采用增量时间戳安全技术的 Mobile Agent 系统*

万 燕, 孙永强, 朱向华, 唐 进

(上海交通大学 计算机科学与工程系, 上海 200030)
E-mail: winniewan@mail.zte.com.cn
http://www.sjtu.edu.cn

摘要: 安全技术决定了 Mobile Agent 的实用性,其主要解决的问题是防范恶意 Mobile Agent 的重复攻击和越权攻击.在介绍了 Mobile Agent 的概念、特点及安全的重要性的基础上,提出了增量时间戳的概念及优点,并给出了具体例证.采用增量时间戳的 Mobile Agent 系统能够较好地避免执行系统受到恶意 Mobile Agent 的损害.增量时间戳技术已经在基于 Lucent 公司的 Inferno 系统上开发的 Mobile Agent 系统中得到了验证和实现.

关 键 词: Mobile Agent;增量时间戳;安全
中图法分类号: TP393 文献标识码: A

Mobile Agent(简称 MA)是 20 世纪 90 年代中期兴起的一种信息服务与任务求解方式.Agent 自主地在异构的网络中迁移,然后可以继续执行.正是由于 Mobile Agent 的这种自主迁移方式,使得安全技术成为决定 Mobile Agent 实用性的最主要因素.本文提出了一种新的安全技术方案——增量时间戳,较好地解决了 Mobile Agent 的几个安全问题.

1 Mobile Agent 技术及安全性

1.1 Mobile Agent概念

MA 是一种新型的 Agent 技术,是 Agent 技术与分布式计算技术相结合的产物.它的主要特征是具有跨平台移动计算的能力,其基本思想是使 Agent 在分布环境中自主地迁徙,依次使所需数据在计算机系统上直接操作,然后返回到用户本地机上,而 MA 在系统中的迁徙对用户而言是完全透明的.在迁徙过程中,Agent 可以在到达目的地后继续执行,并且这种移动执行是持续的,即 Agent 能够保持自身的状态.

1.2 Mobile Agent的技术优势

MA 是一种新的信息服务与任务求解方式.与传统的 Client/Server 模式相比,MA 系统具有许多技术优势[1]:

(1) 移动执行方式.MA 摆脱了传统的 Client/Server 框架,通过将服务请求 Agent 移动到服务器端执行,使得 Agent 能够不经过网络传输而直接访问服务器资源.在 MA 系统中,网络传输的是 Agent 代码,而不是拥有大量冗余的数据,从而降低了系统对通信带宽的依赖,可大幅度减少网络的拥塞.

(2) 动态路由.MA 的具体移动路线和移动目标不是由用户预先设定的,而是由 Agent 在运行过程中,通过自身的感知器动态地取得外界网络环境和服务器负载情况后再进行规划.

(3) 负载均衡.MA 的动态移动策略能够优化网络传输路径和服务器负载,降低网络延迟,避免资源访问的盲目性,并能将服务请求的冲突降低到最低限度.

* 收稿日期: 2000-09-30; 修改日期: 2001-04-03

作者简介: 万燕(1970－),女,湖北黄冈人,博士,主要研究领域为移动代理,安全技术;孙永强(1931－),男,浙江嘉善人,教授,博士生导师,主要研究领域为程序语言,并行分布处理;朱向华(1977－),男,江西人,硕士,主要研究领域为移动代理;唐进(1976－),男,湖南人,硕士,主要研究领域为移动代理.

(4) 异步计算.MA 的运行不需要统一的调度,也不需要用户端与服务器端保持长时间的稳定连接.由用户创建的 MA 可以异步地运行或暂存于不同的网络节点上,待任务完成后再将结果传给用户.

(5) 并行计算.为完成某项任务,用户可以创建多个 Agent,同时在相同或不同的节点上运行.MA 的并行性可将单一节点的负载分散到网络的多个节点上,并能减少局部网络的传输拥塞,这将使得小系统拥有处理大规模、复杂问题的能力.

(6) 隐藏网络细节.Agent 的移动性对应用的开发者而言隐藏了网络的细节,提供了一种通用的开发手段,并且使得在执行其任务时不用考虑网络的质量和可靠性,为 WAN 或无线环境提供了一种解决方案[2].

1.3 Mobile Agent系统的安全问题

MA 为分布计算提供了全新的设计范型,但是它在主机之间迁移并执行会对系统的安全造成极大的威胁.MA 的安全问题综合了传统大型分布式系统和多用户操作系统的安全问题:在分布式系统中,主体(principal)之间相互不信任,它们不存在互相了解;在操作系统中,应用程序对主机系统的 API 有全权访问权,享有主机的资源,并且与不相干的应用程序分享着资源.不仅如此,MA 还引发了新的安全问题:MA 并不信任主机[3].

MA 系统由 4 部分组成[4]:

· 主机——一台计算机和一个操作系统;

· 计算环境(CE)——运行时间系统,即 Agent 执行环境(AEE);

· 移动对象系统——在 CE 上并行执行的计算;

· 将分布在不同主机上的 CE 连接起来的网络或通信子系统(如图 1 所示).

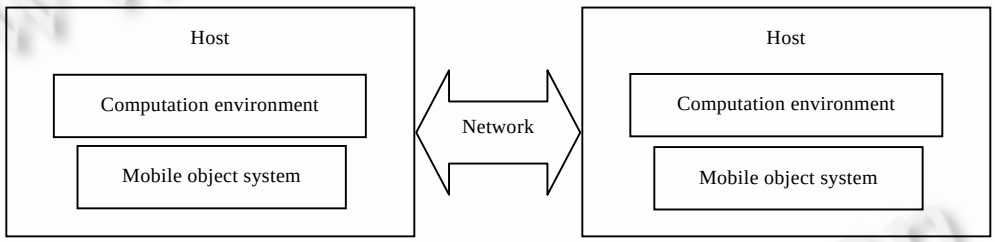

Fig.1 The component of the MA system

图 1 MA 系统组件

MA 系统的安全问题包括:

(1) 如何保护 AEE 不受恶意 Agent 的攻击;

(2) 如何保护 Agent 不受恶意 AEE 的攻击;

(3) 如何保护一个 Agent 不受另一个 Agent 的攻击;

(4) 如何保护一个 AEE 不受另一个 AEE 的攻击;

(5) 如何保护 AEE 之间的通信;

(6) 如何保护主机不受 AEE 的攻击.

目前所有 MA 的安全问题还不能完全得到解决[5].我们现在提出一种增量时间戳的技术,并就可能得到解决的问题加以探讨,同时研究所能采取的措施.

2 增量时间戳技术

针对 MA 系统中的某些安全问题,增量时间戳技术可以进行有效的解决和防范.下面介绍增量时间戳及其管理.

2.1 增量时间戳的定义

下面是有关互质增量时间戳的一系列定义,需要指出的是,该定义中对 $a_1,a_2,a_3,\ldots$ 互质的要求 $gcd(a_i,a_j)=1$ 是

为了简化在某个时间戳基值可发放时间戳个数的最大值 MAXNUM 的控制.

假设某个站点 A 有 K 个相邻的节点 $N_1,N_2,\ldots,N_K$ 为每一个相邻节点 N_i,选取一个正整数 a_i 且 $a_i>1$(增量值),且满足条件:

$$gcd(a_i,a_j)=1,\text{对任何 } i,j,1\le i<j\le K;$$

定义 MAXNUM=$\min(a_1,a_2,\ldots,a_K)-1$,站点 A 在任何时间戳基值 $Base$ 向任何一个相邻节点最多发送 MAXNUM 个增量时间戳.

当时间戳基值为 $Base$ 时,A 站点向相邻节点 N_i 发送的第 m 个($1\le m\le$MAXNUM)增量时间戳定义为

$$stamp(i,m)=Base+a_i*m.$$

取 $a_1,a_2,\ldots,a_K$ 中的最大的两个值,不妨设为 a_K-1,a_K 且 $a_K-1<a_K$,定义时间戳基值的增量为

$$BaseIncr=a_K-1*a_K.$$

2.2 增量时间戳的管理

在每个服务器上要建立并维护两个表:相邻服务器时间戳表(FROM 表)和本地时间戳发放目录表(TO 表).在 FROM 表中,每一项记录对应于一个相邻服务器,其中记录着:(1) 该相邻服务器 N_i 的域名地址;(2) 其当前时间戳基值 Basei;(3) 该相邻服务器时间戳的变动特征值 a_i;(4) 可使用时间戳的位图;(5) 该相邻服务器的公开密钥.在 TO 表中记录着:(1) 本服务器的当前时间戳基值;(2) 对应于每一个相邻站点的时间戳特征值列表.

当本地服务器收到相邻服务器发来的第 j 个时间戳($Base_i+j\times a_i$),如果这时位图的第 j 位为 0,则将其置为 1;若不为 0,则报告有攻击存在.每当有一个要迁徙的代理向本地申请一个时间戳时,本地的服务器先查找到该代理的其目的地服务器对应的时间戳特征值,然后计算发放的时间戳的值.在一定的时间间隔内,相邻的服务器将询问本服务器当前的时间戳,服务器取得系统当前的时间戳基值,用公有密钥签名,传送给询问的服务器.这样,每个服务器可以获知与其相邻服务器最新的时间戳基值,并且所有服务器中的时间戳基值都具有一致性.

2.3 增量时间戳的安全分析

(1) 可以有效地防范重复攻击(replay attack)

当 Mobile Agent 要访问远程服务器 N_i 时(假设该 Mobile Agent 所携带的增量时间戳为 stamp(i,m)),如果 m 已经在 N_i 的 FROM 表中,对应于源服务器的已使用时间戳位图中,标志为 1,则认为试图建立连接的对方是在试图进行重复攻击,终止与它的连接并报告系统管理员.如果对应的位图中的位仍为 0,则将其设置为 1.可参见第 3 节的实例检验.

(2) 可以防止合法的 Mobile Agent 的越权迁徙行为

尽管可以将已经在系统中运行的 Mobile Agent 认为是合法的,但我们不能保证它们不试图违规行为.因此,要求有较完善的时间戳机制.否则,可能会出现混乱的情况.比如某个 Mobile Agent 向本地服务器 A 申请时间戳,要迁徙到相邻站点 X,于是本地服务器发给它完整可靠的时间戳,并记录其下一个目的地为 X,但这个 Mobile Agent 用这个时间戳向相邻站点 Y 发出迁徙请求,于是完成迁徙工作,此时系统中已经出现了不一致的状态,而这是系统无法主动探测的.但在引入互质增量时间戳后,可以防止这种情况的发生.下面给出相应的证明.

求证:某个 Mobile Agent 在站点 A 申请的前往站点 X 的时间戳不能用作前往站点 Y 的时间戳.

证明:不妨设站点 A 有相邻节点 $N_1,N_2,\ldots,N_K$,且 $X\in\{N_1,N_2,\ldots,N_K\}$,$Y\in\{N_1,N_2,\ldots,N_K\}$,此时站点 A 的 TO 表为

Local stamp base	Host name	Stamp increase
BASE A	N_1	a_1
	…	…
	X	a_X
	…	…
	Y	a_Y
	…	…
	N_K	a_K

此时 Agent 的时间戳为

$$stamp(X,m)=Base_A+a_X\times m, \tag{1}$$

其中 $1\le m\le$MAXNUM,

$$MAXNUM=\min(a_1,a_2,\ldots,a_K)-1. \quad (2)$$

假设该 Agent 在站点 A 申请的前往站点 X 的时间戳能够用作前往站点 Y 的时间戳,则根据互质增量时间戳的定义

$$stamp(X,m)=Base_A+a_Y\times n, \quad (3)$$

其中 $1\le n\le MAXNUM$,

$$MAXNUM=\min(a_1,a_2,\ldots,a_K)-1.$$

根据式(1)及式(3)得,$a_X\times m=a_Y\times n\Leftrightarrow a_X\times m=0 \bmod a_Y$.

又因为 $gcd(a_X,a_Y)=1$,对任何 $X,Y,1\le X<Y\le K$;

所以 $$m=0 \bmod a_Y. \quad (4)$$

但根据式(2),

$$m\le MAXNUM<a_Y\Rightarrow m\neq 0 \bmod a_Y, \quad (5)$$

式(4)与式(5)相矛盾,所以假设不成立. □

(3) 防止 Mobile Agent 使用过期的时间戳进行迁徙

当某一个 Mobile Agent 从本地服务器申请到合法的时间戳进行迁徙后,它实际上获得了对该时间戳的控制,但仅限于这次迁徙.如果下一次仍需要从同一个站点迁徙到相同目的地,它必须向本地服务器重新申请时间戳,即它不能使用过期的时间戳进行迁徙,这一点正是传统时间戳机制的关键作用之一.以下是相应的证明.

求证:某个 Mobile Agent 在站点 A 申请的前往站点 B 的时间戳能且仅能被该 Mobile Agent 使用一次.

证明:假设该 Mobile Agent 在站点 A 申请的前往站点 B 的时间戳被该 Mobile Agent 使用了两次,不妨设站点 B 有相邻节点 $N_1,N_2,\ldots,N_K$,且 $A\in\{N_1,N_2,\ldots,N_K\}$,该 Agent 前往站点 B 所使用的时间戳为

$$stamp=stamp(B,m)=Base_{T1}+a_A\times m.$$

该 Agent 第 1 次前往站点 B 时,站点 B 的 FROM 表为

Host name	Remote stamp base	Stamp increase	Used stamp bitmap
N_1	$Base_{N1}$	a_{N1}	…
…	…	…	…
A	$Base_{T1}$	a_A	…
…	…	…	…
N_K	$Base_{NK}$	a_{NK}	…

其中 $a_{NK}>a_{NK-1}>\ldots>a_{N1}$.

该 Agent 第 2 次前往站点 B 时,站点 B 的 FROM 表为

Host name	Remote stamp base	Stamp increase	Used stamp bitmap
N_1	$Base_{N1'}$	a_{N1}	…
…	…	…	…
A	$Base_{T2}$	a_A	…
…	…	…	…
N_K	$Base_{NK'}$	a_{NK}	…

根据第 2.2 节的论述可知,当该 Agent 第 1 次前往站点 B 时,可以被接收.当该 Agent 第 2 次用同样的时间戳前往站点 B 时,会有两种情况:

(1) 当 $Base_{T1}=Base_{T2}$ 时,情况与反复攻击(replay attack)相同,站点 B 不会接收该 Agent.

(2) 当 $Base_{T1}\neq Base_{T2}$ 时,则 $Base_{T2}=Base_{T1}+BaseIncr\times n,(n\in Z^+,n\ge 1)$,其中 $BaseIncr=a_{NK}\times a_{NK-1}$;站点 B 接到该 $stamp$ 后进行验证,则 $stamp=Base_{T1}+a_A\times m\le Base_{T1}+a_A\times MAXNUM\le Base_{T1}+a_{NK}\times MAXNUM<Base_{T1}+a_{NK}\times a_{NK-1}=Base_{T1}+BaseIncr\le Base_{T1}+BaseIncr\times n=Base_{T2}$.

所以站点 B 拒绝接收发出此时间戳的 Mobile Agent. □

综上所述,采用增量时间戳技术能够防范恶意 MA 使用同一时间戳对执行环境进行重复攻击,或者使用发往服务器 A 的时间戳向服务器 B 迁徙(越权迁徙)(参见图 3 中的证明),以及使用过期的时间戳进行迁徙(参见图 4 中的证明),即能够防止不规范的恶意 MA 的行为,避免执行系统受到恶意 MA 的损害.

3 实 例

我们已在设计及实现移动代理系统时,实现并检验了互质增量时间戳理论.

该移动代理系统模型的实现是:硬件环境为 3 台 PC 机构成的局域网络,软件环境为 Lucent 公司开发的 Inferno V2.0 系统,以及基于 Inferno 系统的 Limbo 语言.

在图 2 中,A,B,C 分别为 3 个相邻的服务器,在每个服务器上要建立并维护两个表:FROM 表和 TO 表,分别用于核实从相邻服务器迁徙来的移动代理的时间戳和管理本地的时间戳.

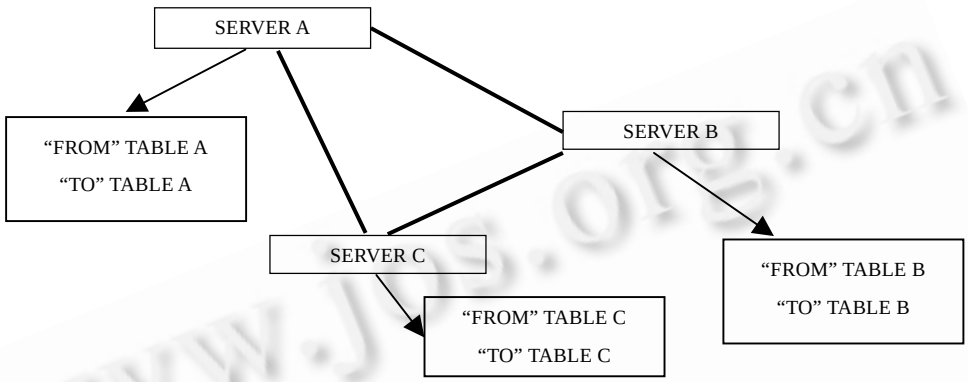

Fig.2 A structure sketch map of the incremental timestamp's example

图 2 增量时间戳实例的结构示意图

图 3 是在某一时刻,各服务器上的 FROM 表和 TO 表中记录的内容.

“FROM” TABLE A

HOST NAME	REMOTE STAMP BASE	STAMP INCREASE	USED STAMP BITMAP
B	1000	7	110100
C	2000	8	0100

“TO” TABLE A

LOCAL STAMP BASE	HOST NAME	STAMP INCREASE
500	B	11
	C	10

“FROM” TABLE B

HOST NAME	REMOTE STAMP BASE	STAMP INCREASE	USED STAMP BITMAP
A	500	11	110000000
C	2000	5	1000

“TO” TABLE B

LOCAL STAMP BASE	HOST NAME	STAMP INCREASE
1000	A	7
	C	12

“FROM” TABLE C

HOST NAME	REMOTE STAMP BASE	STAMP INCREASE	USED STAMP BITMAP
A	500	10	000000000
C	1000	12	110000

“TO” TABLE C

LOCAL STAMP BASE	HOST NAME	STAMP INCREASE
2000	A	8
	B	5

Fig.3 Sometime the contents of table FROM and table TO at servers A, B, C

图 3 某个时刻服务器 A,B,C 上的 FROM 表和 TO 表中的内容

以服务器 A 为例,在 A 的 FROM 表记录的第 1 行表示服务器 B 是本站点的一个相邻服务器,*B* 的当前时间戳基值(remote stamp base)是 1000,从 B 服务器发送到 A 站的所有时间戳的增量(stamp increase)为 7,已用时间戳位图(used stamp bitmap)表明在 B 服务器的任何时间戳基值,最多发给 A 站 6 个时间戳(6 个二进位表示 6 个时间戳).当前已收到的时间戳分别是 6 个时间戳中的第 1,2,4 个(对应于位图 110100).当 A 收到从 B 发来的第 i 个时间戳($1000+i\times7$)(此例中 $1\leq i\leq6$),且位图第 i 位此时为 0,则将其置为 1.若不为 0,则报告有攻击存在(即能防止重复攻击).表中的第 2 行记录着相邻服务器 C 的情况,分析类似.

服务器 A 在 TO 表中的内容说明,本地的时间戳基值是 500,对发向相邻站点 B 和 C 的时间戳增量分别为

11 和 10,则它向任何一个相邻站点发送的时间戳个数不能大于 9(即图 2 中定义的 MAXNUM=min(11,10)−1).当 A 向任何一个相邻站点发送的时间戳到达 9 时,该站点的服务器暂停发放新的时间戳,并等待所有已经获得时间戳的 MA 离开 A,然后在本地的时间戳基值上增加 110(即根据图 2 定义的 BaseIncr=10×11),将 TO 表中的本地时间戳基值(local stamp base)更新为 610,并通知相邻的服务器.相邻服务器 B,C 在收到 A 的新时间戳基值后,将服务器 A 在它们 FROM 表中的时间戳基值更新为 610,并清除对应的已用时间戳位图的所有位.

对于服务器 B,C 上的表和有关工作过程与 A 完全一致,在此不再一一说明.

4 和其他工作的比较

目前,大多数的 MA 系统是基于 Java 编程语言,其安全模型是建立在 Java 的安全模型上的.例如,Aglets 系统[6],Mole[7],MOA[8]以及 Concordia[2]等等.MOA 以及 Mole 并未在标准的 Java 功能上作任何的改进.IBM 的 Aglets 系统中将 MA 称为 Aglet,为了确保返回的 Aglet 的安全,Aglet 的拥有者(owner)定义了该 Aglet 的巡游路线,并限定了 Aglet 的能力(capability),如可消耗的 CPU 时间等等.三菱公司开发的 Concordia 系统中,在 Agent 的用户创建 Agent 时,即产生一个护照(passport),Agent 的身份由该 passport 决定,passport 中包含了所有与 Agent 相关的信息,如全局惟一的 Agent 标识、最大可用资源(资源访问权限)、用户信息、制造商信息等等.

还有一些 MA 系统是各大公司及学院开发的,如 Telescript 系统、TACOMA 系统等等.Telescript 是第 1 个商业化的 MA 系统,Agent 有一个权限(permit)集合来限定该 Agent 可以使用的 Telescript 指令和可用资源的数量.在 Troms 大学和 Cornell 大学联合开发的 TACOMA 系统中,基于对 Agent 的主体(principal)(即 Agent 的拥有者)的身份的认证来对 Agent 可访问的资源加以控制.

以上的各种 MA 系统都是通过限定 MA 的权限或巡游路线来限制 MA 的能力,但都不能有效地防范恶意 MA 的重复攻击以及越权攻击.本文所提出的互质增量时间戳机制在一定程度上较好地解决了这个问题.

5 结束语

Mobile Agent 是在网络范围中迁徙的计算机程序,有人把它与计算机病毒作比较.因此,提供一个安全的环境对于一个 Mobile Agent 系统是否实用是至关重要的.互质增量时间戳技术较好地解决了一些 Mobile Agent 系统的安全问题,这对于今后继续探索开发使用 Mobile Agent 系统有着良好的推动作用.

References:

[1] Ma, Jun-tao, Liu, Ji-ren. Architecture and technology of mobile agent system. Mini-Micro Systems, 1998,19(2):7～14 (in Chinese).

[2] Walsh, T., Paciorek, N., Wong, D. Security and reliability in Concordia. In: ElRewini, H., ed. Proceedings of the 31st Annual Hawaii International Conference on System Science (HICSS'31). Los Alamitos, CA: IEEE Computer Society Press, 1998. 44～55.

[3] Peine, H. Security concepts and implementation in the ara mobile agent system. In: ElRewini, H., ed. Proceedings of the 31st Annual Hawaii International Conference on System Science (HICSS'31), Los Alamitos, CA: IEEE Computer Society Press,1998. 236～242.

[4] Pham, Vu A. Karmouch, A. Mobile software agents: an overview. IEEE Communications Magazine, 1998,7(7):26～37.

[5] Farmer, W.M., Guttman, J.D., Swarup, V. Security for mobile agents: issues and requirements. In: Proceedings of the 19th National Information Systems Security Conference (NISSC'96), 1996. 591～597. http://csrc.nist.gov/ nissc/ 1996/ papers/ NISSC96/ paper033/ SWARUP96.PDF.

[6] Karjoth, G., Lange, D.B., Oshima, M. A security model for aglets. IEEE Internet Computing, 1997,1(4): 68～77.

[7] Baumann, J., Hohl, F., Strasser, M. Mole——concepts of a mobile agent system. Journal of World Wide Web, 1997,1(3):123～137.

[8] Milojicic, D., Laforge, W., Chauhan, D. Mobile objects and agents (MOA)——design, implementation and lessons learned. In: Proceedings of the USENIX Conference on Object–Oriented Technology and Systems (COOTS'98), Vol 4. New Mexico USENIX, 1998. 179～194. http://www.infosys.tuwien.ac.at/Research/Agents/ archive/ara-security.ps.gz.

附中文参考文献:
[1] 马俊涛,刘积仁.Mobile Agent 体系结构及关键技术探讨.小型微型计算机系统,1998,19(2):7～14.

Mobile Agent System by Incremental Timestamp Security Technique*

WAN Yan, SUN Yong-qiang, ZHU Xiang-hua, TANG Jin

(*Department of Computer Science and Engineering, Shanghai Jiaotong University, Shanghai* 200030, *China*)

E-mail: winniewan@mail.zte.com.cn

http://www.sjtu.edu.cn

Abstract: The security technology decides the realization of the mobile agent. The primary problem solved by this paper is to prevent replay attacks and acts beyond its authorities of malicious mobile agent. The concepts, characters and security of mobile agent are described. Afterwards, the concepts and advantages of incremental timestamp are proposed, and a specific example and the demonstrations are also presented. The mobile agent system that adopts incremental timestamp can protect execute system from damages of hostile mobile agents. The incremental timestamp technology has been verified and realized in the mobile agent system which is developed based on the Lucent Company's Inferno system.

Key words: mobile agent; incremental timestamp; security

* Received September 30, 2000; accepted April 3, 2001

全国办公自动化技术及应用学术会议

征 文 通 知

中国计算机学会办公自动化专业委员拟于 2003 年春在扬州召开办公自动化（OA）技术及应用学术会议，同时召开全体委员会议。现将有关事项通知如下:

一、征文范围

所有涉及 OA 技术和应用的动向、研究、开发和应用成果的论文，主要包括：(1) OA 现状及技术发展趋势;(2) 网络技术与分布式系统;(3) OA 中的信息安全技术;(4) OA 开发工具和 OA 语言;(5) 工作流/数据库、数据仓库和数据挖掘技术;(6) OA 中的多媒体技术;(7) 电子商务/电子政务;(8) 人工智能与决策分析在 OA 中的应用;(9) 其他.

二、来稿要求

1. 理论联系实际，具有学术和应用推广价值。未被其它会议、期刊录用或发表。

2. 来稿一般不得超过 6000 字（含图表），为了便于正式出版论文集，来稿必须附中、英文摘要、关键词及主要参考文献，注明作者姓名、工作单位、详细通讯地址（包括电子邮件地址）与作者简介。

3. 欢迎电子投稿，来稿一般不退，请自留底稿。

三、来稿地址

南京 东南大学计算机科学与工程系 胡苏宁； 邮编：210096；

电话：(025)3793044； E-mail：yangming@seu.edu.cn

四、重要日期

征文截止日期： 2002 年 10 月 15 日

录用通知发出日期： 2002 年 11 月 20 日