第 35 卷 第 2 期 计 算 机 工 程 2009 年 1 月
Vol.35 No.2 Computer Engineering January 2009
·安全技术· 文章编号：1000—3428(2009)02—0136—03 文献标识码：A 中图分类号：TP393.08

基于 Pass 和 Visa 的迁移实例认证研究

李 辉，王晓琳，曾广周
(山东大学计算机科学与技术学院，济南 250101)

摘 要：在迁移工作流管理系统中，迁移实例的移动性导致了身份认证和访问控制的复杂性。该文给出一种基于 Pass 和 Visa 的迁移实例认证和访问控制模型，模型采用迁移实例拥有者为其颁发护照、接收者为其签证的双重签名策略。接收者通过对迁移实例携带护照和签证的认证，可以有效识别迁移实例的身份并对其实施访问控制。迁移实例拥有者可根据 Visa 证书链有效追踪迁移实例的移动历史。
关键词：迁移工作流；安全；Pass 技术；Visa 技术

Authentication Research for Migrating Instance Based on Pass and Visa

LI Hui, WANG Xiao-lin, ZENG Guang-zhou
(School of Computer Science and Technology, Shandong University, Jinan 250101)

【Abstract】 In migrating workflow management, identity authentication of migrating instance presents some complicated features owning to migration of migrating instance between sites continuously. Combined with certificate, this paper proposes an authentication model based on Pass and Visa, which applies mutual signature strategy by creator of migrating instance issuing Pass for it and destination issuing Visa. Based on authentication mechanism of migrating instance with Pass and Visa, the destination can recognize the identity of migrating instance efficiently and endow some permission with it. The owner of migrating instance can trace the history path based on Visa certificate chain.
【Key words】 migrating workflow; security; Pass; Visa

1 概述

在迁移工作流管理系统中，迁移实例的移动性导致了身份认证和访问控制的复杂性。为此，文献[1]给出一种迁移工作流管理框架：迁移实例(下文均用 *mi* 表示)是业务活动的执行者，每个工作位置代表一个组织机构，为 *mi* 提供运行环境和工作流服务。*mi* 可以在不同的位置上创建并首先执行。发现当前位置不能满足其任务需求时，*mi* 可携带任务和当前结果迁移到一新位置继续执行。*mi* 在工作位置之间的连续迁移既可能引起恶意主机对 *mi* 的窥探，也可能导致恶意 *mi* 对主机的安全威胁。为了保护主机安全，需要对 *mi* 进行认证和访问控制。文献[2]提出一个基于数字签名的移动代理安全模型，应用数字签名对移动代理认证，可以避免未知移动代理在系统中的蔓延。文献[3]给出一个迁移工作流管理中工作位置的安全模型：模型采用单证书机制，将身份证作为安全加载的唯一依据。文献[4]将 Pass 和 Visa 作为移动 Agent 的官方旅行文件。

本文借鉴文献[4]，将 Pass 和 Visa 用于 *mi* 认证；并将授权包含进 Visa，以支持访问控制要求；为 *mi* 添加签证链 VisaPages，以支持对迁移路径的审计。基本思想是：*mi* 拥有者为其颁发带有签名的 Pass，然后利用 Pass 和 SRR 发出迁移查询；*mi* 接收者为其签发带有签名的 Visa 并对其操作授权形成任务-服务表和服务-资源表，将 Visa 作为迁移查询响应；比较得到的各个 Visa，选出最优和次优的 Visa，并分别以其所在位置作为目的和备份迁入位置，然后可将 *mi* 迁往目的位置，若迁入失败，则直接改迁至备份迁入位置，以有效降低迁移失败时再次迁移的代价；当 *mi* 到达某位置时，接收者可对 *mi* 携带的 Pass 和 Visa 进行认证，有效识别 *mi* 的身份和权限，以决定将 *mi* 迁入并提供 MIE；当 *mi* 成功迁入一个新位置时，将该位置提供的 Visa 包含进 VisaPages，这可有效追踪 *mi* 的移动历史，并作为各位置资源开放度的评价依据。

2 Pass 和 Visa

Pass 是 *mi* 的身份证书，在 *mi* 创建时由创建者为其自动生成。*Visa* 是 *mi* 的授权证书，表示工作位置(接收者)对 *mi* 的许可，需要 *mi* 向工作位置申请。

定义 1 *Pass*是一个七元组(*PassID, HolderInfo, issuer, issue_time, Time, Sign, Kpub*)，其中，*PassID*是护照的唯一标识；*HolderInfo*=(*miID, create_address, create_time, goal*)，是护照持有者(即迁移实例)的基本信息，包括*mi*的标识、创建地址、创建时戳和迁移目标；*issuer*是护照的签发者，即迁移实例的创建者；*issue_time*是护照的签发时戳；*Time*是护照的有效期限；*Sign*是签发者用私钥对*PassID*||*HolderInfo*|| *issuer*|| *issue_time*||*Time*的编码，其中，||表示字符串连接(下同)；$Kpub_{pass_issuer}$是护照签发者的公钥。

定义 2 *Visa* 是一个六元组(*VisaID, issuer, issue_time, Time, Task_serv, Serv_res, Sign*)，其中，*VisaID* 是签证的唯一标识；*issuer* 是签证的签发者(即接收 *mi* 的位置)；*issue_time* 是签证的签发时戳；*Time*≥0 是签证自签发时戳开始的有效期限，若 *Time*=0，则拒绝迁入； 停靠站收到查询请求后，根据服务能力进行访问控制，主要是 *Task_serv*(见图 1)和 *Serv_res*

基金项目：国家自然科学基金项目资助项目(60573169)；山东省科学技术发展计划基金资助项目(031110123)
作者简介：李 辉(1983－)，男，硕士研究生，主研方向：智能计算，协同技术；王晓琳，副教授；曾广周，教授、博士生导师
收稿日期：2008-07-30 **E-mail：**paraden_lihui_ne@163.com

(见图 2)。

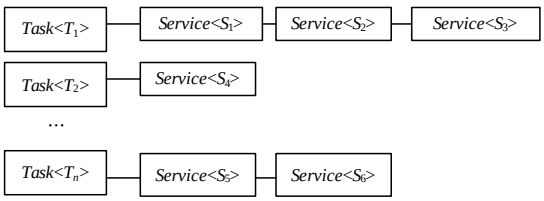

图 1　*Task-serv*

	Resource<R_1>	…	*Resource*<R_n>
Service<S_1>	R	…	R/W
…	…	…	…
Service<S_n>	R/W/C	…	R

图 2　*Serv-res*

*Task_serv*是动态表，由任务($T_1, T_2,\cdots, T_n$)和服务($S_1, S_2,\cdots, S_n$)组成，记录表示对任务$T_i(i=1,2,\cdots,k)$位置提供的服务$S_p(p=1,2,\cdots,n)$，即T_i的服务范围S。*Serv_res*是权限表，由服务($S_1, S_2,\cdots, S_n$)和资源($R_1, R_2,\cdots, R_m$)组成，记录表示每个服务S_p对资源$R_l(l=1,2,\cdots,m)$的操作权限，包括R(读)、W(写)、C(改)、F(完全控制)等。

Sign 是签证签发者用私钥对 *VisaID||issuer||issue_time||Time||Task_serv||Serv_res* 的编码。

定义 3 *VisaPages*是附在*Pass*上的签证链，记为：*VisaPages*=($Visa_1, Visa_2,\cdots, Visa_n$)，下标表示顺序。

3　认证和访问控制

约定：*Enc*(*m, Kpub*)为公钥加密，*Kpub* 为公钥，*m* 为明文；*Dec*(*c, Kpriv*)为私钥解密，*Kpriv* 为私钥，*c* 为密文。设 *mi* 当前位置是 *i*，查询后打算迁入位置 *jn*，如图 3 所示。

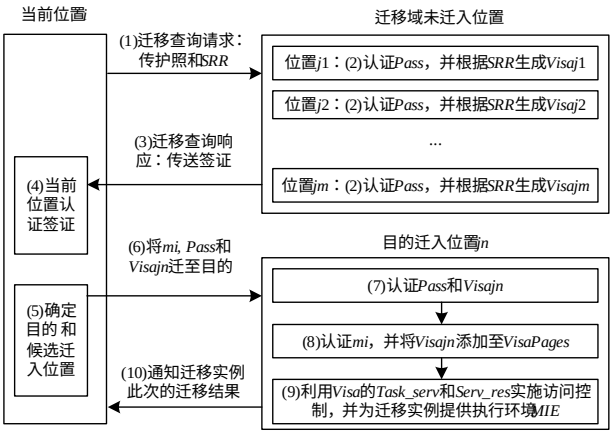

图 3　基于 *Pass* 和 *Visa* 的认证和控制

迁移过程如下：

(1)迁移查询$i\rightarrow j$: *Enc*(*Pass, SRR, Kpub*$_j$*, iID*)送j，即向j申请签证，其中，*iID*是位置i的标识，用于*Visa*回送时j对i的公钥选取。

(2)j：用 *PassAuth*(*Pass*)认证护照，并填写签证页，以表示同意 *mi* 迁入或拒绝，同时生成 *Task_serv* 和 *Serv_res* 表。

(3)迁移查询响应$j\rightarrow i$：*Enc*(*Pass||Visa, Kpub*$_i$)送i。

(4)i：比较各个签证，选出满足服务资源最多和次多的签证，并分别以其位置作为目的和候选迁入位置。

(5)$i\rightarrow jn$:若目的迁入位置为jn，则 *Enc*(*mi||Pass||Visa, Kpub*$_{jn}$)送jn，完成到j的迁移。

(6)*jn*：用 *VisaAuth*(*Pass||Visa*)认证迁入许可。

(7)*jn*：认证 *mi*，并将 *Visajn* 添至签证链 *VisaPages*。

(8)*jn*：利用签证中的任务-服务表(图 1)和服务-资源表(图 2)实施访问控制，若均通过，则激活 *mi*，否则丢弃。

访问控制过程如下：

设 *mi* 当前的任务为 *t*，服务需求为 *s*，资源需求为 *r*，访问表示为 *a*=(*t, s, op, r*)，表示任务 *t* 应用服务 *s* 对资源 *r* 执行操作 *op*，其中，*op*∈{R, W, C, F,…}。

Step1 *MIE* 接收 *mi* 的当前任务信息，并将 *a* 提交给 *SA*。*SA* 解析 *a*，并映射为本机服务信息 *wps*={*wt, ws, wop, wr*}，其中，*wt, wop, ws* 和 *wr* 分别为 *t, op, s* 和 *r* 在本位置的表示。*SA* 将 *wps* 提交给访问控制器(记为 *AC*)：

1)*AC*在访问控制信息库中检查*wt*是否在*Task_serv*的任务标志项中，如果存在匹配的任务标志t_i，转 2)；否则，转Step2。

2)*AC*检查*ws*是否在*Task_serv*表中标记t_i的服务范围*S*内，如果*ws*包含于*S*，转 3)；否则，转Step2。

3)*AC* 进一步检查 *Serv_res*，确定 *ws* 对 *wr* 的 *wop* 是否在权限矩阵规定的范围内，如果操作权限合法，转 4)；否则，转 Step2。

4)停靠站为 *mi* 提供服务，服务完后返回 *mi* 结果。

Step2 停靠站拒绝服务，将事件记入安全日志中，并通知管理机。

(9)$jn\rightarrow I$：通知 *mi* 迁移结果。

下面讨论 *Pass* 和 *Visa* 的认证算法及 *mi* 的访问控制算法。

设*Pass*签名者的公钥为$Kpub_{pass_issuer}$，认证位置为j并且*Pass*已由i用j的公钥$Kpub_j$加密，t_{curr}为j认证加密*Pass*的当前时间(见图 3)。

算法 1 *PassAuth*(*Pass*)

Begin
Dec(Pass,Kpriv$_j$);
If Dec(Pass.sign, Kpub$_{pass_issuer}$)=PassID||HolderInfo||issuer||issue_time||Time ∧ t_{curr} -Pass.issue_time < Pass.Time
{
Visa$_j$.Time ← 签证有效时间(>0);
Visa$_j$.Task_serv←任务和服务承诺;
Visa$_j$.Serv_res←服务和资源承诺;}
else { Visa$_j$.Time ← 0;　//拒绝迁入
Visa$_j$.Task_serv←空表;
Visa$_j$.Serv_res←空表;
}
End;

设*Visa* 的认证位置为i，*Visa*$_j$表示位置j对*Pass*持有者的签证，且*Pass||Visa*已经由j用i的公钥$Kpub_i$加密(见图 3)。

算法 2 *VisaAuth*(*Pass||Visa*)

Begin
Dec(Passp||Visa$_j$,Kpriv$_i$);
If Dec(Pass.sign, Kpub$_{pass_issuer}$) = PassID||HolderInfo||issuer||issue_time||Time
{If Dec(Visa$_j$.sign, Kpub$_j$)=VisaID||issuer||issue_time||Time||Task_serv|| Serv_res
{ if Visa$_j$.Time=0 ∧ Visa$_j$.Task_serv为空表∧ Visa$_j$.Serv_res为空表
放弃到位置 j 的迁移计划;
else Enc(MI||Pass||Visa$_j$, Kpub$_i$)送位置j;}　}
End;

4　系统设计

模块化结合分层化思想和组件技术，给出系统的具体框架。在实现上，密钥管理，P/V证书管理以组件的形式对外提供透明服务，使系统可重构和可扩展。在迁移工作流管理中，各位置的结构是完全对等的，停靠站的设计采用了微内核的架构[5]，底层提供了运行时支持服务，可将此框架作为安全插件动态部署在停靠站的系统服务之上，通过配置文件动态加载，与通信组件、目录组件等协调工作，实现*mi*的安全迁移。

4.1 迁出组件

(1)迁出准备

1)迁出查询：获取 *SRR* 和护照。

2)上下文检测：核查该 *mi* 是否仍与其他 *mi* 存在依赖性。

3)封装 *mi*：将 *mi* 的各组成部分打包封装。

4)加密 *mi* 等待迁出：用 IDEA 算法将 *mi* 加密成密文记为 *EncMI*，并保存加密密钥 *SynKey*。

(2)迁出执行

1)迁出会话与协商：利用 *SRR* 和 *Pass*，向 *mi* 的迁移域中未曾迁入的各个位置进行迁出会话与协商。

2)迁移决策：首先要验证各个签证的真实有效性，然后再选定通过验证的可提供服务资源最多和次多的两个签证，以其所在的位置分别作为目的和候选迁入位置。

3)获取签证：获取目的位置的签证，若失败，则获取候选位置的签证。

4)启动迁出：将 *EncMI*, *SynKey*, *SRR*, *Pass* 和 *Visa* 一起加密传送给目的迁入位置。

5)确认及失败处理：等待目的位置返回成功，若失败，则选择候选位置执行迁入；若也失败，则重新发出查询以获得新的目的和候选位置。

(3)注销管理

1)回收 *MIE*：回收成功迁出的 *mi* 的执行环境和运行时服务。

2)更新注册表：删除已迁出的 *mi*。

迁出模块体系结构如图 4 所示。

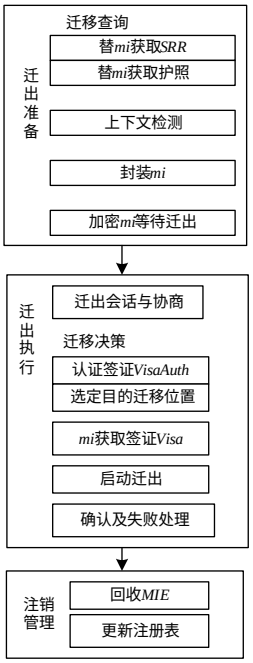

图 4 迁出模块体系结构

4.2 迁入组件

(1)迁移响应

利用迁出方的 *SRR* 和 *Pass* 进行迁入会话建立和协商，并据此颁发签证 *Visa*。

(2)身份认证

调用 *PassAuth* 认证护照，然后调用 *VisaAuth* 认证签证。

(3)访问控制

利用 *Visa* 中的 *Task_serv* 和 *Serv_res* 使 *mi* 只访问位置对其承诺的服务。

(4)加载管理

为 *mi* 生成运行时环境 *MIE*。

(5)注册 *mi*

将成功迁入的 *mi* 加入注册表中。

(6)迁入确认

向源位置发“迁移成功”确认消息，并向监控者报告 *mi* 的当前位置和状态。

迁入模块体系结构如图 5 所示。

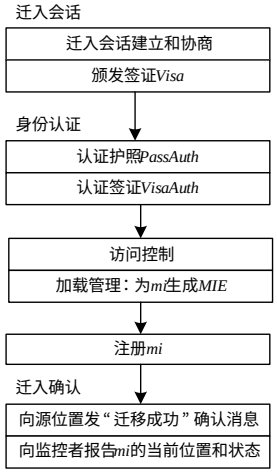

图 5 迁入模块体系结构

5 结束语

本文分别定义了一种 Pass 证书和 Visa 证书，并应用于迁移实例的认证和访问控制，特征如下：(1)证书颁发者用自己的私钥对证书签名，证书接收者用证书颁发者的公钥认证。(2)证书传输时，证书发送者用证书接收者的公钥加密，证书接收者用自己的私钥解密。(3)签证中包含接收位置对迁移实例的服务授权，因此，Visa 除具备门票的作用外，还有权限控制功能。(4)签证链不仅用于审计迁移实例的工作路径，还可作为评价各位置的服务承诺履行情况的依据。实验表明，上述机制在工作流管理应用中是有效的。

参考文献

[1] 曾广周, 党 研. 基于移动计算范型的迁移工作流研究[J]. 计算机学报, 2003, 26(10): 1343-1349.

[2] 胡 涛, 王汝传, 徐小龙. 基于数字签名的移动代理系统安全模型研究[J]. 计算机工程与科学, 2005, 27(12): 7-9.

[3] 李 琳, 曾广周, 熊云萍. 面向迁移工作流管理系统中工作位置的安全问题研究[J]. 计算机应用, 2006, 26(5): 1096-1098, 1105.

[4] Guan Sheng-Uei, Wang Tianhan, Ong Sim Heng. Migration Control for Mobile Agents Based on Passport and Visa[J]. Future Generation Computer System, 2003, 19(2): 173-186.

[5] 谢 浩, 王晓琳, 曾广周. 面向服务的柔性迁移工作流停靠站设计[J]. 计算机应用, 2006, 26(3): 685-687, 694.