第33卷 第23期 计 算 机 工 程 2007年12月
Vol.33 No.23 Computer Engineering December 2007
·博士论文· 文章编号：1000—3428(2007)23—0009—03 文献标识码：A 中图分类号：TN915.08

一个抗截断攻击的移动代理数据保护方案

柳 毅[1,2]，张 凌[1]，王育民[3]
(1. 华南理工大学广东省计算机网络重点实验室，广州 510640；2. 中山大学广东省信息安全技术重点实验室，广州 510275；
3. 西安电子科技大学综合业务网国家重点实验室，西安 710071)

摘 要：研究了如何保护自由漫游的移动代理运行结果安全问题，总结了当前已有方案的特点和不足，指出了这些方案中存在的一个共同缺陷假设，即对于路由主机，同一移动代理只能经过其一次，不能多次访问。提出了一个抗截断攻击的保护移动代理运行结果方案，满足安全要求和抗弱截断攻击，去掉了当前方案的缺陷假设，增强了协议应用的灵活性。
关键词：移动代理；自由漫游；运行结果；截断攻击

Method for Protecting Mobile Agents Data Against Truncation

LIU Yi[1,2], ZHANG Ling[1], WANG Yu-min[3]
(1. Guangdong Key Laboratory of Computer Network, South China University of Technology, Guangzhou 510640;
2. Guangdong Province Key Laboratory of Information Security, Sun Yat-sen University, Guangzhou 510275;
3. National Key Laboratory of Integrated Server Networks, Xidian University, Xi'an 710071)

【Abstract】This paper studies how to protect the computational results of free-roaming agents and analyzes the advantages and disadvantages of the existing solutions. After that, an assumption existing in these protocols which limits the mobile agent's flexibility is pointed out, the mobile agent can pass a host only once and must not visit the same host many times. A new method for protecting mobile agent data against truncation is presented and its security is analyzed. The results show that this method not only satisfies security requirements but also removes the assumption in existing protocols.
【Key words】 mobile agents; free-roaming; computational results; truncation

移动代理技术最显著的特点在于移动代理这个软件具有实体的灵活性和便捷的移动性。如何保护移动代理和路由主机的交互结果，保护移动代理运行结果的数据安全是目前研究移动代理系统安全的重要课题之一[1]。

本文介绍了移动代理自由漫游时的运行结果安全所需满足的安全性质，对目前已有的方案进行了讨论，提出了一个抗截断攻击的保护移动代理运行结果方案，与已有方案相比具有更新的特点和更好的灵活性。

1 符号标记及安全要求

1.1 符号标记

Uid：移动代理的唯一标识。

H_0：移动代理主人，即用户。

H_i：移动代理经过的路由主机，$i=1,2,\cdots$。

o_i：移动代理和H_i的交互结果，即H_i向用户提供的offer，$i=1,2,\cdots$。

O_i：对o_i采取一定的加密签名技术后，H_i发给移动代理的信息，$i=1,2,\cdots$。

x_i：主机H_i的私钥，$i=1,2,\cdots$。

y_i：主机H_i的公钥，$i=1,2,\cdots$。

$E_i(.)$：采用主机H_i的公钥对信息进行加密。

$S_i(.)$：主机H_i利用自己私钥对信息进行签名。

$h(.)$：安全抗碰撞的杂凑函数。

$A \to B:m$：主体 A 向主体 B 传送消息 m。

1.2 安全要求

假设移动代理访问了m个主机后，将要经过恶意主机H_{m+1}。这时，移动代理携带着所经过主机 $H_0 \sim H_m$ 提供的信息：$O_0,O_1,\cdots,O_m$。假设H_{m+1}可能和前面的主机 $H_i(0<i<m)$ 共谋，但要求它不会和H_m共谋（否则可将路由中位置连续的H_m和H_{m+1}看作一个恶意主机）。这里引用Karjoth和Cheng等给出的目前公认的保护移动代理运行结果所须满足的安全要求[2]：

$S1$ 数据保密性：只有H_0可以从O_i中获取正确的o_i。

$S2$ 不可否认性：当H_0得到o_i后，H_i不能否认o_i是它所给出的。

$S3$ 身份保密性：对于o_i的主机身份H_i，只有H_0可以从O_i中提取获得，其他主机不能得到。

$S4$ 前向完整性：任何 $O_j(j<m)$ 都不能被修改。

$S5$ 公开可证实的前向完整性：如果需要，那么通过检查某个链式关系，任何实体都能够证实O_i的合法性。

$S6$ 抗插入攻击：不能插入新的 $O_j'(j<m)$。

$S7$ 抗删除攻击：任何 $O_j(j<m)$ 都不能被删除。

$S8$ 抗强截断攻击：截断攻击是指H_{m+1}和一些主机共谋，将移动代理所携带的 $O_0,O_1,\cdots,O_m$ 从某 $i(0<i<m)$ 处截断，删除i处后面所有的 $O_k(i \le k \le m$或$i<k \le m)$，重新产生新的信息来替代它们。抗强截断攻击是指：即使H_{m+1}和任意数目

基金项目：广东省信息安全技术重点实验室开放基金资助项目
作者简介：柳 毅(1976－)，男，博士后，主研方向：电子商务安全，网络安全协议，移动代理；张 凌、王育民，教授、博士生导师
收稿日期：2006-12-20 **E-mail：**liuyi_xd@126.com

的主机共谋也不能发动该截断攻击。

S8′抗弱截断攻击：H_{m+1}不能发动截断攻击，除非H_{m+1}能和移动代理路由中连续的L个主机共谋。这里$L>1$ 为一安全门限值。

2 已有方案的讨论

关于如何保护自由漫游的移动代理运行结果，Yee曾利用局部认证来确认路由主机offer的完整性[2]，但是该方案存在很多安全隐患，如前向完整性。Karjoth等改进了Yee的结果，提出的方案[3]可以很好地满足安全要求S1~S7，但是仍无法抵制两个不相邻主机联合发动的截断攻击。

自从 Karjoth 方案提出以来，其后的研究工作就主要集中在如何在其基础上提出一个能够满足抗截断攻击的方案。对此，文献[4~6]分别给出了不同的解决方案。文献[4]是通过认定移动代理再次经过同一主机的行为为恶意来达到抗弱截断攻击(否则恶意主机可以利用其前驱主机提供的多次签名来达到抗截断攻击)。文献[5]则是通过绑定未来要经过的多个主机的身份来实现抗弱截断攻击，但是该方法不仅限制了移动代理漫游过程中的灵活性，而且会泄漏不相邻主机的身份。文献[6]是目前为止唯一能实现抗强截断攻击的方案。方案中引入了可信第三方，通过路由主机和可信第三方预先进行的秘密信息交互，使得企图发动截断攻击的路由主机将会面临泄漏自己的私钥来达到抗强截断攻击的目的。

通过以上分析可以看出，目前为止在抗截断攻击方面能够得到较好应用的方案为文献[4,6]。但是这两个方案的实现都存在一个共同的前提假设：对任何一个路由主机来说，同一个移动代理只能经过一次。即认定同一移动代理多次经过同一主机的行为为恶意行为。如果去掉这个假设，这些方案或者不能满足安全要求，或者会泄漏主机的秘密信息，如泄漏路由主机的私钥[6]。

第 3 节提出了一个抗截断攻击的保护移动代理运行结果方案。方案不仅满足安全要求，而且去掉了已有方案的这一共同缺陷假设，使得同一移动代理可以多次经过同一路由主机，增强了方案实际应用的灵活性。

3 一个新的抗截断攻击方案

为叙述简单，给出的协议要求不能有两个连续的主机共谋，但是通过协议描述可以看出，该方案能够很方便地推广到抵制连续 L 个主机共谋的情形。

3.1 方案描述

基于ElGamal公钥体制，公开的系统参数为：p, q, g。其中，p, q是大素数；q是p-1 的一个因子；g是 z_p^* 中的一个q阶生成元。主机H_i的私钥为 $x_i < q$ ，公钥为 $y_i = g^{x_i} \bmod p$ 。

(1)协议准备阶段

每个路由主机H_i将其私钥x_i分拆为 $\hat{x}_i$ 和 $\tilde{x}_i$ ，要求满足 $\hat{x}_i + \tilde{x}_i = x_i \bmod q$ ，并计算 $\hat{y}_i = g^{\hat{x}_i} \bmod p$ 和 $\tilde{y}_i = g^{\tilde{x}_i} \bmod p$ ，则必然有下式成立：

$$\hat{y}_i \cdot \tilde{y}_i = g^{\hat{x}_i} \cdot g^{\tilde{x}_i} = g^{\hat{x}_i + \tilde{x}_i} = g^{x_i} = y_i \bmod p \qquad (1)$$

其中，$\hat{x}_i$ 和 $\tilde{x}_i$ 均为一次性的，即每次移动代理经过时，H_i都要重新计算不同的 $\hat{x}_i, \tilde{x}_i, \hat{y}_i, \tilde{y}_i$ 。

(2)协议执行阶段

设移动代理（唯一标识为Uid）自由漫游经过n个主机。代理在主机H_i产生运行结果o_i，H_i进行如下步骤：

1) H_i与其后继主机H_{i+1}进行信息交互

$$H_i \to H_{i+1} : E_{i+1}(S_i(H_i, H_{i+1}, Uid, count))$$

$$H_{i+1} \to H_i : E_i(S_{i+1}(\hat{y}_{i+1}, Uid, count))$$

其中，$count$值表示同一移动代理是第几次经过主机H_i。

2)H_i与其前驱主机H_{i-1}进行信息交互

$$H_i \to H_{i-1} : E_{i-1}(S_i(H_{i-1}, H_i, \hat{y}_{i+1}, Uid, count))$$

$$H_{i-1} \to H_i : E_i(S_{i-1}(h(H_{i-1}, H_i, \hat{y}_{i+1}, Uid, count)))$$

3)计算O_i

$$O_0 = E_0(H_0, H_1, S_0(o_0, \tilde{x}_0, \hat{y}_1), Uid, 1), h_0$$

$$O_i = E_0(H_{i-1}, H_i, H_{i+1}, S_i(o_i, \tilde{x}_i, \hat{y}_{i+1}, Uid, count),$$

$$S_{i-1}(h(H_{i-1}, H_i, \hat{y}_{i+1}, Uid, count))), h_i, \quad 1 \le i \le n$$

4)计算链式关系

$$h_0 = h(r_0, H_1)$$

$$h_i = h(O_{i-1}, H_{i+1}), 1 \le i \le n$$

5)信息传递

$$H_i \to H_{i+1} : \{O_k \mid 0 \le k \le i\}$$

移动代理开始漫游时，H_0产生一个随机数 r_0 和一个 o_0 ，然后和 H_1 交互计算 O_0 和 h_0 ，并将 O_0 发送给 H_1 。当移动代理到达 $H_i, 1 \le i \le n$ 时，H_i首先检查链式关系 $h_k, 0 \le k \le i-1$ 是否合法，然后会检查移动代理的Uid是否与先前H_{i-1}传来的一致。接着开始与H_{i+1}交互：H_i通知H_{i+1}移动代理的Uid和$count$，H_{i+1}传回签名的一次性 $\hat{y}_{i+1}$ (每次与H_{i+1}交互时都要求产生不同的 $\hat{y}_{i+1}$)；与 H_{i-1} 进行交互：H_{i-1} 收到 $E_{i-1}(S_i(H_{i-1}, H_i, \hat{y}_{i+1}, Uid, count))$ ，解密验证签字后，先查看 H_{i-1}, H_i 的身份，验证代理Uid和$count$。然后对其进行杂凑并签名后，将信息返回H_i；H_i计算 O_i ：O_i 中包括H_{i-1}、H_i和H_{i+1}的身份，H_i签字后的 $o_i, \tilde{x}_i, \hat{y}_{i+1}$ 以及移动代理Uid和$count$，H_{i-1}签字的杂凑信息以及链式关系 h_i ；步骤 4)计算步骤 3)中所需的链式关系 h_i ；H_i将 $O_0, O_1, \cdots, O_i$ 传给H_{i+1}。以上任何检查出现异常，H_i都会向用户报告出错信息。当移动代理决定返回 H_0 时，$H_n \to H_0 : \{O_k \mid 0 \le k \le n\}$ 。H_0除了进行以上相同检查外，还要根据式(1)进行相应检查。

3.2 方案安全性分析

S1 数据保密性：只要所采用的加密算法是安全的，从H_i对信息的处理过程可以看出，只有H_0可以从O_i中获取正确的o_i。

S2 不可否认性：由于O_i中包含了H_i对o_i的签字，因此当H_0得到o_i后，H_i不能否认o_i是由它所给出的。

S3 身份保密性：只要加密系统是安全的，只有用户H_0能利用自己私钥解密O_i后才能获得提供o_i主机身份H_i（O_i中的第2 项）。

S4 前向完整性：如果攻击者企图保持O_m完整，而修改 $O_j, j < m$ 。不妨假设 $j = m-1$ ，攻击者利用 O'_{m-1} 来替代 O_{m-1} 。由于在O_m中包含链式关系 $h_m = h(O_{m-1}, H_{m+1})$ ，为了保持该关系，就要求有 $h(O_{m-1}, H_{m+1}) = h(O'_{m-1}, H_{m+1})$ 。因此，只要采用安全抗碰撞的杂凑函数 $h(.)$ ，攻击者就不能修改 O_{m-1} 。类似归纳可得：攻击者不能修改任意 $O_j, j < m$ 。

S5 公开可证实的前向完整性：通过检查链式关系 h_i ，任

何实体都能够证实O_i的合法性。

*S*6 抗插入攻击：给定 $O_0,O_1,\cdots,O_m$ 后，链式关系保证了攻击者不能在其中插入任何新的 $O_j^{'}, j<m$。

*S*7 抗删除攻击：链式关系也保证了任何 $O_j, j<m$ 都不能被删除。

*S*8′抗弱截断攻击：假设H_{m+1}和主机H_i联合，企图发动截断攻击，删除所有 $O_k, i\le k\le m$ 或 $O_k, i<k\le m$。但是当代理返回H_0后，从下面的分析可以得出：这种截断攻击会被发现。

因为 H_{i-1} 给出的 O_{i-1} 中包含H_i身份信息，所以为保证信息的连贯合法性，H_i必须给出某个O_i，而且代理也将由它再次发出。下面分两步对此进行讨论（假设不会有路由连续的两个主机共谋）：

(1)H_i给出的信息中必定包含 $\hat{y}_{i+1}$。

原 O_i 中包含 H_{i-1} 的签字信息 $S_{i-1}(h(H_{i-1},H_i,\hat{y}_{i+1},Uid,count))$，即使$H_i$想给出新的 $O_i^{'}$，但为了保证信息的连贯合法性，H_i在 $O_i^{'}$ 中也只能保留原有的 $S_{i-1}(h(H_{i-1},H_i,\hat{y}_{i+1},Uid,count))$（否则，$H_{i-1}$ 通过向 H_0 出示 $S_i(H_{i-1},H_i,\hat{y}_{i+1},Uid,count)$ 就可以认定H_i恶意），所以，$O_i^{'}$ 中必须包含 $\hat{y}_{i+1}$。

(2)不论H_i想把代理再次派发给谁，H_0 都会认定H_i恶意。

如果H_i依然想把移动代理传给H_{i+1}，只能通知H_{i+1}为代理第 $count+1$ 次经过（因为H_{i+1}知道代理已经经过自己 $count$ 次）。在移动代理完成任务后，H_{i+1}会在新的 $O_{i+1}^{'}$ 中加入新的 $\tilde{x}_{i+1}^{'}\ne\tilde{x}_{i+1}$，而当代理返回用户，$H_0$在检查 $O_i^{'},O_{i+1}^{'}$ 时会发现，$O_i^{'}$ 提供的 $\hat{y}_{i+1}$ 与 $O_{i+1}^{'}$ 提供的 $\tilde{x}_{i+1}^{'}$ 不能满足等式：$\hat{y}_{i+1}\cdot g^{\tilde{x}_{i+1}^{'}}=y_{i+1} \bmod p$。用户$H_0$追究责任时，$H_{i+1}$会提供 H_i 发给它的信息 $E_{i+1}(S_i(H_i,H_{i+1},Uid,count+1))$，但是$H_i$则无法提供$H_{i-1}$应给出的信息 $E_i(S_{i-1}(h(H_{i-1},H_i,\hat{y}_{i+1},Uid,count+1)))$，因此 H_0 会认定H_i恶意。

如果H_i不再把代理传给H_{i+1}，而是选择某个其他的主机 H_l，那么 $O_i^{'}$ 中必须包含 $\hat{y}_{i+1}$，这样即使H_i与 H_l 共谋，令 $\hat{y}_l=\hat{y}_{i+1}$，虽然 H_l 根据 $\tilde{y}_l\cdot\hat{y}_l=y_l \bmod p$ 可以求出 $\tilde{y}_l$，但是由于求离散对数的困难性，H_l 并不能在 O_l 中给出 $\tilde{x}_l$，使得满足 $g^{\tilde{x}_l}=\tilde{y}_l \bmod p$，即满足等式：$\hat{y}_l\cdot g^{\tilde{x}_l}=y_l \bmod p$，因此$H_0$仍旧会发现$H_i$的恶意行为。

H_i也不能任意伪造主机身份来达到截断攻击的目的。例如H_i谎称存在主机H_F并且声称将移动代理发送给了H_F。而H_i却令 $\hat{y}_F=\hat{y}_{i+1}$，任取 $\tilde{x}_F$，私自计算并谎称 H_F 的公钥 y_F 为：$y_F=\hat{y}_F\cdot g^{\tilde{x}_F} \bmod p$，进而产生相应的 y_F。但是当移动代理返回H_0后，H_0会发现 H_F 并不具有公开合法的公钥证书，因而依然会认定H_i恶意伪造主机身份。

另外，协议中任意主机 H_j 不会因为暴露了 $\tilde{x}_k$ 和 $\hat{y}_k$ 而危害它自身私钥 x_k 的安全。这是因为：$x_k=\tilde{x}_k+\hat{x}_k \bmod q$，而由离散对数的困难性，通过等式 $\hat{y}_i=g^{\hat{x}_i} \bmod p$ 并不能求出 $\tilde{x}_k$，也就无法得出 x_k。

通过以上分析可以得出，方案满足安全要求*S*1~S*7* 以及 *S*8′（此处给出的方案要求H_{m+1}至少能和移动代理路由中连续的 2 个主机共谋，如H_{m+1}和 H_{i-1},H_i 共谋或者和 H_i,H_{i+1} 共谋，才能采取截断攻击。从方案的实现步骤可以看出，该方案可以很容易地推广至H_{m+1}必须与连续的L个主机共谋，才能采取截断攻击的情形）。

4 结束语

本文分析了对于如何保护移动代理自由漫游时的运行结果，总结了当前已有方案的特点和不足，并且指出了当前方案中存在的一个共同缺陷假设。在讨论已有方案的基础上，提出了一个新的抗弱截断攻击的保护移动代理运行结果方案。方案满足安全要求 *S*1~*S*7 和抗弱截断攻击，更重要的是，该方案去掉了当前方案的缺陷假设，增加了协议应用的灵活性。下一步的目标是考虑如何将本文的方案应用到移动代理系统中，以在保持系统灵活性的同时，增加系统的安全性。

参考文献

1 Borselius N. Mobile Agent Security[J]. Electronics & Communication Engineering, 2002, 14(5): 211-218.

2 Yee B S. A Sanctuary for Mobile Agents[M]. Berlin Heidelberg: Springer-Verlag, 1997: 261-273.

3 Karjoth G, Asokan N, Gulcu C. Protecting the Computation Results of Free-roaming Agents[M]. Berlin: Springer-Verlag, 1998: 195-207.

4 Cheng J S L, Wei V K. Defenses Against the Truncation of Computation Results of Free-roaming Agents[M]. Berlin: Springer-Verlag, 2002: 1-12.

5 Maggi P, Sisto R. A Configurable Mobile Agent Data Protection Protocol[M]. New York, USA: ACM Press, 2003: 851-858.

6 Ming Y, Kun P, Matt H, et al. Using Recoverable Key Commitment to Defend Against Truncation Attacks in Mobile Agents[M]. Berlin: Springer-Verlag, 2004: 164-173.

（上接第 8 页）

6 Driverworks Help Documents[Z]. (2006-05). http://www. compuware. com/ products/ driverstudio/.

7 Ivar J, Grady B, James R. The Unified Software Development Process[M]. 2nd ed. [S. l.]: Addison Wesley Longman, 1999.

8 Gamma E. Design Patterns: Elements of Reusable Object-oriented Software[M]. [S. l.]: Addison Wesley, 1995.

9 Alexandrescu A. Modern C++ Design[M]. 2nd ed. [S. l.]: Addison Wesley, 2001.

10 gcc manual[Z]. (2006-05). http://gcc.gnu.org/onlinedocs/ .

11 Meyers S. Effective C++[M]. 2nd ed. [S. l.]: Addison Wesley, 1998.

12 候　捷. STL 源码剖析[M]. 武汉: 华中科技大学出版社, 2002.