基于多 Agent 的 CA 系统的设计与实现

缪祥华[1], 何大可[2]
(1. 西南交通大学 计算机与通信工程学院, 四川 成都 610031;
2. 西南交通大学 网络通讯安全与应用中心, 四川 成都 610031)

摘要: 认证中心是公钥基础设施的核心问题, 文章设计并实现了一个基于多 Agent 的认证中心系统. 文章首先设计系统的体系结构, 然后设计了多个Agent来完成认证中心的相应功能. 该系统将认证中心的任务分布到各相应的 Agent 上执行, 同时各Agent 之间可以通过认证中心相互协商、协调工作, 从而提高了认证中心的工作效率.

关键词: Agent; CA; 信息安全
中图分类号:TP393.08 文献标识码:A 文章编号: 1007-855X(2004)03-0040-04

Design and Implication of CA System Based on Multi-Agent

MIAP Xiang-hua[1], HE Da-ke[2]
(1. School of Computer & Communication Engineering, Southwest Jiaotong University, Chendu 610031, China;
2. Center of Network Communication Security & Application, Southwest Jiaotong University, Chendu 610031, China)

Abstract: It is known that certificate authority is the kernel of public key infrastructure. A certificate authority system based on multi-agent has put forward. The system structure is presented at first, and then a lot of agents which can deal with the tasks of the certificate authority are designed. The system distributes the tasks of the certificate authority to every corresponding agent, and at the same time all these agents can communicate with each other through the certificate authority, so its work efficiency will be much improved.

Key words: agent; CA; information security

0 引言

PKI(Public Key Infrastructure)提供了一个框架, 使人们可以在这个框架下实施基于加密的安全服务. 人们已经尝试了其他纯粹使用对称加密的基础设施, 但是因为遇到了管理、可伸缩性和生命周期等问题, 所以这些尝试都失败了. PKI 使人们可以创建鉴定和认证过程所需要的身份和相关信任, 管理基于公/私密钥的加密, 从而提供了一个比以前的加密和安全基础设施的可伸缩性强得多解决方案. 而 CA(Certificate Authority)是 PKI 的核心问题, 因此, 只要解决了 CA 问题, PKI 的问题也就迎刃而解了. 文中将 Agent 技术引入 CA 中, 设计了一个基于多 Agent 的 CA 系统, 用来解决 CA 的问题.

1 系统的功能要求

在 PKI 中主要有三种 CA[1]: 公共 CA, 企业内 CA 和外购企业 CA. 在本系统中的 CA 指公共 CA. 公共 CA 服务是使用认证机构向企业外或者组织外的普通人颁发证书, 证书的目的是为了在公共环境下证明身份. CA 不具备或者只具备有限的策略制定功能, 按照上级 PCA(Policy Certificate Authority)制定的策略, 担任具体的用户公钥证书的生成和发布以及 CRL(Certificate Revocation List)的生成和发布的职能. CA 的具体功能有: 发布本地 CA 对 PCA 策略的增补部分; 对下属各成员进行身份认证和鉴别; 产生和管理下属证书; 发布自身证书和上级证书; 证实 RA 的证书申请请求; 向 RA 返回证书制作的确认信息或已经制定好

收稿日期: 2003-09-08.
第一作者简介: 缪祥华(1972~), 男, 在读博士研究生, 讲师. 主要研究方向: 信息安全、密码学和人工智能. E-mail: jointhenet@yip.sina.com

的证书; 接收和认证对它所签发证书的作废申请请求; 对它所签发的证书产生 CRL; 保存证书、CRL、审计信息和它所签发的策略; 发布它所签发的证书和 CRL.

2 系统的设计

2.1 系统的体系结构

Agent 是一种具有自主性、交互性、移动性和智能性的软件主体[2]. Agent 的研究起源于人工智能、人机界面设计和面向对象编程, 它提供了一种新的分析、设计和实现复制软件系统的方法和一个通用、灵活的分布式计算模式. 根据系统的功能要求, 设计出该系统的体系结构如图 1 所示:

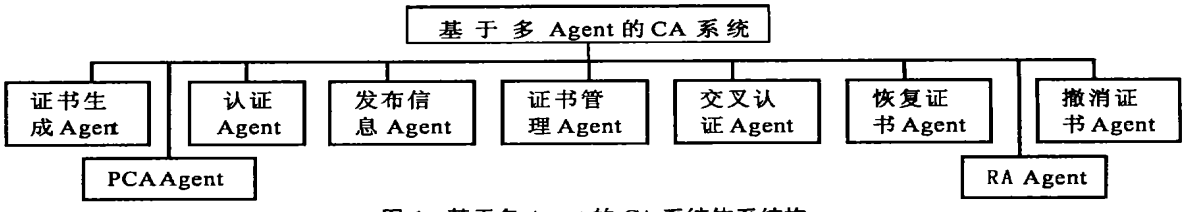

图 1　基于多 Agent 的 CA 系统体系结构

2.2 各 Agent 的设计

根据 X.509 证书的格式, 我们定义证书模板的数据结构如下:

```
class TCertificate//证书模板的数据结构[3]
{public:
    Version                 C_Version;              //证书版本号
    CertificateSerialNumber C_SerialNumber;         //证书序列号
    AlgorithmIdentifier     C_Signature;            //证书所用的签名算法
    Name                    C_Issuer;               //证书签发者名称
    Validity                C_Validity;             //证书的有效期
    Name                    C_SubjectName;          //证书主体名称
    PublicKeyInfo           C_PublicKeyInfo;        //证书的公钥信息
    UniqueIdentifier        C_IssuerUniqueID;       //证书签发者 ID
    UniqueIdentifier        C_SubjectUniqueID;      //证书主体 ID
    Extension               C_Extension;            //证书的扩展信息
};
```

定义各 Agent 与 CA 的通信、下级 CA 和用户的通信的数据结构如下:

```
class     CommunicateMSG//进行通信的数据结构
{public: //各 Agent 和 CA 的通信方法
   Boolean   SendCAMSG(MessageType   MSG); //向 CA 发送消息
   Boolean   ReceiveCAMSG(MessageType   MSG); //接收 CA 发来的消息
private: //各 Agent 和下级 CA 或用户的通信方法
   Message   ReceiveSubordinateMSG(MessageType MSG); //接收下级 CA 的消息
   Boolean   SendSubordinateMSG(MessageType MSG); //向下级 CA 或用户发送消息
friend   class   RegistrationAuthorityAgent; //定义 RA Agent 为友元类
friend   class   CertificateAgent; //定义认证 Agent 为友元类
friend   class   RepealCertificateAgent; //定义撤消证书 Agent 为友元类
```

```
friend class ResumeCertificateAgent; // 定义恢复证书 Agent 为友元类
friend class IntercrossCertificateAgent; // 定义交叉认证 Agent 为友元类
}
```

2.2.1 证书库

```
class CertificateDB: public TCertificate // 证书库的数据结构
{private:
  BIT C_Flag; // 证书的标记,标识证书的状态(有效、过期和撤消)
friend class CertificateManageAgent; // 定义证书管理 Agent 为友元类
};
```

2.2.2 证书管理 **Agent**

```
class CertificateManageAgent: public CommunicateMSG // 证书管理 Agent 的数据结构
{public:
  SelectResult CertificateSelect(ParameterType parameter/, ...); // 根据参数查询证书库
  UpdateResult CertificateUpdate(ParameterType parameter/, ...); // 根据参数修改证书
  DeleteResult CertificateDelete(ParameterType parameter/, ...); // 根据参数删除证书
  SaveResult CertificateSave(ParameterType parameter/, ...); // 根据参数保存证书
  BackupResult CertificateBackup(ParameterType parameter/, ...); // 根据参数备份证书库
  RestoreResult CertificateRestore(ParameterType parameter/, ...); // 根据参数恢复证书库
private:
};
```

2.2.3 PCA Agent

```
class PCA Agent: public CommunicateMSG // PCA Agent 的数据结构
{public:
  Boolean SendMSG(MessageType MSG); // 向 PCA 发送消息
  Boolean ReceiveMSG(MessageType MSG); // 接收 PCA 发来的消息
private:
};
```

2.2.4 RA Agent

```
class CertificateApply Agent: public CommunicateMSG // RA Agent 的数据结构
{public:
  Boolean CheckupMSG(MessageType MSG); // 审查用户的信息是否真实
  Boolean CertificateGrant(TCertificate Certificate); // 向下级 CA 或用户发放证书
private:
};
```

2.2.5 证书生成 **Agent**

```
class CertificateCreat Agent: public CommunicateMSG // 证书生成 Agent 的数据结构
{public:
  TCertificate AutoCreatCertificate(ParameterType parameter/, ...); // 证书自动生成
private:
};
```

2.2.6 认证 **Agent**

```
class CertificateAgent: public CommunicateMSG // 证书认证 Agent 的数据结构
{private:
```

```
};
```

2.2.7 撤消证书 **Agent**

```
class  RepealCertificateAgent: public CommunicateMSG // 撤消证书 Agent 的数据结构
{private:
};
```

2.2.8 恢复证书 **Agent**

```
class  ResumeCertificateAgent: public CommunicateMSG // 恢复证书 Agent 的数据结构
{private:
};
```

2.2.9 交叉认证 **Agent**

```
class  IntercrossCertificateAgent: public CommunicateMSG // 交叉认证 Agent 的数据结构
{public:
  Messgae  ReceiveMSG(MessageType MSG); //接收同级 CA 发来的交叉认证消息
  Boolean  SendMSG(MessageType  MSG); // 向同级 CA 发送交叉认证消息
private:
};
```

2.2.10 发布信息 **Agent**

```
class  IssuanceInfoAgent: public CommunicateMSG //发布信息 Agent 的数据结构
{public:
  Boolean  IssuanceInvalidationCertificate(TCertificate Certificate1, …); //发布失效证书
  Boolean  IssuanceRepealCertificate(TCertificate Certificate1, …); //发布撤消的证书
  Boolean  IssuanceResumeCertificate(TCertificate Certificate1, …); //发布恢复的证书
  Boolean  IssuanceSafetyStrategy (String SS); //发布 PCA 制定的安全策略
private:
}
```

3 系统的实现

根据上述的设计, 我们决定用 B/S 模式来实现该系统, 操作系统用 Windows XP, WEB 服务器用 IIS V5.1, 数据库服务器用 MS SQL SERVER 7.0, 编程语言用 ASP. 建立了一个证书库, 用于分类存储正在使用的证书、已经失效的证书和撤消了的证书, 用 ASP 编写了各 Agent 对应的程序来完成相应的功能. 通过数据的测试, 这个系统能够满足 CA 的功能要求.

4 结束语

在信息技术高速发展的今天, 电子商务如雨后春笋般的发展, 但安全性问题严重制约了电子商务的普遍应用. 本文首先分析了一个 CA 系统应完成的基本功能, 然后设计出了该系统的体系结构和各模块。该系统将认证中心的任务分布到各相应 Agent 上执行, 各 Agent 之间可以通过认证中心相互协商、协调工作, 从而提高了认证中心的工作效率。

参考文献:

[1] PKI 实现和管理电子安全[M]. 张玉清, 等译. 北京: 清华大学出版社, 2002, 114~118.

[2] 贾志勇, 等. Agent 互操作性研究[J]. 计算机科学, 2003, 30(2): 147~150.

[3] 陈波, 等. 公钥基础设施 CPKI 系统的设计与实现[J]. 计算机工程与科学, 2002, 24(5): 30~33.