软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(6):1273−1290 [doi: 10.13328/j.cnki.jos.004414]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种可信终端运行环境远程证明方案*

谭 良[1,2], 陈 菊[1]

[1](四川师范大学 计算机学院,四川 成都 610068)

[2](中国科学院 计算技术研究所,北京 100190)

通讯作者: 谭良, E-mail: jkxy_tl@sicnu.edu.cn

摘 要: 可信终端的远程证明无论是基于二进制的证明方案还是基于属性的证明方案,针对的均是终端的静态环境,反映的是终端的软件配置结构,并不能证明终端运行环境的真正可信.针对这一问题,提出了一种终端可信环境远程证明方案.针对静态环境,该方案考虑了满足可信平台规范的信任链以及相关软件配置的可信属性证明;针对动态环境,该方案考虑了终端行为的可信属性证明.并分别给出了信任链、平台软件配置和终端行为等属性证明的可信性判定策略和算法,以及终端运行环境远程证明的综合性判定策略和算法.另外,在 Windows 平台上,设计和实现了该方案中的两个核心实体:证明代理和验证代理,并设计了证明代理和验证代理之间的通信协议.最后,介绍了该方案在 Windows 平台上的一个典型应用案例以及证明代理在该应用实例中的性能开销.应用实例验证了该方案的可行性.

关键词: 可信计算;远程证明;属性证明;终端行为;证明代理;验证代理

中图法分类号: TP309

中文引用格式: 谭良,陈菊.一种可信终端运行环境远程证明方案.软件学报,2014,25(6):1273−1290. http://www.jos.org.cn/1000-9825/4414.htm

英文引用格式: Tan L, Chen J. Remote attestation project of the running environment of the trusted terminal. Ruan Jian Xue Bao/Journal of Software, 2014,25(6):1273−1290 (in Chinese). http://www.jos.org.cn/1000-9825/4414.htm

Remote Attestation Project of the Running Environment of the Trusted Terminal

TAN Liang[1,2], CHEN Ju[1]

[1](College of Computer, Sichuan Normal University, Chengdu 610068, China)

[2](Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100190, China)

Corresponding author: TAN Liang, E-mail: jkxy_tl@sicnu.edu.cn

Abstract: Remote attestation, whether binary-based or property-based, mostly undertakes the static environment of the trusted terminal where only part of software configurations in the trusted terminal are demonstrated, leaving trustworthiness of the dynamic running environment unproved. To resolve the problem, a new property-based remote attestation project for the dynamic running environment of the trusted terminal is presented. The project focuses not only on trusted chain and software configuration for the static environment of the trusted terminal, but also on the behaviors of the trusted terminal for the dynamic environment. Moreover, the decidability and algorithm for the trustworthiness of each property by each specific trusted policy is analyzed, and the comprehensive decision strategy is put forward. After that, attestation agent and verification agent which are critical entities in the project, are designed and implemented on Windows, and the communication protocol between them are designed too. Finally, an application case of the project on Windows is introduced, the performance of attestation agent in this application is studied, and the feasibility of the project is demonstrated.

Key words: trusted computing; remote attestation; property attestation; behaviors on terminal; attestation agent; verification agent

当前,基于 TPM[1]和 TCM[2]构建可信计算环境已经成为可信计算的研究热点.基于 TPM/TCM 的可信度量、

* 基金项目: 国家自然科学基金(60970113); 四川省青年科技基金(2011JQ0038)
收稿时间: 2012-05-20; 修改时间: 2012-07-23; 定稿时间: 2013-03-11

报告和远程证明功能实现了通信双方信任关系建立,因此成为了构建分布式可信计算环境的核心功能.目前,关于远程证明的研究主要包括基于实体标识的二进制证明[3−7]和基于属性的远程证明[8−14].

在二进制证明过程中,证明请求包含一系列关于被证明系统中当前运行实体的声明,其中,每个实体都通过度量机制生成的唯一性标识来表示.接收到证明请求后,验证方基于表示实体标识间信任传递关系的信任链组成的可信策略,从而确定被证明系统状态是否可信.TCG 体系中,远程证明的主要作用是针对平台部件的完整性进行测量,包括检查部件代码内容的非授权篡改以及鉴别部件的提供者身份,实质是一种基于二进制的完整性验证.其优点是证明过程简单、可靠,不需要可信第三方的参与,而验证方则能够可靠地构建起证明平台上从硬件信任根开始的信任链.但是,它最大的缺点是对平台配置隐私的暴露,证明过程中要求出示整个平台配置的完整性度量值.

针对二进制证明的上述缺点,研究者们设想从抽象的属性或者语义上去理解和构建证明平台.基于属性的远程证明中,证明方只需要根据验证方的目标属性给出相应的属性声明,不需要将实体唯一性标识暴露给验证方;同时,由于系统中运行实体的唯一性标识也可认为是系统的属性之一,因此基于属性的远程证明在保护系统隐私的同时也提高了证明方案的灵活性.但基于属性的远程证明只有抽象的概念模型,难以度量、比较和推导,从而无法构建类似二进制度量的信任链传递信任的功能.

值的注意的是,无论是二进制证明方案还是基于属性证明方案,针对的均是终端的静态环境,即反映的是终端的软件配置结构,并不能证明终端运行环境的真正可信.例如,即使是经过度量的应用程序,也不可能保证终端整个运行环境的可信,因为对应用程序进行完整性度量只能保证该应用程序没有被恶意程序修改,但不能保证该应用程序本身是否破坏终端的运行环境,如泄露内存、劫持网络、破坏文件等;再如,即使终端配置有符合可信策略的软件,但如果这个软件本身存在漏洞,终端的运行环境也会存在被破坏的可能.针对这一问题,本文提出了一种新的终端可信环境远程证明方案.该方案不仅考虑了满足可信平台规范的信任链以及相关软件配置的可信属性证明,而且针对运行环境考虑了终端行为的可信属性证明,并分别给出了信任链、相关软件配置和终端行为等属性证明的可信性判定策略和算法以及终端运行环境远程证明的综合性判定策略和算法,从而证明终端运行环境的真正可信.

本文第 1 节给出相关工作的描述.第 2 节给出基于属性的远程证明模型.第 3 节给出本模型中可信策略的可判定性定理及证明.第 4 节给出终端运行环境的远程证明方案在 Windows 平台上的实现.第 5 节给出终端运行环境远程证明方案的应用案例研究.第 6 节是比较与评价.最后是本文的总结.

1 相关工作

终端远程证明最初是由 TCG 组织给出的[1],TCG 组织一直是这一工作积极的主导者和推动者.在 TCG 的远程证明过程中,主要包括证明方与验证方,假设平台使用者想要向验证方证明自己的平台上有一个合法的 TPM,并利用这个 TPM 对平台的 PCR(platform configuration register)进行签名,实现向第三方证明其平台的合法性.最直接的方法是:平台用 EK 私钥对 PCR 签名并发送给验证方,验证方验证签名后信任平台为一个可信平台,且其配置信息 PCR 为可信.上述方法的问题在于:平台使用者的 EK 固定,当他/她与不同的验证方多次进行上述协议时,其交易记录可被第三方关联起来(linkable),从而无法保护平台使用者的隐私.TCG 组织在 TPM 规范 1.1 版本中提出了以(privacy CA)认证 ID 密钥(attestation identity key,简称 AIK)的方案来解决上面的问题.在该方案中,平台不再以 EK 作为签名的密钥,而是每次临时生成一个新的 AIK 作为签名密钥.为了证明 AIK 的合法性,平台必须首先向隐私 CA 申请一个 AIK 证书.当隐私 CA 收到平台的申请后,隐私 CA 用自身的私钥对 AIK 签名.平台获得这个签名后可以发送给第三方的验证者,验证者根据隐私 CA 的公钥验证 AIK 是否合法,合法的 AIK 对 PCR 的签名被视为 TPM 的合法签名.Privacy CA 方案的致命缺陷在于:其必须在保证 PrivaCy CA 可信的同时,还必须有高响应能力,所以其应用必将成为可信平台验证的瓶颈.TCG 定义 TPM 规范 1.2 版中提出了直接匿名认证(direct anonymous attestation,简称 DAA)协议[15],该协议在实现 TPM 芯片认证的同时,保证了认证的匿名性,验证方无法得到芯片的唯一标示.但该协议非常复杂,包含 4 次零知识证明,运算量非常大,包括大量的

模指数运算,因此,该协议还缺乏实用性.目前,对该协议的优化工作是一个研究热点[16−27].

除此以外,国内外众多研究机构和学者提出了许多不同的远程证明方法.从遵循 TCG 规范的直接二进制证明,到基于高级语言的语义证明;从嵌入式设备的基于软件证明,到 Web Service 的证明.在这众多的证明方法中,研究成果最多的还是基于实体标识的二进制证明(binary attestation)[3−7]和基于属性的远程证明[8−14].

在实体标识的二进制证明方面,IBM 研究院提出了完整性度量方案 IMA[4],该方案对于 Linux 系统加载的可运行实体进行度量,从而生成该实体的唯一性标识符.在证明时,验证方以可信计算平台作为信任根,从而建立对于证明方包含实体标识的判定.基于 IMA 方案,Jager 等人提出了基于策略规约的远程证明方案 PRIMA[5],该方案基于 C-W 信息流模型对远程证明过程中需证明的实体进行规约,从而在一定程度上防止了远程证明过程中平台配置的隐私暴露.

基于属性的远程证明方面,2004 年,德国波鸿鲁尔大学的 Sadeghi 等人提出了基于属性的远程证明概念和模型[10],给出了基于属性的远程证明方案的软硬件实现方法,该方法在可信引导程序 Trusted Grub 的基础上对基于属性的远程证明方法、属性验证和撤销等实现技术进行了研究.随后,为了从框架上解决远程证明过程中隐私泄露问题,同时保证证明方案的灵活性,Poritz 等人率先从体系架构层次提出了基于属性的远程证明[11],在该框架中,Poritz 等人讨论了基于属性远程证明在隐私保护、扩展性等方面的优点.在属性证明的实现方面,波鸿鲁尔大学利用在线可信第三方颁发了属性证书,实现了引导过程中的二进制度量转换为属性,基于属性实现了系统的证明和封装.该方案建立在在线的可信第三方基础上,便于完整性管理和安全属性管理,并采用 CRL 验证属性的撤销.随后,他们又对这个方案进行改进,实现了基于属性的系统引导器解决属性证明过程中的属性证书的版本回滚问题.在协议方面,Chen 提出了基于属性的远程证明协议(简称 PBA 协议)[9].文献[26]提出了使用远程证明扩展 SSL 协议的方案,通信终端通过协商安全参数和在 SSL 协议上证明平台配置,以此达到建立远程可信通道.该方案的提出,为基于现有通信协议实现远程证明过程中证明请求的发送给出了很好的解决方法.文献[13]提出了一种基于属性的远程证明模型.该模型将传统远程证明中信任链模型扩展为信任图,使得模型能够表达更为灵活的可信策略;文献[14]基于可信计算中的二进制系统完整性测量模型,增加证书权威和可信属性权威,提出一种属性远程证明系统完整性测量模型,并利用谓词逻辑证明其可信性.

2 终端运行环境的远程证明方案

终端运行环境的远程证明方案包含 5 个实体:证明代理、验证代理、属性证书颁发中心、可信策略库管理方和服务提供方.验证过程如图 1 所示.其中,

(1) 证明代理将终端完整性测量值、软件配置以及终端用户行为发送给验证代理;

(2) 验证代理根据完整性测量值、软件配置结构以及终端用户行为,对照其可信策略库中的安全策略,包括平台组件属性及完整性特征值、软件配置目录结构以及恶意行为特征.如果均满足可信策略库的属性安全策略,则向属性证书颁发中心申请一个终端运行环境的可信属性证书;

(3) 属性证书颁发中心颁发终端运行环境的可信属性证书,并返回给证明代理;

(4) 终端向服务方证明自己的可信属性,并获得服务方资源.

在此远程证明过程中,证明代理的主要功能是获取 TPM 的 PCR 二进制标示、软件配置以及终端环境中的用户行为;验证代理的主要功能是基于可信策略库对证明代理收集的信息进行确认,包括满足可信平台规范确认、软件配置确认、行为确认以及属性证书申请;属性证书颁发中心的主要功能是颁发和管理属性证书.显然,此远程证明过程不仅具有基于属性的远程证明方案的所有优点,如:(1) 易实现复杂开放网络环境下的异构平台之间的验证;(2) 无论厂商是谁,具备相同属性的任何配置均被赋予相同的属性证书;(3) 避免直接暴露平台配置给未知验证方,使恶意对手攻击平台更容易等等,而且还具有如下特征:①增加了对终端环境用户行为的可信属性证明,使得该远程证明方案不仅能证明终端的静态环境可信,而且通过对终端活动进程的行为属性分析远程证明终端的动态环境可信;②远程证明过程中,证明代理、验证代理、可信策略库、证书颁发中心和服务方的关系更加具体明确.

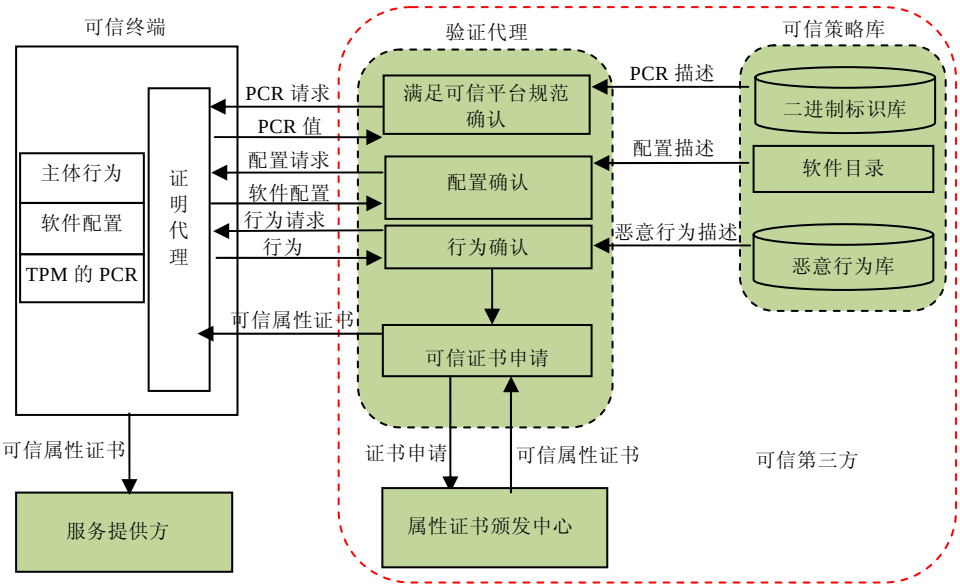

Fig.1 Remote attestation project of the running environment of the trusted terminal

图 1 终端运行环境的远程证明过程方案

值得注意的是,在一个安全域中,验证代理、可信策略库以及属性证书颁发中心通常可以看成为可信第三方.一方面,它们接受此域中各个可信终端无条件信任,接收来自可信终端的 PCR 二进制标示、软件配置以及终端环境中的用户行为,并保证这些信息不被泄漏;另一方面,它们接受此域中各种服务提供方无条件信任,均把它们发出的属性证书作为获得服务的主要凭证和相关授权的主要凭证;第三,验证代理、可信策略库以及属性证书颁发中心本身是安全可靠的.

3 终端运行环境远程证明的判定策略

终端运行环境远程证明方案的核心是判定问题.判定依赖于具体的策略,而策略必须与终端实际环境相吻合.因此,必须详细分析终端环境的信任链传递、软件配置和用户行为,为远程证明的判定策略提供基础保证.为了便于叙述,我们先形式化定义软件组件、属性、属性集等基本概念.

定义 1. 终端环境的软件组件定义为 $c_i=\{(c_{i\text{-}code},c_{i\text{-}d\text{-}code},c_{i\text{-}cofile}),(c_{i\text{-}f},c_{i\text{-}v},c_{i\text{-}o})\}$,其中,$(c_{i\text{-}code},c_{i\text{-}d\text{-}code},c_{i\text{-}cofile})$表示软件的代码特征,$c_{i\text{-}code}$ 代表软件 c_i 的源代码,$c_{i\text{-}d\text{-}code}$ 代表软件 c_i 的依赖,如其依赖的静态库、动态库或第三方代码库,$c_{i\text{-}cofile}$ 代表软件 c_i 的策略配置文件.$(c_{i\text{-}f},c_{i\text{-}v},c_{i\text{-}o})$表示软件 c_i 的功能特征,$c_{i\text{-}f}$ 表示 c_i 的功能,$c_{i\text{-}v}$ 表示测组件 c_i 的版本号,$c_{i\text{-}o}$ 表示软件组件 c_i 的其他相关属性,比如软件名、开发者、发布时间等.

依据定义 1,软件组件集为 $C=\{c_1,c_2,\ldots,c_n,\ldots\}$.

定义 2. 属性是指软件组件满足某一功能要求而具有的内在特征,形式化为 p_i;属性集 P 是指包含各种属性的集合,即 $P=\{p_1,p_2,\ldots,p_n,\ldots\}$.

我们用 p_{tpm} 表示终端运行环境满足信任链传递的可信平台规范属性,$p_{soft\text{-}configration}$ 表示终端运行环境满足一定可信策略需要的软件配置属性,$p_{behavior}$ 表示终端运行环境满足一定可信策略需要的用户行为属性.下面将分析属性 p_{tpm},$p_{soft\text{-}configration}$ 和 $p_{behavior}$ 及其判定策略.

3.1 属性p_{tpm}的分析与判定

属性 p_{tpm} 的实质含义是指终端系统满足基于完整性测量的信任链传递.从第 2 节的分析可以看出,在本方案的远程证明过程中,证明代理 T_{Agent} 运行在被验证平台上,起着非常重要的作用.为了保证 T_{Agent} 可信,我们扩展终

端的信任传递过程为

$$CRTM\rightarrow BIOS\rightarrow OS\ Loader\rightarrow OS\rightarrow T_{Agent}\rightarrow Application.$$

如图2所示,将T_{Agent}作为可信平台链式度量的重要一环.这种方式是可行的,T_{Agent}的度量可由OS主导完成,并由OS将T_{Agent}程序作为第1个应用程序首先启动.

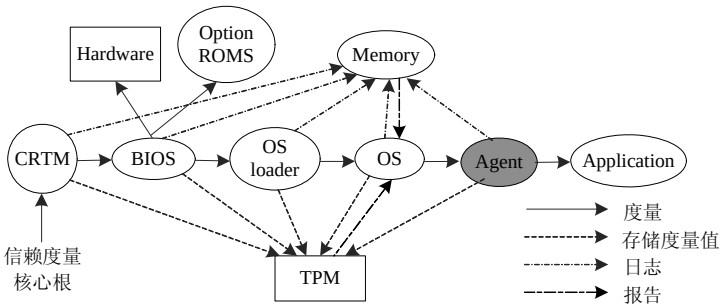

Fig.2 Measurement of attestation agent by TPM

图2 基于TPM的证明代理T_{Agent}的可信度量

在图2中,实现信任传递的相关参数主要包括:

(1) 系统配置:将完整性测量组件的哈希值存入TPM的24个PCR中;

(2) 存储测量值日志(storage measurement log,简称SML),其中包含了存储在TPM中的所有测量值的事件结构以及被测量的软件组件的$(c_{i\text{-}f},c_{i\text{-}v},c_{i\text{-}o})$.

为了详细描述系统完整性测量模型,我们定义信任根、完整性测量组件两个概念以及测量函数、完整性验证函数和完整性传递函数.

定义3. 信任根集(trusted root set)为一个由特殊属性组成的集合,包含所有其可信性无需进行证明的属性,形式化表示为$TRS=\{r|r\in P\}$.

根据定义3,由图2可知,可信终端的信任根集只有一个元素,即具有"首先启动、防被物理攻击以及能度量BIOS等功能"的CRTM,即$TRS_t=\{CRTM\}$.

定义4. 完整性测量组件集$I=\{i_1,i_2,\ldots,i_n,\ldots\}$,其中,$i_n\in C$.

由图2可知,终端的完整性测量组件集$I_t=\{CRTM,BIOS,OS\ Loader,OS\ Kernel,T_{Agent},Applications\}$,显然,$I_t\subset C$.

定义5. $Measure(i_n,i_{n+1},PCR_{n+1},sml_{n+1})$表示在完整性测量过程中,$i_n$对$i_{n+1}$的完整性进行测量,其增量哈希值存储于$PCR_{n+1}$,相对应的存储测量值日志为$sml_{n+1}$,$n$是正整数且$0\leqslant n\leqslant 23$.

定义6. 完整性验证函数$Verify(i_n,i_{n+1},PCR_{n+1},LPCR_{n+1})$表示将$i_n$对$i_{n+1}$的完整性测量值$PCR_{n+1}$与可信策略库中对应的标准$LPCR_{n+1}$进行比较.如果完全匹配,返回true;否则,返回false.

定义7. 完整性传递函数$Integrity(i_n,i_{n+1})$表示完整性能够从i_n有效地传递至i_{n+1},而不遭受破坏与损失.

根据以上定义,我们得出属性p_{tpm}的判定算法,如图3所示.

图3算法首先判断信任根集是不是空,然后判断信任根集里的信任根是不是唯一.如果这两个条件均成立,则调用*measure*函数对$CRTM\rightarrow BIOS\rightarrow OS\ Loader\rightarrow OS\rightarrow T_{Agent}$进行完整性测量,并调用完整性验证函数*Verify*进行验证,如果成立,则调用*Integrity*进行完整性可信传递,然后输出为p_{tpm},表示终端运行环境满足信任链传递的可信平台规范属性;否则,输出*NULL*.

算法 1. 属性 P_{tpm} 的判定算法 Check-trusted Platform Specification (I_t).
输入:I_t;
输出:p_{tpm} 或 *NULL*.
1. IF (TRS_t==∅) OR (|TRS_t|==0) THEN return NULL ENDIF; //|I_t|表示集合 I_t 的元素个数
2. IF (|TRS_t|==1) AND CRTM in TRS_t THEN //判断信任根是否存在且唯一
3. i_1=CRTM
4. FOR n=1 to |I_t| DO
5. *Measure*(i_n,i_{n+1},PCR_{n+1},sml_{n+1})
6. IF (*Verify*(i_n,i_{n+1},PCR_{n+1},$LPCR_{n+1}$)==true) THEN
7. *Integrity*(i_n,i_{n+1});
8. ELSE
9. RETURN *NULL*;
10. ENDIF
11. RETURN p_{tpm}
12. ENDFOR
13. ENDIF

Fig.3 Decidability algorithm of P_{tpm}
图 3 属性 P_{tpm} 的判定算法

3.2 属性$p_{soft\text{-}configration}$的分析与判定

属性 $p_{soft\text{-}configration}$ 的实质含义是指终端运行环境的软件配置满足一定可信策略.

为了详细分析属性 $p_{soft\text{-}configration}$,我们首先定义平台软件配置、基本配置等基本概念以及 *Draw*(), *Search-property*()和 *Verify_softwareconfiguration*()函数.

定义 8. 平台软件配置是一个平台运行的软件序列,记为$\sigma=\{c_1,c_2,\ldots,c_m\}$,其中,$c_i\in C$.显然,$\sigma\subset C$.

一个平台在不同任务要求下可能运行不同的软件,也可能以不同的顺序运行软件.因此,可能有多个不同的软件配置,所有的平台软件配置记为 C 的幂集 $P(C)$.

一个配置$\sigma=\{c_1,c_2,\ldots,c_m\}$中,若 c_i 在 c_j 的前面执行,记为 $c_i \xrightarrow{\sigma} c_j$.

定义 9. 设σ是一个配置,$c_i,c_j,\ldots,c_k\in C$ 并且 $c_i \xrightarrow{\sigma} c_j \xrightarrow{\sigma} \ldots \xrightarrow{\sigma} c_k$,那么 $\tau=\langle c_i,c_j,\ldots,c_k\rangle$是$\sigma$的一个平台基本软件配置.

例如,对于配置$\sigma=\langle c_1,c_2,\ldots,c_m\rangle$,$\langle c_i\rangle(1\leqslant i\leqslant n)$是$\sigma$的基本配置;由于 $c_1 \xrightarrow{\sigma} c_n$,因此$\langle c_1,c_n\rangle$也是σ的一个基本配置.平台基本软件配置与软件配置不同基,平台基本软件配置是一个有序集合.如果去掉平台基本软件配置的有序性,则有$\tau\subset\sigma$.

属性 $p_{soft\text{-}configration}$ 由终端运行环境中满足一定可信策略需要的基本软件配置提供,验证代理将证明代理获得的平台基本软件配置转换成对应的属性 $p_{soft\text{-}configration}$.值的注意的是,属性 $p_{soft\text{-}configration}$ 与平台的绑定不是直接到单个软件,而是绑定到每个平台基本软件配置上,因为许多属性是由几个软件按照一定顺序运行才提供的.比如,SELinux 提供 MLS 支持需要完整的 Linux 内核基础上装载 LSM 钩子模块,然后再装载 MLS 模块才能使得配置具有 MLS 能力.因此说,平台基本软件配置⟨*kernel*,*LSM*,*MLS*⟩提供 MLS 属性.

定义 10. *Draw*(*Processes*)表示从平台的 *Processes* 列表中获取平台基本软件配置.

定义 11. *Search-property*(*p*)表示从可信策略库中查询属性 *p* 对应的平台基本软件配置.

定义 12. 软件配置验证函数 *Verify_softwareconfiguration*(τ_1,τ_2)的功能是验证平台基本配置τ_1,τ_2 是否匹配:如果匹配,返回 ture;否则,返回 false.

根据以上定义,我们得出属性 $p_{soft\text{-}configration}$ 的判定算法,如图 4 所示.

图 4 算法首先判断终端平台的已启动进程列表是不是空,如不是空,就从该进程列表中获取平台基本软件配置,将此基本软件配置与可信策略库中需要的基本软件配置进行比较:如果匹配,则输出为 $p_{soft\text{-}configration}$,表示终端运行环境满足一定可信策略需要的软件配置属性;否则,输出 *NULL*.

算法 2. 属性 $P_{software\text{-}configuration}$ 的判定算法 *Check-softewareconfigration*(*Processes*).
输入:*Processes*;
输出:$P_{software\text{-}configuration}$ 或 *NULL*.
1. IF (*Processes*==∅) THEN return *NULL* ENDIF
2. α_1=*Draw*(*Processes*)
3. IF (α_1==∅) THEN return *NULL* ENDIF
4. α_2=*Search-property*($p_{soft\text{-}configration}$)
5. IF (α_2==∅) THEN return *NULL* ENDIF
6. IF (*Verify_softwareconfiguration*(α_1,α_2)) THEN
7. RETURN $p_{soft\text{-}configration}$;
8. ELSE
9. RETURN *NULL*;
10. ENDIF

Fig.4 Decidability algorithm of $P_{software\text{-}configuration}$
图 4 属性 $P_{software\text{-}configuration}$ 的判定算法

3.3 属性$p_{behaviour}$的分析和判定

属性 $p_{behavior}$ 的实质含义,是指终端运行环境中的主体行为满足一定可信策略.为了详细描述主体行为属性,我们首先定义与主体行为相关的基本概念.

定义 13(基本变量). S 为主体集合,也可看作是用户和用户启动的进程集合,在此,把已启动的进程称为活动主体;O 为客体集合,对于单个客体 o,有 $o\in O$,客体可以是文件、程序、设备等资源.

定义 14(行为集合). 行为集 $B=(S\times O\times A),\forall b\in B$ 表示主体对客体的操作行为.$S=\{s_1,s_2,\ldots,s_m,\ldots\}$表示该状态下所有行为主体集合.在终端系统中,主体包括用户和进程,用户启动程序后产生进程,进程代表用户执行访问操作,$O=\{o_1,o_2,\ldots,o_m,\ldots\}$表示行为的客体集合,$A=\{r,w,e\}$表示访问操作集合,包括读、写和执行.

定义 15. 恶意行为特征是指破坏终端运行环境的主体行为表现出来的特征,用 r_i 表示,其中,i 是正整数;恶意行为特征集是所有恶意行为特征的集合,记为 $R=\{r_1,r_2,\ldots,r_m,\ldots\}$,其中,$m$ 是正整数.

对于确定的终端运行环境,恶意行为普遍存在共同属性,无论是内部威胁、恶意代码和还是病毒,最终表现出来的恶意行为特征可以概括为自传播性(感染文件、自我复制等)、自激活性(注册启动项、文件关联)、自保护性(隐藏目录、守护进程)、破坏性(篡改破坏文件、破坏主机相关设备等)、非法连接性(非法连接并发起网络攻击等).这里,我们可以用 r_1 表示自传播性,可以用 r_2 表示自激活性,可以用 r_3 表示自保护性,可以用 r_4 表示破坏性,可以用 r_5 表示非法连接性,则终端运行环境恶意行为特征 $R=\{r_1,r_2,r_3,r_4,r_5\}$.

定义 16. 行为记录函数 *RecordAct*(s_i,x,o,t)表示在 t 时刻将主体 s_i 对 o 执行的动作 x 记录到系统日志 *log* 中.

定义 17. *Get-Badbehavior*(s_i,*log*)表示从系统日志 *log* 中获取平台主体 s_i 的行为集.

定义 18. $h(s_i,x,o)$表示根据主体 s_i 的行为$\langle s_i,x,o\rangle$计算恶意行为指数.

用 $h=\{h_{r_1}(s_i,x,o),h_{r_2}(s_i,x,o),h_{r_3}(s_i,x,o),h_{r_4}(s_i,x,o),h_{r_5}(s_i,x,o)\}$ 表示主体 s_i 的行为$\langle s_i,x,o\rangle$对终端运行环境的恶意影响,其中,$h_{r_1}(s_i,x,o),h_{r_2}(s_i,x,o),h_{r_3}(s_i,x,o),h_{r_4}(s_i,x,o)$ 和 $h_{r_5}(s_i,x,o)$ 分别表主体行为$\langle s_i,x,o\rangle$体的自传播指数、自激活指数、自保护指数、破坏指数以及非法连接指数.值为 0 表示对系统无影响;值越大,表示影响越大.用α,β,χ,δ和γ分别表示自传播指数、自激活指数、自保护指数、破坏指数以及非法连接指数在计算主体行为恶意指数中所体现的权重,则主体行为$\langle s_i,x,o\rangle$的恶意行为指数为

$$h(s_i,x,o)=\alpha h_{r_1}(s_i,x,o)+\beta h_{r_2}(s_i,x,o)+\chi h_{r_3}(s_i,x,o)+\delta h_{r_4}(s_i,x,o)+\gamma h_{r_5}(s_i,x,o).$$

当终端运行环境中的任意主体 s_i 的恶意行为指数超过设定的安全阈值 H 时,终端运行环境就不满足属性 $p_{behavior}$.根据以上定义,我们得出属性 $p_{behavior}$ 的判定算法,如图 5 所示.

图 5 算法首先判断终端平台的行为日志是否为空,如果不为空,调用 *Get-Badbehavior* 函数从系统行为日志 *log* 中获取平台主体 s_i 的行为集,并计算其恶意行为指数,如果没有超过安全阈值,则输出为 $p_{behavior}$,表示终端运行环境中的主体行为满足一定可信策略的需要;否则,输出 *NULL*.

算法 3. 属性 $p_{behavior}$ 的判定算法 *Check-behavior*(*log*,*H*).
输入:*log*, *H*;
输出:$p_{behaviorn}$ 或 *NULL*.
1. IF (*log*==∅) THEN return *NULL* ENDIF
2. IF (*Get-Badbehavior*(*s*,*log*)==∅) return *NULL*
3. ELSE
4. FOR *i*=1 to *m* DO //*m* 表示终端环境的进程数
5. FOR *j*=1 to |*Get-Badbehavior*(s_i,*log*)| Do
6. *h*=*h*(s_i,x_j,*o*)
7. IF *h*⩾*H* THEN return *NULL*
8. ENDFOR
9. ENDFOR
10. RETURN $p_{behavior}$
11. ENDIF

Fig.5 Decidability algorithm of $p_{behaviour}$

图 5 属性 $p_{behaviour}$ 的判定算法

3.4 终端运行环境远程证明的综合判定策略

在终端运行环境的远程证明方案中,存在一个被所有其他计算实体信任并且自身安全能够得到保证的属性证书颁发中心 TPA(trusted property authority),它的主要作用是发布与可信策略相对应的属性证书.

定义 19. 可信属性证书是一种轻量级的数字证书,这种数字证书不包括公钥信息,只包含证书所有人 ID、发行证书 ID、签名算法、有效期以及可信属性 p_{tpm},$p_{soft\text{-}configration}$,$p_{behavior}$,它是由属性证书颁发中心 TPA 签发的标志实体属性信息的一系列数据的集合.

在某一可信安全域中,可信属性证书是获得服务的主要凭证和相关授权的主要凭证.

定义 20. 属性证书颁发函数 *Cert*(*p*)的功能是属性证书颁发中心为属性 *p* 颁发属性证书.

综合判定策略:如果终端运行环境具有属性(p_{tpm},$p_{soft\text{-}configration}$,$p_{behavior}$),属性证书颁发中心 TPA 向终端颁发满足属性(p_{tpm},$p_{soft\text{-}configration}$,$p_{behavior}$)的属性证书,即:

$$Cert((p_{tpm},p_{soft\text{-}configration},p_{behavior})),$$

则终端运行环境是可信的.

根据算法 1~算法 3 定义的函数=*Check-trusted Platform Specification*,*Check-softewareconfigration* 和 *Check-behavior*,我们得出终端运行环境是否可信的综合判定算法,如图 6 所示.

算法 4. 终端运行环境可信综合判定策略算法 *check-whole*().
输入:I_t, *Processes*, *log*, *H*;
输出:*Property-Certificate* 或 *NULL*.
1. *P*1=*Check-trusted Platform Specification*(I_t)
2. IF (*P*1==p_{tpm}) THEN
3. *P*2=*Check-softewareconfigration*(*Processes*)
4. IF (*P*1==$p_{soft\text{-}configration}$) THEN
5. *P*3=*Check-behavior*(*log*,*H*)
6. IF (*P*3==$p_{behavior}$) THEN
7. *Property-Certificate*=*Cert*((p_{tpm},$p_{soft\text{-}configration}$,$p_{behavior}$))
8. ELSE
9. return *NULL*
10. ENDIF
11. ELSE
12. return *NULL*
13. ENDIF
14. ELSE
15. return *NULL*
16. ENDIF

Fig.6 Combined decidability algorithm

图 6 综合判定算法

图 6 算法首先判断终端平台是否满足属性 p_{tpm},然后判断是否满足属性 $p_{soft\text{-}configration}$ 和属性 $p_{behavior}$.如果这几个属性都满足,则调用 Cert 为该终端颁发可信属性证 Property-Certificate,表示终端运行环境是可信的(基于可信策略);否则,输出 *NULL*.

4 终端运行环境的远程证明方案在 Windows 平台上的实现

终端运行环境的远程证明方案中,证明代理与验证代理是两个核心实体.本节主要介绍证明代理与验证代理在 Windows 平台上的实现.

4.1 证明代理的设计与实现

4.1.1 证明代理的需求规定

终端运行环境的远程证明方案中,证明代理运行在可信终端,通过接受来自验证代理的命令对终端动态运行环境里 PCR 值、软件配置信息以及用户行为信息进行收集,为验证方依据可信策略库来评估终端运行时环境提供依据.因此,证明代理应该具有如下的功能需求和非功能需求.

功能需求包括:

- 协议管理.验证发送命令,证明代理解析协议并完成相应的功能;
- 信息收集.主要是收集终端运行环境里 *PCR* 值、软件配置信息以及用户行为信息;
- 信息签名与加密.代理需要对信息签名,保证来源可信;也需要对信息加密,保证传输可信.

非功能需求包括:

- 可信保障机制,保障代理服务端静态可信和动态可信.

4.1.2 证明代理的体系结构

在 Windows XP 平台上,我们用 Jtpm 模拟 TPM(Jtpm 是一个具有 TPM 功能的中间件平台),开发了终端运行环境远程证明方案中的证明代理.证明代理是一个 deamon 服务程序,没有用户界面.服务程序的体系结构如图 7 所示.当终端启动时,证明代理自动启动.证明代理通过读取配置文件和相关本地文件进行一系列的初始化工作,并启动证明代理线程.证明代理在和验证代理建立连接时,我们并没有选择典型的一个 Socket 主线程开多个子线程的服务器模型,而采用 Socket 非阻塞模式的 Select 模型来将证明代理服务器设计成只有一个线程、而该线程中包含了多条连接的服务器模型.当证明代理监听到验证代理的连接请求时,建立连接,并创建新的套接字对象.建立连接完成,一方面,证明代理接收来自验证代理转发的指令,调用指令实施模块进行指令解析、存储、转发或执行等功能,在辅助文件和 TPM 的帮助下进行 PCR 值、终端进程列表以及用户行为的收集;另一方面,证明代理将收集的信息存储于本地,并反馈信息给验证代理.

TPM 及证明代理之间的关系如图 8 所示.由图 8 可以看出,证明代理与 TPM 之间由 TSS 链接.TSS 由 3 个逻辑组件构成:TCG 设备驱动程序库(TDDL)、TCG 核心服务(TCS)、TCG 服务提供者(TSP).

- TDDL 为 TPM 定义了标准接口(TDDLI)以及提供了用户模式和内核模式之间的转换;
- TCS 提供了标准接口(TCSI)以及管理 TPM 的资源;
- TSP 为应用程序提供了丰富的面向对象的接口(TSPI).

证明代理通过 TSP 对 TPM 进行访问.TSP 通过接口 TSPI 接收来自收集代理的命令参数,TSP 做了相应的操作、处理后,将命令进一步封装成包,通过 TCSI 交给 TCS;TCS 做了相应的操作、处理后,将包转化成 TPM 能够识别的字节流,经由 TDDL 和 TDD 交送给 TPM.TDD 主要负责接收 TDDL 发送过来的字节流并且将其转发给 TPM,待 TPM 处理完后再将信息传回.

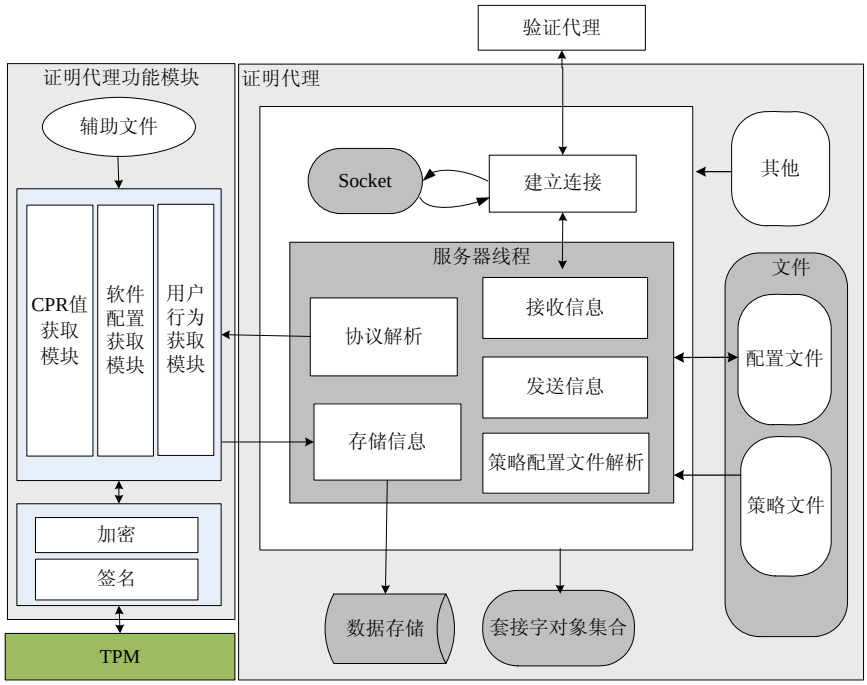

Fig.7 Architecture of server of a trusted attestation agent

图 7 可信代理服务端的模块结构

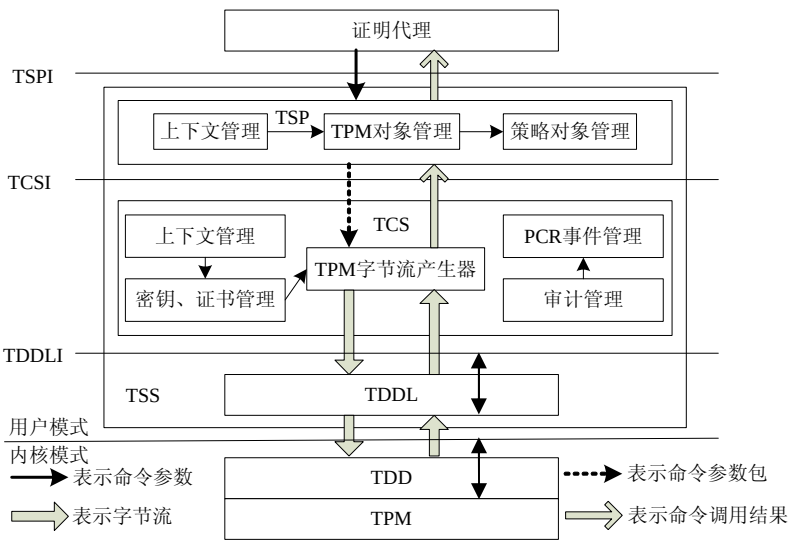

Fig.8 Mechanism of TPM and attestation agent

图 8 TPM 及证明代理之间的关系

4.2 验证代理的设计与实现

4.2.1 验证代理的需求规定

终端运行环境的远程证明方案中,验证代理是一个运行在网络中的应用程序,通常和服务提供方处于同一

个安全域.验证代理通过向证明代理发送命令,包括 PCR 值获取命令、软件配置获取命令以及用户行为获取命令等来获取证明代理所在终端的相关信息,并通过可信策略库对此类信息进行分析和处理.如果终端运行环境满足可信策略库确定的策略,验证代理需要向属性证书办法中心申请属性证书,并转发给证明代理所在的可信终端.因此,终端运行环境的远程证明方案中,验证代理应该具有如下主要功能需求,包括:

- 命令管理.验证代理向证明代理发送命令,收集相关信息;
- 信息接收与处理.主要接收来自证明代理的 PCR 值、软件配置信息以及用户行为信息,并对这些信息进行处理;
- PCR 属性验证.根据可信策略库中的 PCR 属性描述对终端 PCR 值进行验证;
- 软件配置验证.根据可信策略库中的软件配置描述对终端软件配置进行验证;
- 用户行为验证.根据可信策略库中的恶意行为描述对终端用户行为验证进行验证;
- 综合判定.综合判定终端运行环境是否可信.

4.2.2 验证代理的体系结构

如图 9 所示为终端运行环境的远程证明方案中验证代理的模块结构,从验证代理的体系结构可以看出,该验证代理主要包含建立连接模块和验证代理主线程模块.建立连接模块的作用与证明建立连接模块的作用相同.由于一个验证代理需要收集证明代理所在终端的多项内容,因此验证代理使用了多线程,主线程可以生成多个子线程,一个子线程收集证明代理所在终端的一类信息.验证代理子线程包含发送命令模块、接收信息与处理模块、PCR 属性验证模块、软件配置验证模块、用户行为验证模块、综合判定模块、协议解析模块以及属性证书申请模块.命令管理模块的功能是发送命令信息给证明代理;信息接收和处理模块的功能是接收证明代理发送来的信息,并对信息进行处理;PCR 属性验证模块的功能是根据可信策略库中的 PCR 属性描述对终端 PCR 值进行验证;软件配置验证模块的功能是根据可信策略库中的软件配置描述对终端软件配置进行验证;用户行为验证模块的功能是根据可信策略库中的恶意行为描述对终端用户行为验证进行验证;综合判定模块的功能是综合判定终端运行环境是否可信;协议解析模块的功能是处理证明代理与验证代理之间的通信工作;属性证书申请模块的功能是申请属性证书.

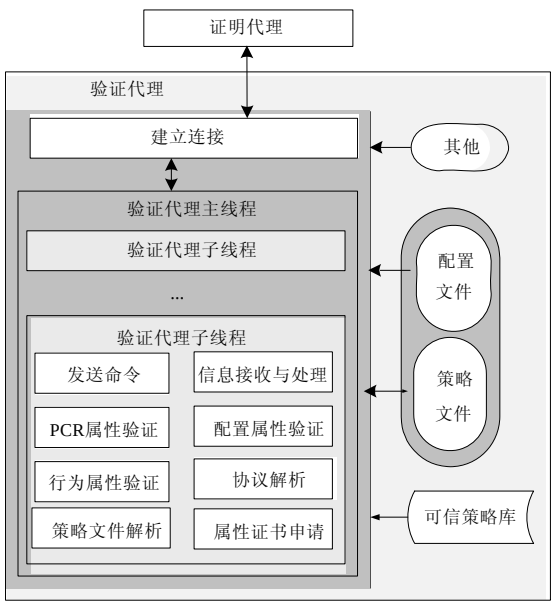

Fig.9 Architecture of verificaiton agent

图 9 验证代理模块结构

为了便于呈现验证代理的验证结果,我们通过 Web 服务器调用验证代理的相关接口.用户可以通过浏览器直观看出验证代理的相关验证结果.

验证代理结果呈现的运行界面如下图 10 所示.

Fig.10 Property-Based verification system of the trusted terminal

图 10 可信终端运行环境属性验证系统

4.3 证明代理与验证代理通信协议的设计

在证明代理和验证代理共同约定了一种通信协议,使得它们在交换信息时遵循统一的规则.该协议的具体包格式如图 11 所示,该协议包只包含 Type 和数据段两段:.

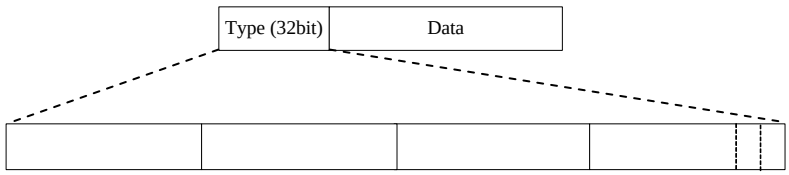

Fig.11 Communication protocol between attestation agent and verification agent

图 11 证明代理与验证代理的通信协议

Type:表示的是验证代理所选择的指令.由于证明代理需要获取 PCR 值、软件配置和用户行为信息,因此 Type 段的设计如图 12 所示.即 Type 共 32 位、4 个字节,高 30 位保留,用于证明代理日后的功能扩展.用低 4 位比特的二进制来表示不同的指令,见表 1.

Data:用来填充与协议相关的数据.当证明代理要发送一个指令的相应反馈结果时,就将其填充到 data 字段.例如获取 PCR 信息时,证明代理必须将终端的 PCR 值发送给验证代理.

Table 1 Kind of communication protocol

表 1 通信协议种类

消息标识	操作描述
00	接收命令准备
01	获取 PCR 信息
10	获取软件配置信息
11	获取用户行为信息

5 终端运行环境的远程证明的应用案例研究

本节以一个开放的局域网为实验环境分析本文提出的终端运行环境的远程证明方案.在该实验环境中,我们选用 4 台计算机,其中,一台普通 PC,基本配置为 Windows vista home basic @2007 service pack 1,CPU: Intel(R)Core(TM)2 Duo CPU T6670 @2.20GHz,1.0GB RAM,用于部署证明代理,该机器上配置 Jtpm;第 2 台为服务器,基本配置为 Windows XP Professional 2002 Service pack 3,CPU:Intel(R)Core(TM)2 Duo CPU E7500@2.93GHz,2.0GB RAM,用于部署验证代理;第 3 台为服务器,基本配置为:Ubanq2,CPU:Intel(R)Core(TM)2 Duo CPU E7500@2.93GHz,2.0GB RAM,用于部署策略数据库和小型开源 CA,最后一台也为服务器,基本配置为:Ubanq2,CPU:Intel(R)Core(TM)2 Duo CPU E7500@2.93GHz,2.0GB RAM,用于部署 Web 服务.本实验的基本思路是:终端在访问 Web 服务之前,需要向 Web 服务器证明自己满足访问 Web 服务器的可信策略.本实验也可以应用于实现网络可信接入等这类开放的分布式计算环境中.

5.1 证明代理的设计与实现

终端是满足可信计算平台规范的终端.在访问某一服务时,该服务必须满足可信策略库中的访问策略,包括满足可信计算平台规范策略、软件配置策略(包括 Linux 本地安全服务已运行、Linux 安全补丁已安装、Linux 防火墙已开启等等)以及终端用户行为可信策略,该可信策略是服务访问决策点判定终端是否可以访问服务时的判定依据.需要注意的是,本节给出的可信策略表示了访问服务时服务管理员对于网络安全要求的定义,根据安全要求不同,可以定义各类可信策略实例.图 12 给出了在终端系统中满足可信策略的实体图,其中,信任根是系统符合可信计算平台规范.

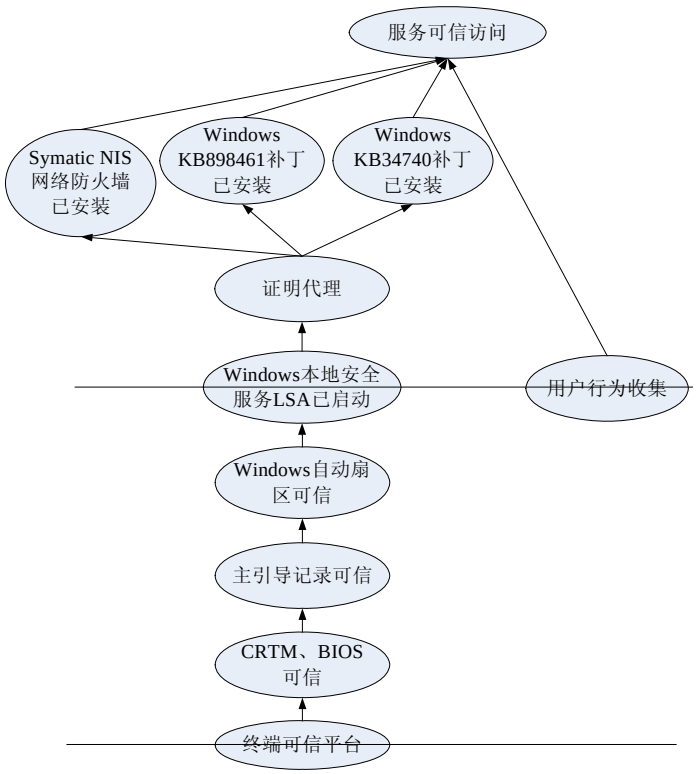

Fig.12 State diagram for the trusted service access on the trusted terminal

图 12 可信服务访问的终端运行环境的状态图

该实体图描述了在服务访问点处终端的硬件、固件以及软件属性的定义以及属性间的信任关系和用户行为的信任关系,例如根据可信策略,可以看出,只有当系统同时安装有操作系统补丁、运行防火墙软件,并且用户无恶意行为时,才可以认为该系统处于安全状态,可以访问该服务.

图 13 给出了根据一次服务访问时可信策略,利用评估过程得出的对于平台满足各个安全属性的验证结果.

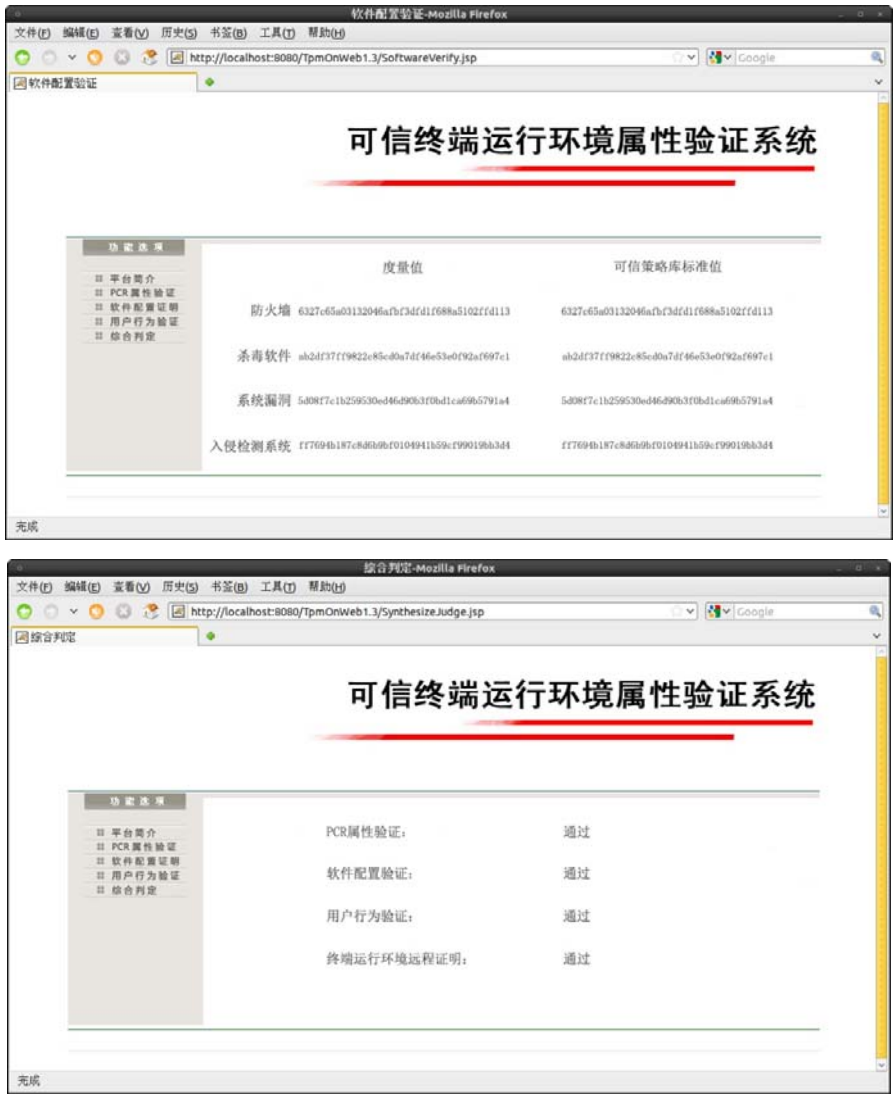

Fig.13 Property verification by our project for the trusted service access on the trusted terminal

图 13 可信服务访问的终端运行环境属性验证

根据服务访问系统中可信策略定义可以看出:传统二进制远程证明过程在能力上无法满足此类远程证明要求,例如对于图 13 中定义的信任关系,当系统同时安装有操作系统补丁、并且运行防火墙软件时,可以认为该系统处于安全状态;对于已有的属性证明方案,在能力上无法满足终端用户行为可信属性证明的要求,例如对于图 13 中定义的信任关系,当系统中的用户无恶意行为时,则可以认为该系统处于安全状态.通过本文提出的用户行为属性判定策略,可以证明终端运行环境的用户行为可信.

5.2 证明代理在终端中的性能分析

证明代理在一定程度上会影响终端的性能,我们将从启动时间、CPU 负载、内存消耗等方面对此进行详细的讨论.而验证代理运行在第三方,对终端运行环境和服务访问的影响很小,本文不予讨论.

图 14 所示为终端在使用了证明代理和没有使用证明代理情况下的启动时间对比(深色为没有使用可信证据收集代理,浅色为使用的情况),其中,代理文件的大小为 1.28MB,这里的启动时间是指终端的证明代理开始执行到加载完毕开始执行的时间间隔.在一次系统启动之后,我们连续 4 次启动证明代理,这 4 次执行的时间如图 14 所示,其中,横坐标为启动次数,纵坐标为执行时间(单位:s).实验是在如下的环境中进行的:Windows XP Professional 2002 Service pack 3,CPU:Intel(R)Core×2 Duo CPU E7500@2.93GHz,2.0GB RAM.从实验结果可以看出:在使用了证明代理的情况下,代理程序第 1 次启动的时间显著地增加了,这部分增加的时间是用于完成对代理文件的度量;在第 2 次启动开始,由于代理文件没有发生改变,而且度量值是直接在高速缓存中获得的,所以之后的启动时间与没有使用可信证据收集代理时的启动时间是基本相同的.这个结果与文献[17]相似.

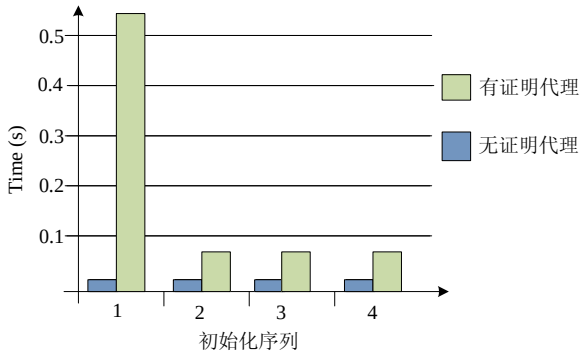

Fig.14 Initiation time for server of the trust attestation agent

图 14 可信代理服务端启动时间

为了测试终端 CPU 负载,我们首先关掉终端所有无关的应用进程,待其稳定后测试 CPU 的利用率.如图 15 所示,图中显示的是在 4 个不同的时间点 CPU 的利用率(间隔 5min).从该图可以看出,关闭所有应用程序,此时 CPU 值在 0~2%之间浮动,CPU 的利用率几乎为 0.

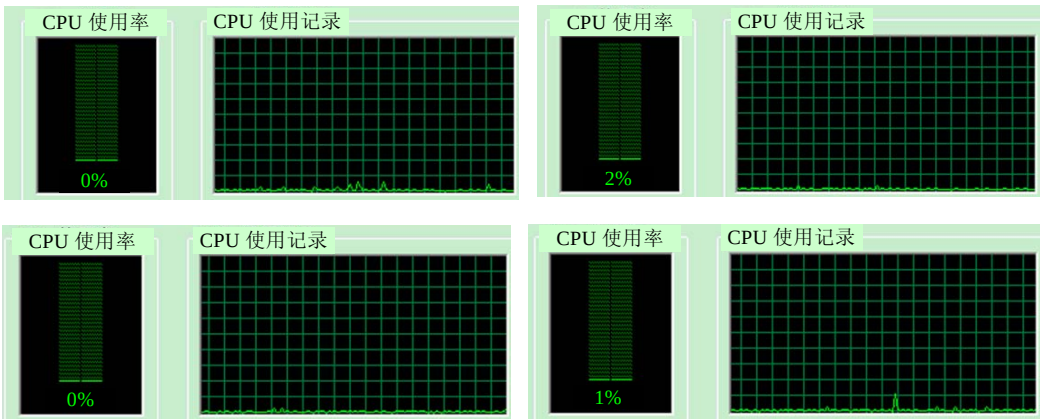

Fig.15 CPU performance of terminal when server of the trusted attestation agent shuts down

图 15 未开启可信代理服务端时终端 CPU 的性能

然后启动证明代理,待其稳定后再测试 CUP 的利用率.如图 16 所示,图中显示的是在 4 个不同的时间点

CPU 的利用率(间隔 5min).从该图可以看出,启动监控代理后 CPU 的利用率变动范围在 3%~6%,常在 3%~6%之间波动.也就是说,收集代理给 CPU 带来低于 6%的开销.

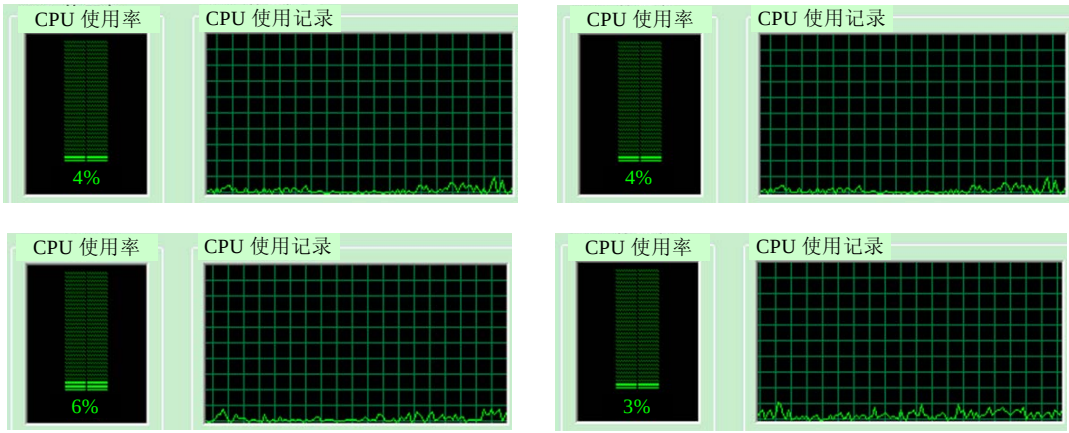

Fig.16 CPU performance of terminal when server runs

图 16 开启代理服务端时终端 CPU 的性能

为了测试内存消耗,我们首先关掉终端所有无关的应用进程,测试内存占用率,如图 17 所示;然后启动证明代理,再次测试内存占用率.从前后两个图可以直观看出:证明代理带来的内存开销为 7M,非常小,对终端内存的使用几乎没有影响.

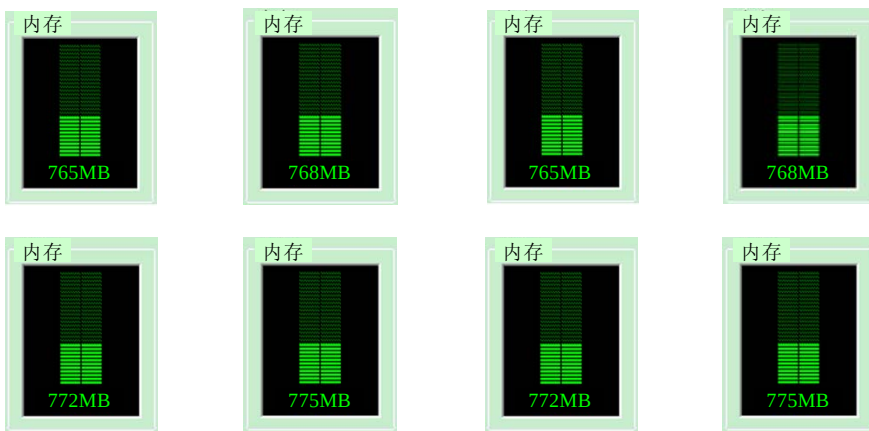

Fig.17 Memory expense of terminal when sever of the trusted attestation agent runs and shuts down

图 17 未开启和开启可信代理服务端时终端内存开销

6 比较与评价

目前,为了促进远程证明方案的实用化,众多研究机构和学者提出了许多不同的远程证明方法.从遵循 TCG 规范的直接二进制证明,到基于高级语言的语义证明;从嵌入式设备的基于软件证明,到 Web Service 的证明.在这众多的证明方法中,研究成果最多的还是基于实体标识的二进制证明和基于属性的远程证明.本方案与已有基于实体标识的二进制证明和基于属性的远程证明方案相比,具有如下特点:

(1) 与已有的基于实体标识的二进制证明相比,本方案克服了基于实体标识的二进制证明的最大缺点:对平台配置隐私的暴露,证明过程中不再要求出示整个平台的配置的完整性度量值;

(2) 与已有的基于属性的远程证明方案相比,本方案不仅包括了满足可信计算平台规范的属性判定和软件配置的属性判定,而且还包括终端用户行为的属性判定.因此,本方案不仅可以判断终端的静态环境可信,还可以判定终端的动态环境是否可信,为服务访问或网络接入提供更高的可信保障;

(3) 本方案在 Windows 平台上实现了证明代理和验证代理,并进行了实际的案例研究,最后分析了证明代理在终端的性能开销.方案完整详细,具有较高的参考价值.

7 总 结

本文首先提出了终端运行环境远程证明的总体方案,分别介绍了属性 p_{tpm}、属性 $p_{soft\text{-}configration}$ 和属性 $p_{behavior}$ 的判定策略以及终端运行环境远程证明的综合判定策略.然后,详细介绍了终端运行环境的远程证明方案在 Windows 平台上的实现,包括证明代理与验证代理的设计与实现以及它们之间的通信协议;其次,我们将证明代理与验证代理应用于可信服务访问,并对证明代理在终端中的性能分析;最后,将本方案与已有基于实体标识的二进制证明和基于属性的远程证明方案进行了比较.

我们下一步的工作将在两个方面展开:一是开发本方案的实际产品适用系统;二是寻找更优化的终端运行环境远程证明方案.

References:

[1] Trusted Computing Group. TPM main specification, version 1.2. 2003. http://www.trustedcomputinggroup.org

[2] Functionality and Interface Specification of Cryptographic Support Platform for Trusted Computing. 2007 (in Chinese).

[3] Garfinkel T, Pfaff B, Chow J. Terra: A virtual machine-based platform for trusted computing. In: Proc. of the 19th ACM Symp. on Operating Systems Principles. 2003. [doi: 10.1145/945445.945464]

[4] Sailer R, Zhang X, Jaeger T. Design and implementation of a TCG-based integrity measurement architecture. In: Proc. of the 13th Conf. on USENIX Security Symp. 2004.

[5] Trent T, Sailer R, Shanker U. PRIMA: Policy-Reduced integrity measurement architecture. In: Proc. of the 11th ACM Symp. on Access Control Models and Technologies. 2006. [doi: 10.1145/1133058.1133063]

[6] Peinado M, Chen Y, England P. NGSCB: A trusted open system. In: Proc. of the Australasian Conf. on Information Security and Privacy. 2004. [doi: 10.1007/978-3-540-27800-9_8]

[7] Shi E, Adrian P, Doom LV. BIND: A fine-grained attestation service for secure distributed systems. In: Proc. of the 2005 IEEE Symp. on Security and Privacy. 2005. [doi: 10.1109/SP.2005.4]

[8] Kuhn U, Selhorst M, Stuble C. Realizing property-based attestation and sealing with commonly available hard-and software. In: Proc. of the 2007 ACM Workshop on Scalable Trusted Computing. 2007. [doi: 10.1145/1314354.1314368]

[9] Chen L, Landfermann R, Lohr H. A protocol for property-based attestation. In: Proc. of the 1st ACM Workshop on Scalable Trusted Computing. 2006. [doi: 10.1145/1179474.1179479]

[10] Sadeghi AR, Stubl C. Property-Based attestation for computing platforms: Caring about properties, not mechanisms. In: Proc. of the 2004 Workshop on New Security Paradigms. 2004. [doi: 10.1145/1065907.1066038]

[11] Poritz J, Schunter M, Herreweghen EV. Property attestation-scalable and privacy-friendly security assessment of peer computers. Technical Report, RZ 3548, IBM Research, 2004.

[12] Sheehy J, Coker G, Guttman J. Attestation: Evidence and trust MITRE. Technical Report, MTR080072, 2008.

[13] Yu AM, Feng DG, Wang D. Property-Based remote attestation model. Journal on Communications, 2010,31(8):1−8 (in Chinese with English abstract).

[14] Cui YL, Shen CX. Credibility attestation of integrity measurement in property remote attestation. Computer Engineering, 2010, 36(21):11−16 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-3428.2010.21.004]

[15] Gu L, Guo Y, Wang H, Zou YZ, Xie B, Shao WZ. Runtime software trustworthiness evidence collection mechanism based on TPM. Ruan Jian Xue Bao/Journal of Software, 2010,21(2):373−387 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3789.htm [doi: 10.3724/SP.J.1001.2010.03789]

[16] Brickell E, Camenisch J, Chen LQ. Direct anonymous attestation. In: Proc. of the 11th ACM Conf. on Computer and Communications Security. 2004. 132−145.

[17] Ge H, Tate SR. A direct anonymous attestation scheme for embedded devices. In: Proc. of the Public Key Cryptography (PKC 2007). 2007. 16−30. [doi: 10.1007/978-3-540-71677-8_2]

[18] Brickell E, Chen LQ, Li JT. Simplified security notions for direct anonymous attestation and a concrete scheme from pairings. Int'l Journal of Information Security, 2009,8(5):315−330. [doi: 10.1007/s10207-009-0076-3]

[19] Brickell E, Chen LQ, Li JT. A new direct anonymous attestation scheme from bilinear maps. In: Proc. of the Trusted Computing-Challenges and Applications (TRUST 2008). 2008. 166−178. [doi: 10.1007/978-3-540-68979-9_13]

[20] Chen LQ, Morrissey P, Smart NP. Pairings in trusted computing. In: Proc. of the Pairings in Cryptography (Pairing 2008). London: University of London, 2008. 1−17. [doi: 10.1007/978-3-540-85538-5_1]

[21] Chen LQ, Morrissey P, Smart NP. DAA: Fixing the pairing based protocols. Cryptology ePrint Archive, Report 2009/198, 2009. http://eprint.iacr.org/2009/198

[22] Chen LQ, Li JT. A note on the Chen-Morrissey-Smart direct anonymous attestation scheme. Information Processing Letters, 2010, 110(12):485−488. [doi: 10.1016/j.ipl.2010.04.017]

[23] Chen X, Feng D. Direct anonymous attestation for next generation TPM. Journal of Computers, 2008,3(12):43−50.

[24] Brickell E, Li JT. Enhanced privacy ID from bilinear pairing. Cryptology ePrint Archive, Report 2009/095, 2009. http://eprint.iacr.org/2009/09

[25] Chen LQ. A DAA scheme requiring less TPM resources. In: Proc. of the 5th China Int'l Conf. on Information Security and Cryptology. 2009. 350−365. [doi: 10.1007/978-3-642-16342-5_26]

[26] Brickell E, Li JT. A pairing-based DAA scheme further reducing TPM resources. In: Proc. of the 3rd Int'l Conf. on Trust and Trustworthy Computing. 2010. 181−195.

[27] Yacine G, Ahma-Reza S, Patrick S. Beyond secure channels. In: Proc. of the ACM Workshop on Scalable Trusted Computing. 2007. [doi: 10.1145/1314354.1314363]

附中文参考文献:

[2] 国家密码管理局.可信计算密码支撑平台功能与接口规范.2007.

[13] 于爱民,冯登国,汪丹.基于属性的远程证明模型.通信学报,2010,31(8):1−8.

[14] 崔艳莉,沈昌祥.属性远程证明中完整性测量的可信性证明.计算机工程,2010,36(21):11−16. [doi: 10.3969/j.issn.1000-3428.2010.21.004]

[15] 古亮,郭耀,王华等.基于 TPM 的运行时软件可信证据收集机制.软件学报,2010,21(2):373−387. http://www.jos.org.cn/1000-9825/3789.htm [doi: 10.3724/SP.J.1001.2010.03789]

谭良(1972－),男,四川泸县人,博士,教授,主要研究领域为可信计算,网络计算.
E-mail: jkxy_tl@sicnu.edu.cn

陈菊(1987－),女,硕士,主要研究领域为网络安全.
E-mail: ccchenjulin@163.com