [C]//Proceedings of the 3rd International Conference on Trust and Trustworthy Computing, TRUST 2010. Berlin: Springer-Verlag,2010:63-67

[6] Ateniese G,de Medeiros B. On the key exposure problem in chameleon hashes[C]// Proceedings of the 4th International Conference on Security in Communication Networks,SCN2004. Italy:Springer-Verlag,2005:165-179

[7] 张志勇，牛丹梅. 数字版权管理中数字权利使用控制研究进展[J]. 计算机科学，2011，38(4)：48-54