高技术通讯 2012 年 第22 卷 第2 期:147 ~ 152

doi:10.3772/j.issn.1002-0470.2012.02.006

# 可演化网络中移动代码的安全机制①

刘 涛②*** 王 锐*** 钱德沛③***

(*西安交通大学电子与信息工程学院 西安 710049)
(**中国科学院深圳先进技术研究院 深圳 518055)
(***北京航空航天大学计算机学院 北京 100191)

**摘 要** 针对可演化网络的动态可加载结构带来的潜在安全性问题,研究了移动代码的安全机制。针对移动代码在传输时表现为数据,在执行时表现为程序的特点,研究中考虑了静态的数据安全性和动态的程序安全性两个方面。提出了移动代码的完整性验证机制,利用可信计算平台生成用于加密的密钥,并验证节点的软硬件配置信息,保证了移动代码在传输、部署时的安全性。提出了运行时动态访问控制和资源监控管理的机制,保证了移动代码执行时的安全性。这些机制改善了可演化网络运行的安全性。

**关键词** 可演化网络,移动代码,网络安全,完整性验证

## 0 引 言

现有互联网络的基本设计理念是数据包在网络的中间节点即路由器被存储转发,不进行任何额外处理。目前这种数据包的转发大都是由实现了转发协议的特定硬件完成,网络的功能与形态一旦部署就难以更新。研究人员已经以多种方式探索了提高网络应对变化的适应能力,包括对网络硬件角色重定义[1]以及开放路由器结构[2]等。可演化网络[3]是一种动态可变换的网络结构,其网络设备能感知网络环境的变化,并根据环境变化对自身做相应的调整,通过及时更新整个网络的软件系统,提供更加适宜的网络服务,从而提高网络在应用环境变化下的服务质量。可演化网络原型的可演化能力由动态部署执行的移动代码来实现。这种代码的动态部署执行机制带来了新的安全问题。在网络节点间传输的代码和参数如果被途径的节点恶意篡改,将会导致部署的网络软件无法正常工作,造成网络设备故障。由于代码的移动性,错误操作和恶意攻击有可能传染到其他路由器,甚至造成整个网络出现问题。

可变结构的网络大都采用基于移动代码的代码迁移机制,很多研究人员对这种机制的安全性开展了研究。Lu[4]对移动代码的安全问题进行了全面的建模和分析,并提出了移动代码和宿主的相互认证模型。徐斌[5]在主动网络环境下实现了用于主动网络管理的执行环境,采用了基于软件的沙箱技术,保证了主动网络的安全性。Luan[6]在分布式管理系统中利用移动代码签名技术设定分布式的管理者所使用的权限,由此保证管理系统的正常运行。Adams[7]和林闯[8]分析了基于可信计算平台的网络系统访问控制技术,提出结合可信计算开发网络协议,可以用于保证网络安全。这些方法在一定程度上解决了代码移动的安全问题。与这些研究所涉及的场景不同,可演化网络中的移动代码均存于已知的代码服务器中,移动代码的正常运行需要网络节点、代码依赖关系、用户权限三方面均得到验证和满足。本文重点研究了移动代码部署时的完整性保护和运行时的资源监控,基于可信计算平台(trusted platform module, TPM),提出了移动代码完整性验证机制和动态访问控制和资源管理机制,以保证移动代码在传输和运行时的安全性,使网络节点始终执行正确安全的移动代码,从而提高可演化网络的整体安全性。本文给出了移动代码的完整性验证机制的实现方法,并进行了理论推导验证。

① 863 计划(2006AA01A109,2009AA01A131,2009AA01Z144)和国家自然科学基金(60673180, 90812001)资助项目。
② 男,1980 年生,博士;研究方向:计算机系统结构;E-mail: taobell@sina.com
③ 通讯作者,E-mail: depeiq@263.net
(收稿日期:2010-07-09)

## 1 移动代码的安全问题

由于可演化网络的主要功能由移动代码的部署、移动和执行来实现，因此移动代码的安全性尤为重要。本节分析可演化网络中移动代码传输、部署和加载执行过程中的安全威胁，定位可演化网络移动代码的安全性问题。

### 1.1 移动代码的工作方式

图 1 给出了可演化网络中移动代码的典型运行方式。其中管理员通过管理界面在可演化网络中部署移动代码，移动代码则在网络节点中逐步部署逐步执行，完成相关的网络任务。移动代码部署和执行的基本过程如下：

步骤 1：管理员从界面中选择需要部署的移动代码、配置参数，例如部署的目标节点名称等，并将部署命令发送到目标网络节点上。

步骤 2：目标网络节点接收命令，从代码服务器中下载相应的移动代码，并将移动代码和命令中的参数数据装载到执行环境中。

步骤 3：移动代码在目标节点上执行，完成其在单个节点上的任务。

步骤 4：移动代码根据执行结果，可能会打包必要的参数数据，有选择地迁移到下一节点，继续执行步骤 3 和步骤 4，直至整个任务完成。

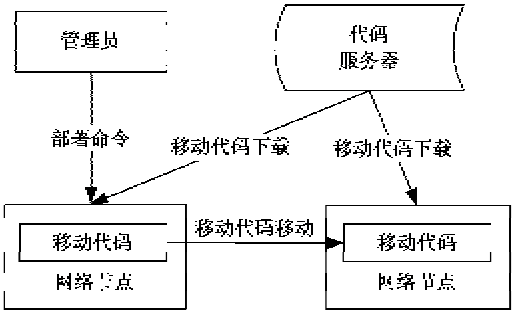


**图 1 可演化网络移动代码的典型工作方式**

### 1.2 移动代码的安全威胁

移动代码在网络中的传输和部署会面临各种安全威胁，这些威胁有可能来自恶意主机或者其他恶意的移动代码。在可演化网络中，移动代码具有数据和程序双重特性，在移动代码存储和传输时，可看作普通的数据块，而在移动代码被启动执行后，则应当作程序来看待。根据移动代码的这两种特性，其所面临的安全威胁也可以分为两类。

第一类威胁是当移动代码在网络中传输时受到的外部安全威胁。一种是被动攻击，攻击者在不干预正常传输的情况下，尝试从窃听到的流量中非加密部分如 IP 地址、数据包长度、数据包接收时间等进行统计分析，以获取敏感信息。另一种攻击是主动攻击，即攻击者截获并修改网络层的数据报，甚至对数据进行恶意的修改和伪造。

第二类威胁是当移动代码驻留在网络节点上执行时存在的安全威胁，既存在该移动代码对网络节点其他程序可能造成的威胁，也存在其他程序对该移动代码的安全威胁。主要有以下几种情况：

（1）破坏。被篡改的移动代码和网络节点之间可能会相互攻击，运行在同一个节点上的恶意移动代码之间也可能会相互攻击。

（2）服务拒绝。怀有恶意的用户可能滥用公共资源，发送大量无效的移动代码，导致正常的移动代码无法得到资源而被拒绝。

（3）偷窃。移动代码可以访问和窃取网络节点上的保密信息，即使经加密的数据也不一定安全，因为执行时，往往需要把它解密。

（4）组合攻击。网络节点的最大威胁是具有特定目的的组合攻击和分布式攻击。

（5）欺诈。不同的移动代码可具有不同的优先级和权限，低优先级的程序有可能采取欺骗手段以获得高优先级的权利。

## 2 移动代码的安全部署与执行机制

针对移动代码给可演化网络带来的两类安全威胁，分别在移动代码部署阶段和运行阶段采用不同的安全措施。在部署阶段将移动代码看作普通数据，确保其在传输中不被篡改，保证移动代码的完整性。在运行阶段将移动代码看作程序代码，确保移动代码合理使用节点的资源，不影响其他移动代码和网络节点的安全。

### 2.1 移动代码的完整性验证

在移动代码部署的特定时刻，对其完整性进行计算和验证，得到代码映像属性的散列值，将这些值与代码服务器所事先记录的标准值进行比较分析，从而判断代码的完整性是否遭到破坏。完整性验证方法不仅可以保证代码的安全性，还可以为其他的安全机制如访问控制等提供基本的安全数据保障。

在移动代码部署时所使用的完整性验证主要涉及以下 4 个方面的属性。

(1)代码的属性,记为 $P_{code}$,包括代码名称、代码版本、代码编写者、代码长度等。

(2)节点硬件的属性,记为 $P_{node\_hardware}$,包括节点的硬件性能等固有属性。

(3)节点软件的属性,记为 $P_{node\_software}$,包括节点的软件版本、节点已有软件库的版本等。该属性在运行时可以随着所安装软件的不同发生变化。

(4)用户的属性,记为 $P_{user}$,包括本次移动代码所使用的用户名、用户权限等。

图2显示了可演化网络移动代码的安全部署机制。其中可演化网络节点和代码服务器中的安全策略模块负责管理网络节点和代码服务器在进行交互通信时所使用的安全策略,其中包括节点标识、节点权限、通信协议、加密算法、认证算法等内容。

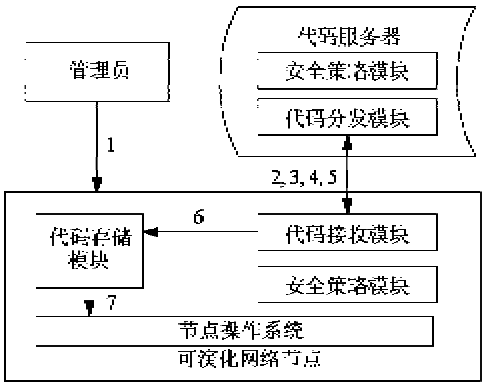


**图2　移动代码的安全部署机制**

移动代码安全部署的主要步骤如下:

步骤1:管理员配置需要发送的移动代码名称和参数数据,并利用私钥对其进行加密和签名,将命令发送到目标网络节点上。

步骤2:目标网络节点接收该命令,经验证后,向代码服务器请求下载移动代码映像。

步骤3:代码服务器与目标网络节点进行双向认证,检查本次部署的属性是否符合。

步骤4:代码服务器与目标网络节点协商加密算法和密钥。

步骤5:代码服务器利用协商好的算法和密码向目标网络节点传输移动代码映像。

步骤6:网络节点利用代码存储模块将映像存储到安全空间中。

步骤7:目标网络节点验证收到的移动代码映像,只有通过验证的代码映像才能进一步在执行环境中以特定的身份被加载和执行,并被赋予相应的权限。

移动代码的部署是否能正常首先取决于当前网络节点的软硬件环境和系统配置能否满足移动代码的执行,例如,硬件性能如何,软件模块的版本是否满足移动代码的最低要求,只有完全满足运行移动代码的所有基本条件时,移动代码才会被部署。

代码服务器除了存储移动代码映像及代码本身的属性之外,还要存储代码运行所需要的条件 $PC_{code}$,包括移动代码相对应的节点硬件配置 $P_{node\_hardware}$ 和软件配置 $P_{node\_software}$。此外,对于有权部署该移动代码的用户 $P_{user}$ 也加以限定。

**定义1:** $PC_{code}=\{P_{node\_hardware}, P_{node\_software}, P_{user}\}$ 是指移动代码所需要的运行环境。

**定义2:** $PC_{node}=\{P_{node\_hardware}, P_{node\_software}\}$ 是指网络节点现在的软硬件环境。

在这里,假定管理员所发出的启动移动代码的命令是安全的,本文只分析移动代码传输和启动时的安全性。约定如下符号:$S$ 为代码服务器;$N$ 为需要执行移动代码的网络节点;$MC$ 为移动代映像,即需要下载的代码;$K$ 为网络节点选择的用于数据加密的对称加密密钥。

为完成前述安全部署的步骤3中的属性验证和步骤4、步骤5中的移动代码传输,设计了如下协议流程:

(1)代码服务器上已经存储与该移动代码有关的所有合理配置集合 $PC_{code}=\{P_{node\_hardware}, P_{node\_software}, P_{user}\}$,该配置集合中的各个配置规定了相关移动代码运行的基本条件。

(2)网络节点从代码服务器下载移动代码的配置集合,并与本机的配置进行对比。如果本机配置满足条件,则启动属性验证过程。网络节点将本机的配置信息利用可信计算平台(TPM)生成用于加密的密钥集 $KI=TPM(K, PC_{node})$,其中 $K$ 为用于数据对称加密的密钥,$PC_{node}$ 为 $P_{node\_hardware}$,$P_{node\_software}$ 的所有值。

(3)网络节点向代码服务器请求下载移动代码。代码服务器发送公钥 $K_S$ 和随机数 $R$。

(4)网络节点利用TPM的身份认证密钥AIK对 $KI$, $R$, $K$ 进行计算,得出一个基于TPM可信度的签名 $SI=TPM(KI, R, K)$。

(5)网络节点利用代码服务器的公钥和 $R$ 加密 $K$,得到 $K_S(K, R)$。网络节点利用密钥集 $KI_{node}$ 中的所有密钥对 $R$ 进行环签名,得到 $S_R$,并将 $K_S(K, R)$, $SI$, $S_R$ 发送给代码服务器,并宣称本机具有该移动代码运行所需的所有条件。

(6)代码服务器利用 $PC_{code}$ 对 $R$ 进行环签名验证,并与,利用 TPM 对 $SI$ 进行验证,如果通过,则表示网络节点上软硬件配置和用户配置信息可以满足移动代码运行的所有条件。代码服务器利用 $K_S^{-1}$ 解开密钥 $K$。

(7)代码服务器利用 $K$ 加密移动代码映像,并发送给网络节点。网络节点利用 $K$ 解密移动代码映像,并启动执行。

协议流程中的(1)和(2)不存在通信过程,而从(3)开始进行属性验证协议的形式化描述如下:

1) $S \rightarrow N$: $K_S$, $R$

2) $N \rightarrow S$: $S_R = \{R\}_{KI}$, $\{K,R\}_{KS}$, $SI = TPM(KI, R, K)$

3) S: $Sign(R,PC_{code}) > S_R$, $K = K_S^{-1}(\{K, R\}_{KS},R)$, $TPM(SI,R)$

4) $S \rightarrow N$: $K(MC)$

为了考察该协议的完整性,我们利用 BAN[9] 逻辑对上述协议进行验证。BAN 逻辑是一种基于知识和信仰的形式逻辑分析方法,具有“看见规则”、“消息含义规则”、“随机数验证规则”等多种推理规则,被普遍用来推理和验证安全认证协议。上述消息用形式化方式可表示如下:

消息(1): $N \lhd (K_S,R)$

消息(2): $S \lhd (\{R\}_{KI},\{K,R\}_{K_S},\{K\}_{KI})$

消息(3): $N \lhd (< MC >_K)$

协议最初的基本信仰如下:

1) $S \mid \equiv \#(R)$

2) $S \mid \equiv \xrightarrow{K_S} S$

3) $N \mid \equiv \#(K)$

4) $N \mid \equiv \#(KI)$

5) $S \mid \equiv N \xleftarrow{KI} S$

根据这些基本信仰,我们期望在需要执行移动代码的网络节点 $N$ 上得到与代码服务器 $S$ 中相同移动代映像 $MC$。推导如下:

由消息(2)及“看见规则”,可得

$$S \lhd \{R\}_{KI} \quad (1)$$

由信仰 5)、式(1)及“消息含义规则”可得

$$S \mid \equiv N \mid \sim (R) \quad (2)$$

由式(2),信仰 1)及“随机数验证规则”可得

$$S \mid \equiv N \mid \equiv (R) \quad (3)$$

由消息(2)及“看见规则”,可得

$$S \lhd \{K,R\}_{K_S} \quad (4)$$

由式(4),信仰 2)及“消息含义规则”,可得

$$S \lhd \{K,R\} \quad (5)$$

由式(5)及“看见规则”,可得

$$S \lhd \{K\} \quad (6)$$

由式(6),消息(3)及“消息含义规则”,可得

$$N \mid \equiv S \mid \sim MC \quad (7)$$

即执行移动代码的网络节点接收到的移动代码映像是完整的。

在步骤 6 中,代码服务器通过判断,保证了网络节点软硬件和所使用的用户权限能够完全运行该移动代码。移动代码发送到网络节点后,仍然处于 TPM 机制的管辖范围内,代码的完整性得到保证。如果网络节点的实际属性并不能满足运行条件,则移动代码无法被网络节点的 TPM 正确解出,无法运行。

如果节点中的硬件属性 $P_{node\_hardware}$ 无法满足条件,则该节点无法运行该移动代码,部署将失败。如果节点的软件属性 $P_{node\_software}$ 不满足条件,则网络节点可以主动从代码服务器上下载完整可靠的软件库模块,在安装的版本满足条件后,节点可再启动部署的属性验证过程。

移动代码传输时的完整性由安全传输协议来满足。图 2 中,为了保证在代码传输的第 4 至 5 步交互的安全。移动代码在部署时由代码分发模块通过步骤 3 属性验证协议所得到的密钥 $K$ 对移动代码映像进行加密传输。在传输完成后,利用 TPM 对移动代码映像进行摘要签名,并在传输结束后进行对比,如果一样,则说明通过了完整性验证,部署工作正常结束,否则,将丢弃该移动代码。

### 2.2 移动代码运行时控制机制

当移动代码被完整地部署到网络节点后,网络节点将以移动代码所带的用户名和权限启动移动代码的执行。这时,移动代码以程序代码的形式存在于系统中。为保护网络节点资源和已运行的其他移动代码的安全,需要对移动代码的运行进行访问控制和资源管理。采用了两种机制,其一是树形的沙箱机制,保证移动代码与节点系统之间的逻辑隔离。其二是运行的资源监控管理机制,保证系统资源的安全使用。

在可演化网络节点的原型实现中,网络节点的系统软件和移动代码均采用 Java 语言来实现。各类代码均在执行环境(execution environment, EE)中进行加载和运行。网络节点中有 4 种代码加载器,采用树型结构来组织,如图 3 所示。根代码加载器加载最基本的软件模块,如 Java 自带的类,加密解

密算法类等。通用加载器用来加载网络节点的应用程序编程接口(API)类和移动代码基类。系统执行环境提供了网络节点中系统软件和管理软件的执行环境。而其它的移动代码执行环境分别加载不同的移动代码。

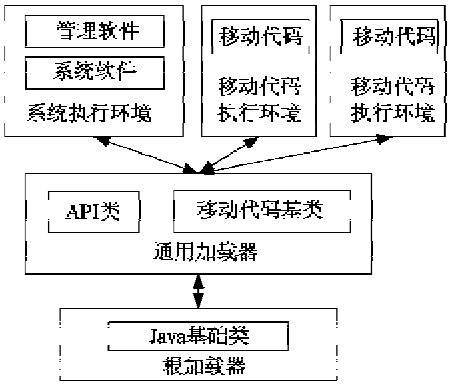


**图3　移动代码运行的沙箱机制**

在移动代码映像运行时,网络节点将生成一个临时的执行环境 $EE_i$, 每个临时执行环境均采用独立的代码加载器 $L_i$, 加载一个移动代码映像 $MC_i$。由此可以将各个移动代码的实际运行环境分配到不同的空间中,即不同的沙箱中。移动代码可以使用由本身的代码加载器及其父加载器所加载的代码,即叶子节点可以访问其父节点及根节点所加载的程序库代码,允许移动代码使用节点所提供的 API 资源。而处于不同沙箱中的叶子节点之间不能相互访问,保证了移动代码之间不会相互影响。

在可演化网络节点操作系统与执行环境之间增加了资源监控模块和访问控制模块,如图 4 所示。

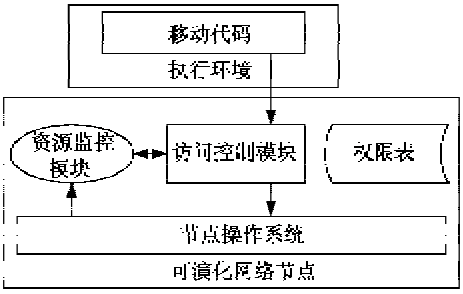


**图4　移动代码运行时访问控制和资源管理**

资源监控模块记录移动代码对节点资源的使用情况,同时通过对移动代码的活动评估来控制对节点资源的访问,防止单个移动代码对资源的滥用。

访问控制模块截取移动代码对操作系统 API 的调用,并将调用与从代码服务器下载的移动代码权限表进行对比,只有符合权限的调用才能继续正常执行,否则将抛出安全异常,终止移动代码的执行。

可演化网络节点的动态可配置资源监控管理结构提供了可扩展沙箱加载机制、权限表管理、API 访问、资源使用实时监控等安全服务,构建了易于扩展的安全结构,可以管理动态变化的移动代码所产生的安全事件,为移动代码运行提供了可配置的安全保障。

## 3　结　论

采用代码的动态部署与运行来实现网络的自适应性给可演化网络带来了新的安全问题。可演化网络原型系统采用移动代码完整性验证保证了移动代码在传输部署时的安全,通过动态访问控制和资源管理保证了移动代码在执行时的安全性。这两种安全措施解决了移动代码以数据形态传输部署和以程序代码形态动态执行时的安全性,有效地保证了可演化网络的安全运行。

**参考文献**

[1] Martin C, Teemu K, Daekyeong M, et al. Rethinking packet forwarding hardware. In: Proceeding of the 7th ACM Workshop on Hot Topics in Networks (Hotnets-VII), Calgary, Canada, 2008.1-6

[2] Jeffrey M, Praveen Y, Jean T, et al. API design challenges for open router platforms on proprietary hardware. In: Proceeding of the 7th ACM Workshop on Hot Topics in Networks (Hotnets-VII), Calgary, Canada, 2008

[3] 刘涛,钱德沛,王锐等.一种可演化网络的研究与实践.西安交通大学学报,2008,42(10):1193-1203

[4] Lu M, Tsai J. Formal modeling and analysis of a secure mobile-agent system. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2008, 38(1): 180-196

[5] 徐斌,钱德沛,张文杰等.主动网络管理代理的执行环境.计算机研究与发展,2002,39(11):1478-1483

[6] Luan Z Z, Qian D P, Zhang X J, et al. A novel model and architecture on NMS dynamically constructed network management. *Lecture Notes in Computer Science*, 2003, 2834: 398-403

[7] Adams W, Davis N. Toward a decentralized trust-based access control system for dynamic collaboration. In: Proceeding of the IEEE Workshop on Information Assurance and Security, Piscataway, USA, 2005. 317-324

[8] 林闯,封富君,李俊山.新型网络环境下的访问控制技

术.软件学报,2007,18(4):955-966

[9] Michael B, Martin A, Roger N. A logic of authentication. *ACM Transactions on Computer Systems*, 1990, 8(1):18-36

# Security mechanisms for mobile code in evolutionary networks

Liu Tao[*,**], Wang Rui[***], Qian Depei[*,***]

([*] School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an 710049)

([**] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055)

([***] School of Computer Science and Engineering, Beihang University, Beijing 100191)

**Abstract**

Considering that evolutionary networks with dynamic program loading structures introduces potential security problems, some security mechanisms for the mobile code in this kind of networks were proposed. Because the mobile code is in the form of data when in transmission while in the form of program when in execution, the research addressed both the security of static data and the security of dynamic program. A mechanism for verifying the integrity of the mobile code was proposed to ensure the safe transmitting and deploying of the mobile code. The trust platform module was used to generate the encryption key and to verify the hardware and software configurations of the node. The runtime dynamic access control mechanism and the resource monitor management mechanism were proposed to guarantee the mobile code's safe execution. These mechanisms improve the security of evolutionary networks.

**Key words**: evolutionary network, mobile code, network security, integrity verification