

文章编号:1001-9081(2009)11-3001-04

# 远端非可信平台 Agent 完整性保护机制研究与设计

杨 翠, 谭成翔

(同济大学 计算机科学与技术系, 上海 201804)

(fionaletitgo@gmail.com)

**摘 要:**服务端采用 Agent 部署移动代码实现业务交互存在诸多安全问题。为提高软件的可信度,确保 Agent 在复杂运行环境中可靠运行,通过分析传统的完整性验证机制,借鉴身份认证、携证代码以及反射技术,提出对终端 Agent 进行完整性验证的分级保护机制,设计了互补的验证方案,实现软件行为的监控,从而提升了移动代码的可信度。

**关键词:**软件完整性验证;携证代码;反射技术;分级保护;可信软件

**中图分类号:** TP31; TP393.08 **文献标志码:** A

## Research and design of Agent integrity protection mechanism on remote untrusted platform

YANG Cui, TAN Cheng-xiang

(Department of Computer Science and Technology, Tongji University, Shanghai 201804, China)

**Abstract:** Plenty of security problems may occur when servers adopt Agent to deploy mobile codes so as to realize interactive storage between different business clients. In order to pursue a higher reliability of the software, and to make sure those Agents healthily running in an untrusted complex environment, after analyzing traditional integrity validating mechanism, combining I&A, PCC and reflection techniques, a new classified mechanism of enabling the integrity of trusted terminal Agents was proposed, and an efficient validating model with multiple interacting modules was designed, aiming at improving the reliability of the mobile codes by realizing its behaviors-monitoring.

**Key words:** software integrity protection; Proof-Carrying-Code (PCC); reflection; classified-protection; trusted software

## 0 引言

互联网的使用便于软件的分发与业务数据交换,同时也带来了严重的安全问题。大量服务提供商脱离了传统的货架式软件发布模式,选择通过网络分发具有一定业务处理能力的 Agent,降低大量业务交互给服务器带来的压力。Agent 以其便捷、高效的部署优势,被广泛采用,但该类软件的完整性保护问题也随之成为首要的安全问题之一。Agent 的运行平台不可信性与自身的复杂性都会使其完整性极易遭到破坏。本文对目前几种软件完整性保障机制作详细分析,比较各种方案的可行性,并在此基础上研究和设计一种新的基于远端非可信平台的 Agent 完整性保护机制,力求在一定程度上提高软件的可信度。

Agent 自身具有处理终端业务数据的能力,一方面有频繁访问终端系统资源、用户数据信息的权限,另一方面又与远程服务端进行通信交互,获取服务资源,协助服务器进行业务处理。由此提出运行于终端的 Agent 需要具有保护用户隐私数据以及服务提供商的重要资源不被非法访问和篡改等特点,基于一套合理的完整性保障机制来提高其自身的可信度。目前符合上述属性的应用场景如电子市场中外遣 Agent 携带银行支票、私钥等。

接下来通过对实际问题的描述以及各种解决方案的对比分析,本文将提出一种完整性验证机制,并给出具体的实施

方案。

## 1 问题描述

需要与服务提供商交互的软件涉及到的安全问题主要包括以下几个方面<sup>[1]</sup>:

1) 应用软件应保障自身的可靠性运行:软件不被异常终止;软件提供的基本服务能够正常进行;不会引起本地系统故障(如格式化磁盘、删除文件等攻击)。

2) 应用软件应保护用户敏感信息:不泄露重要机密数据;相应的服务权限不被盗取和冒用。

3) 保护局端主机不被恶意 Agent 攻击。

导致软件不可信的因素较多,由于软件不可避免地与操作系统平台发生交互,在程序初始化、程序在内存运行中都会暴露很多弱点,攻击者可以从一个简单的入口得到程序的控制权,甚至整个系统的控制权,从而能够修改软件可以控制的敏感信息或执行恶意系统操作等。

我们的关注点在于保护服务端,并且在不可信终端运行环境下 Agent 受到恶意篡改后能够及时停止与服务端交互。

本文涉及的主要实体定义如下:

**终端** 该终端部署有某服务商提供的分发式应用软件 Agent,该软件需要使自身的完整性得到保护,软件运行于不可信的操作系统环境。

**服务端** 该服务端部署有具体业务服务系统,并能够以

收稿日期:2009-04-29;修回日期:2009-06-03。

**作者简介:** 杨翠(1985-),女,内蒙古赤峰人,硕士研究生,主要研究方向:计算机网络、系统安全; 谭成翔(1965-),男,湖北红安人,研究员,博士生导师,主要研究方向:计算机网络安全、网络数据库。

某种通信协议与终端进行接入认证、Agent 完整性验证。

本文讨论的问题基于以下前提:

- 1) 服务端为服务提供商维护的可信环境;
- 2) 终端安装的 Agent 为可信版本,即软件分发过程已经完成,且保证安装过程可信。

## 2 现有解决方案简介和本文方案的提出

很多针对软件完整性保护问题的解决方案都在实际应用中暴露出或多或少的缺陷,以下将对这些方法进行详细分析对比,本文所提出的验证机制正是有机结合这些方案的优势后提出的。

### 2.1 身份识别与认证服务

保护终端应用软件不受未经授权修改可以被视为一个标准的访问控制问题,通过适合的身份识别与认证服务即可解决。身份识别与认证 (Identification and Authentication, I&A) 服务的两个主要过程为:首先进行身份识别,软件具有唯一标志,能够被局端识别为一个注册实体,并且进一步通过认证来确保该实体的合法性。为了实施对软件 Agent 的认证机制,可以制定一系列可选择的认证策略。能够作为认证信息的元素有以下几种:

- 1) 机密信息,指由用户掌握的授权信息,通常采用密码的形式;
- 2) 设备属性,指用户设备具备的特点,通常为机器硬件信息如 MAC 地址;
- 3) 生物特征,指用户自身具备的生物学特征;
- 4) 物理位置信息,指用户 IP 地址。

总之,唯一性与隐秘性是认证信息最根本也是最关键的特点。此种方案的主要缺陷在于,攻击者可以对应用软件实施反向工程,篡改或禁掉 I&A 模块功能,从而使认证机制失去意义。

### 2.2 代码校验

对终端运行程序的内存部分进行校验和计算,并周期性的发送回局端,与局端所保存的本地版本进行比对,以保障软件运行时的完整性。这一方案的主要弱点是,流氓软件可以编写伪校验和生成器,提前获取正确校验和,并保存其副本,响应局端的校验请求,绕过完整性检查<sup>[2]</sup>。

### 2.3 全代码传输

针对代码校验方案存在的问题,可以采用将全部代码压缩传输。由于攻击者很难在内存同时保留一份原始代码副本和一份修改后的代码段,此方案也有一定可行性。但传输数据量过大,占据较高带宽,会造成通信性能下降。

### 2.4 基于自反应机制的代码段动态校验<sup>[3]</sup>

通过对代码校验与全代码传输方案的改进,从随机性入手给出一种动态校验方案,即建立基于 RIPEMD-160<sup>[4]</sup> 的消息通信认证协议,通过局端和终端的 4 次认证信息交换来完成软件运行时的完整性验证。具体过程如图 1 所示,其中:代码段逻辑内存地址区间为  $0 \sim L$ ,其中  $0 < M_1 < M_2 < L$ ,  $M_1$ 、 $M_2$  由局端动态生成;  $H(S1, S2)$  表示局端请求获得终端代码段  $S1$  到  $S2$  之间的 RIPEMD-160 散列值;  $V, H(S1, S2)$  表示终端返回给局端 Agent 版本信息以及代码段  $S1$  到  $S2$  之间的 RIPEMD-160 散列值;  $V_1$  为局端本地备份的该 Agent 版本

信息。

由图 1 可知,该方案最大的优势在于:  $M_1$ 、 $M_2$  为随机数,两次交叉代码段的 MD 值正好覆盖全部程序代码特征。该方案的缺陷在于:该机制的建立前提在于确保攻击者无法获得代码段以外的内存空间(即散列函数无法探测到的部分)来压缩保存原始代码副本,从而采取及时解压技术来响应局端认证消息请求。

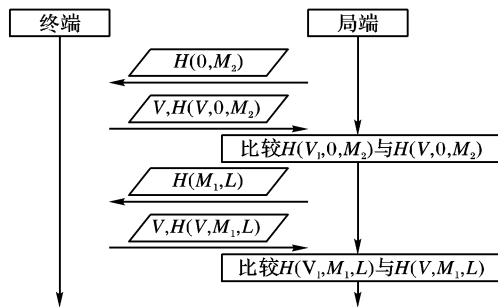


图 1 内存代码动态完整性验证流程

### 2.5 携证代码<sup>[5-6]</sup>

该技术可以用于保证软件分发过程的可信性,以及保护服务提供商局端不被恶意 Agent 攻击。携证代码 (Proof Carrying Code, PCC) 的理论模型为:欲执行代码的一方预先制定安全策略,代码提供者据此赋予代码一个安全证明 (proof) 以说明其遵守安全守则,再把该证明送至代码执行者携带的证明确认器;证明被确认为对的,则说明真的符合安全需求,方可将此代码交付执行。我们将在下面详细讨论对该方案的合理应用,有效运用于保护局端不受恶意 Agent 攻击,并提供给终端主机欲执行的 Agent 外存完整性证明。

### 2.6 本文的解决方案

由此可见,要确保 Agent 在非可信终端的完整性,必须考虑到两个阶段。

阶段 1 软件加载时与局端通信进行身份识别与认证,并发送外存代码完整性验证条件。

在此过程中,Agent 向局端发送认证三元组 (AgentID, CA, proof)。其中,AgentID 和 CA 是对 Agent 合法身份的认证信息,提供给局端作接入授权认证;proof 由 Agent 所带的代码完整性证明生成器生成,提供给局端作外存代码完整性验证。

具体实现步骤如下:

- 1) 采用证书 (CA) 的方式对 Agent 进行身份识别与认证;
- 2) 应用 PCC 方案并做适当改进来进行外存代码完整性验证。

要求终端与局端维护同样的安全策略集,可在分发过程中实现。

①Agent 将自带的安全策略集与外存二进制代码同时输入验证条件生成器 (Verification Conditions Generator, VCG),生成与程序语义和程序逻辑相关联的验证条件 (VC);

②Agent 的证明生成器 (prover) 对 VC 进一步处理生成原始完整性证明 (proof),并交由证明编码器 (LF encoder) 进行编码转换;

③发送编码转换后的完整性证明给局端,由局端依据同样的安全策略集利用证明检查器 (proof checker) 确认该证明为真,并返回确认信息给终端。

实施框架如图 2 所示。

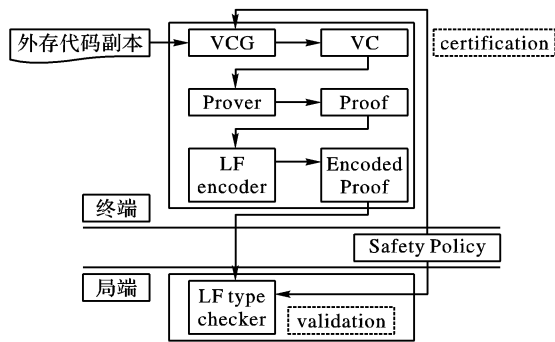


图 2 PCC 外存代码完整性验证框架

阶段 2 运行时的完整性验证。

基于前述代码段动态校验方案,采用自反应机制,通过与局端的消息通信进行动态代码校验,保证运行时内存代码不被恶意修改。

### 3 本文方案的主要优势

本文方案主要采用分级保护的思想,从身份认证、外存代码完整性验证以及运行时代码完整性验证逐层对 Agent 进行保护,保护级别逐级加强。

1) 身份认证从根据 Agent 用户角色授权其访问资源的角度,对服务端重要数据及业务服务进行保护,在未实施 Agent 行为监控之前先确保其身份的合法。

2) 由于不可信代码在执行前被静态验证,节省了执行时间。我们能够更早地探测到潜在的危险操作,因此避免了当代码使用者必须结束不可信进程时,该进程已经获得了部分资源或修改了系统状态。

3) 运行时的代码完整性验证能够保证 Agent 的行为可控,能够在与终端非可信运行环境交互最多、最容易被恶意代码攻击的阶段实时防范,发现危险即终止 Agent 与服务端的交互,同时进一步终止 Agent 在终端的运行。

## 4 方案实施

终端与局端体系结构如图 3 所示,以终端与局端软件体系结构的模块化形式表现 Agent 完整性保障机制的具体实施。

### 4.1 终端 Agent 软件

终端 Agent 软件主要包括:软件加载模块、运行时完整性验证模块、基本业务模块以及管理控制模块,图 3 中省去了网络通信模块,替换为具体通信内容。

1) 软件加载模块:完成 Agent 的身份认证与外存完整性验证。包括 PCC 代码完整性证明生成器与 Agent 身份认证信息寄存器。其中,代码完整性证明生成器由 VCG, Prover, LF Encoder 三个子模块构成,将外存代码经过三步处理生成编码后的验证证明 proof 回传给服务器进行验证。

2) 运行时完整性验证模块:完成 Agent 运行时内存代码的完整性验证。主体操作由 RIPEMD-160 生成器完成。该模块周期性的接收服务端发送的验证请求,根据随机数  $M_1$ 、 $M_2$  对内存代码生成相应 Hash 验证序列回传给服务器进行验证。

3) 基本业务模块:完成 Agent 与服务端进行交互的基本业务,获取业务数据进行相应处理。

4) 管理控制模块:实时控制 Agent 的起停。当接收到服务端确认失败信息时,立即中止 Agent 与服务端的交互,将数据缓存,或依终端安全策略停止 Agent 进程。

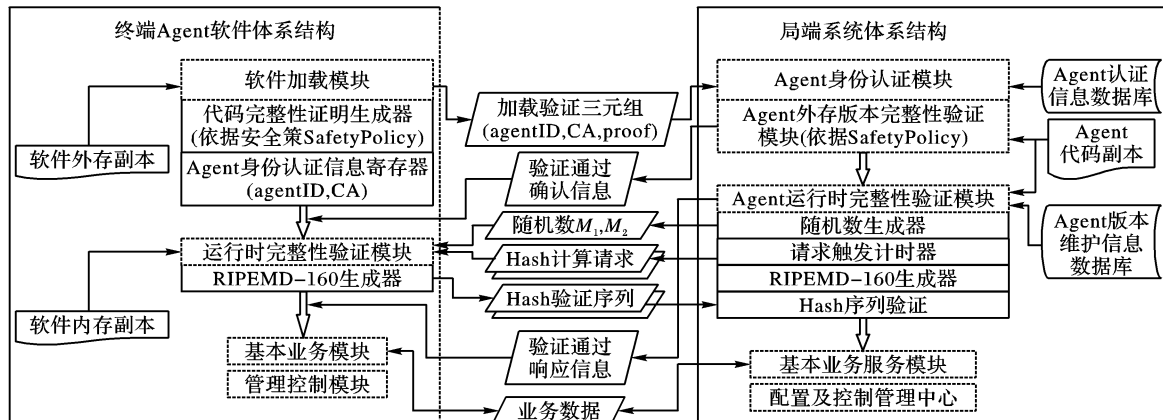


图 3 Agent 完整性保障机制体系结构

### 4.2 局端系统

1) Agent 身份认证模块:将接收到的 Agent 身份信息进行分析处理,查找 Agent 身份认证策略库,对请求接入的 Agent 分配合法的访问权限。

2) Agent 外存版本完整性验证模块:将接收到的 Agent 代码完整性证明进行快速简便的类型检查,判断其是否为真,并发送确认信息给终端。主要操作由证明确认器(proof checker)执行。

3) Agent 运行时完整性验证模块:完成 Agent 运行时的行为监控。包括随机数生成器,请求触发计时器,RIPEMD-160 生成器以及 Hash 序列验证模块四个主要子模块。由请求触发计时器周期性触发验证请求,该请求启动随机数生成器生

成两个随机数发往终端;接收到终端回传的 Agent 版本信息与 hash 序列后,RIPEMD-160 生成器根据随机数及本地 Agent 代码副本,进行分析计算,生成待比较的 hash 序列;Hash 序列验证模块将比对终端发送的 hash 序列与局端生成的 hash 序列是否相同,继而发送验证确认信息给终端。

4) 基本业务服务模块:通过两次完整性验证并具有访问资源权限的 Agent 将与该模块进行基本业务的交互。

5) 管理控制中心:该模块可配置 Agent 的安全访问规则,设置 Agent 保护等级。

## 5 待解决的问题

1) PCC 机制要求 Agent 的代码完整性证明生成器能够保

证可信;

2) 运行时完整性验证的消息机制经过多步握手操作,将导致一定的网络传输性能开销;

3) 升级 Agent 所带来的分发、维护等问题:在 PCC 机制中,要求局端和终端维护相同的安全策略集,这样在大量 Agent 做版本升级时,会增加一定难度。

以上问题为本文尚未解决的几个主要问题,也是该方案在实施过程中所遇到的一些实际性困难。针对第一个问题,可引入一系列已经被用于软件行为预测的技术,如沙盒技术给予解决<sup>[7]</sup>,将软件加载模块注入沙盒,尽可能确保其运行时可靠。

此外,以一定的性能开销作为保障安全的代价是不可避免的,因此,设计该系统时考虑到了分层保护这一特点,对于一些安全性需求较高的服务,Agent 可信度要求较高的应用可配置较高等级的安全策略规则;而对于一些普通应用,可以按需求配置相应的不同保护级别。

## 6 结语

本文基于服务分发式 Agent 的应用环境,对远端 Agent 可靠运行的安全隐患作了一系列探讨,并从保障 Agent 完整性入手,有机改造和结合 PCC、reflection 技术,提出了一套新的软件完整性验证体制,该验证机制以分级保护为主要优势,通过对 Agent 的身份识别与访问控制、外存代码的完整性验证以及运行时代码的完整性验证三个方面来确保 Agent 在非可

信平台的行为可监控性,从而在一定程度上提升了 Agent 的可信度。

### 参考文献:

- [1] VOAS J. Trusted software's holy grail [C]// HICSS'03: Proceedings of the 36th Annual Hawaii International Conference on System Sciences. Washington, DC: IEEE Computer Society, 2003, 9: 338.
- [2] KIROVSKI D, DRINIC M, POTKONJAK M. Enabling trusted software integrity [C]// Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems. New York: ACM Press, 2002: 108-120.
- [3] SPINELLIS D. Reflection as a mechanism for software integrity verification [J]. ACM Transactions on Information and System Security (TISSEC), 2000, 3(1): 51-62.
- [4] DOBBERTIN H, BOSSELAERS A, PRENEEL B. RIPEMD-160: a strengthened version of RIPEMD [C]// Proceedings of the Third International Workshop on Fast Software Encryption, LNCS 1039. Berlin: Springer-Verlag, 1996: 71-82.
- [5] NECULA G C, LEE P. Proof-carrying code [EB/OL]. (2002-07-22) [2009-03-05]. <http://raw.cs.berkeley.edu/pcc.html>.
- [6] 冯扬悦,陶先平,吕建. PCC 技术在移动 Agent 系统安全中的应用初探[J]. 计算机科学, 2002, 29(10): 20-25.
- [7] DEVANBU P T, FONG P W-L, STUBBLEBINE S G. Techniques for trusted software engineering [C]// Proceedings of the 20th International Conference on Software Engineering. Washington, DC: IEEE Computer Society, 1998: 126-135.

## 2010 年重点组稿方向

先进计算: 网格计算、可信计算、普适计算、可视计算、智能计算、移动计算、仿生计算、分布式计算。

信息安全: 网络安全、密码与安全协议、计算机系统安全、应用安全。

计算机网络: 网络系统技术、网络与通信、网络协议、无线网络。

数据库与知识工程: XML 数据管理、数据仓库、数据流管理、Web 数据库、数据集成、数据挖掘、特种数据库、语义 Web、Deep Web 挖掘、Web 侦察与反侦察、海量数据存储与处理。

软件过程技术: 软件重用、软件更新、组件技术、软件平台、软件环境、软件过程管理、过程评价、软件测试、软件可靠性、敏捷式软件开发方法、SAAS。

信息系统集成: 面向服务的软件构架、中间件、Web 服务技术、基于 Internet 的应用模式、系统协同设计与验证。

智能感知与嵌入式系统: 自动识别、机器学习、语音识别、传感器网络、RFID、嵌入式软件、智能化软件。

图形图像处理: 机器视觉、合成视觉与图像处理、生物特征识别。

现代服务业信息技术( 社会服务、数字服务、物流服务): 电子商务、电子政务、数字医疗、数字教育、数字旅游、数字媒体、数字地图、数字娱乐、现代物流。

典型应用: 先进制造、过程控制、智能交通、节能环保、虚拟现实、图形图像处理、生物医学计算、RFID、数字金融、数字农业。

本刊目前采用在线投稿的方式收取稿件,作者可到我刊主页 <http://www.computerapplications.com.cn> 上注册后投稿,投稿前请仔细阅读投稿须知及论文模板上的要求。