KeyNote 信任管理系统
KeyNote Trust Management System

孙美凤[①,②]　　龚 俭[①]
Sun Meifeng　　Gong Jian
（msun、jgong）@njnet.edu.cn

①东南大学计算机系 南京 210096
Computer Science and Technology Department, SouthEast University, NanJing, 210096
②扬州大学工学院　扬州
Engineering College,YangZhou University, YangZhou

摘要：由 Blaze 等人提出的信任管理方法具有灵活、通用的特点，适合解决开放的、分布式应用的授权问题。本文简单介绍了信任管理方法的概念和相关工作，重点描述了 KeyNote 信任管理系统的实现规范，试图推动信任管理方法在各种网络服务中的应用。

Abstract：The trust management approach , introduced by Blaze, is a unified and flexible approach that allows direct authorization of security-critical actions in open and distributed applications. This paper gives an overview of trust management and a brief discussion of related works, describes the implementation specification of KeyNote trust management system, and aims to facilitate the use of trust manage approach in a variety network services.

关键词：信任管理系统，网络安全，KeyNote
Keywords：trust manage system, network security, KeyNote

1 背景

现有的和正在出现的大量应用都需要解决一个共同的问题：授权。传统的授权机制分两步进行：鉴别和存取控制。鉴别回答“是谁提出请求？”，存取控制回答“请求者是否有权完成请求的行为？”。这种源于操作系统的方法因其概念简单继续在集中式或小规模的分布式应用中使用。然而开放的、动态的 Internet 服务与集中式应用有显著的不同特性：

保护内容：在一个传统的客户/服务器环境中，需要保护的资源集中在服务器方。服务器鉴别并区分用户，客户无条件地信任服务器。在今天的 Internet，用户访问许多服务器，提出各种请求，并拥有自己的资源（个人信息、信用卡号），这样的用户不会信任任何服务器。事实上，一个简单的 Web 游览就是充满风险的行为：下载的软件携带病毒、材料不真实、个人信息泄露等等。从授权的角度，任何实体既可能是请求者，也可能是授权者。

委托：在大规模、动态变化的网络环境中，有大量的潜在的用户。Internet 服务（如 Internet 商店[1]）既无法预知请求者，也不能预知潜在的风险行为。因此，授权更多的委托给第三方进行，委托简化了系统的设计并使系统具有良好的可扩展性。考虑一个现实的电子银行系统的贷款规则：a) 本系统只处理 $10,000 以下的贷款并需要主管行长的签名；b)主管行长允许负责贷款的部门的各个职员直接办理￥500 以下的贷款；c)￥500~￥1000 之间的贷款业务至少需要两个职员的共同签名。可见应用的政策和委托关系可能很复杂。

授权信息的存储：传统的授权信息即存取控制表由服务方保存并管理。Internet 服

[1] 本文受国家自然科学资金(90104031)资助
孙美凤、1970、女、博士生、计算机网络安全，龚俭、教授、博士生导师

务发展很快并且不能预知潜在的用户和行为，决定了授权信息将动态创建、管理并分布式存放。授权者为了评估请求，需要通过网络收集足够的证书。为了防止证书被篡改和冒充，新的授权体系必须包含公钥签名机制。

因为 Internet 服务所具有的新属性，基于 ACL 的授权机制不再满足要求。 Blaze 在[1]文中首先提出了“信任管理方法”，该方法直接授权行为，统一表示安全政策、委托书（以下称证书）以及信任关系，形式证明请求与本地政策的一致性，具有灵活、通用、可靠的特点，全面解决开放分布式应用的授权问题 。

本文在第 2 部分简单介绍信任管理方法，重点在第 3 部分介绍 KeyNote 系统的概念和方法，最后是结束语。

2 信任管理

2．1 信任管理的内容

按照[1]的观点，信任管理方法的内容可归纳为以下三点：

（1）对等的授权模型

信任管理方法使用对等的授权模型，每个实体既可能是请求者也可能是授权者。作为授权者，实体维护本地政策作为授权决策的最后依据；作为请求者，实体维护一组证书并在请求时作为证据提交证书，供授权者参考。比较 X.509 公钥体系（隐含通信双方有相同的信任根），对等的授权模型没有任何隐含的信任假定，信任模型完全由本地控制，更灵活、更具延展性。

（2） 编程的授权证书

X.509 公钥体系使用“身份”证书绑定公钥和主体名字。“身份”证书使用名字标识“身份”，符合人们在物理世界的思维习惯。事实上，作为一个新的生存空间，数字世界有新的规则，许多新事物在物理世界是没有原像的。Internet 服务的授权者也许根本不认识请求者，在授权决策时回答“是谁提出请求？”是毫无意义的。授权证书是公钥和权力的绑定，适合描述数字世界形成的信任关系。首先，信任管理方法使用授权证书，其授权过程回答的问题是：“证书集合 C 证明了请求 R 符合本地安全政策 P 吗？”；其次，信任管理方法是编程的授权证书，授权项允许是通用程序，从而可以描述数字世界的复杂信任关系。证书编程的思想给网络应用带来两点好处：a．统一表示政策、证书和信任关系，b．以一种可理解的、一致的、灵活透明的方式处理安全。

（3）通用的证明机制

按照信任管理方法，信任管理系统应是一个通用的、与应用独立的查询引擎，其一致性检查算法应避免涉及应用的语义，寻找请求服从政策的形式化证据，即形式证明。实现通用的信任管理引擎的好处：a．授权的可靠性，保证授权决策仅仅依靠输入而不是任何隐含政策（bug）；b．程序可以复用，减少重复劳动。

2．2 信任管理的实现模型

从实现的角度，信任管理系统类似查询引擎，它接收请求 R、证书集 C、政策 P 作为输入，返回支持或拒绝请求的建议 Approve 或 Reject。信任管理的实现模型如图 1 所示 ：

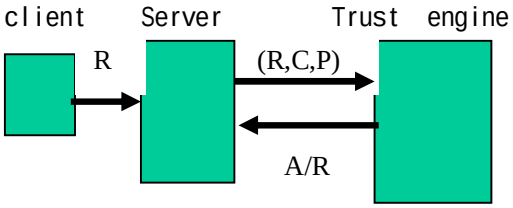

图 1 信任管理系统的实现模型

设计和实现信任管理系统需要考虑三方面因素：a．如何定义请求服从政策的“一致性证据”，b．政策和证书是完全编程或部分编程，使用何种语言？，c．在应用和信任管理引擎之间如何分工？特别是证书收集和签名验证由谁来完成？在信任管理方法提出以后，相继出现了 PolicyMaker 、KeyNote 、REFEREE、 DL 等，因为设计目标的不同这些系统对这三个问题有不同的回答。

PolicyMaker :PolicyMaker 是由 Blaze 等人设计的第一代信任管理原型系统。首先，PolicyMaker 不坚持特定的编程语言；其次，PolicyMaker 的设计目标是最小化，因此证书收集和签名验证是应用程序的责任；最后，PolicyMaker 的证据观点是形式化、可分析和证明的[2]。

KeyNote: KeyNote 是第二代信任管理系

统，它沿用了 PolicyMaker 的大部分思想和原则。KeyNote 的主要特点是使用了专门语言，KeyNote 语言简练并功能强大，同时和一致性检查算法紧密结合在一起。keyNote 语言和一致性算法的基本内容是本文重点，将在第三部分介绍。

REFEREE[3]：REFEREE 集中解决 web 应用的信任问题，其基本原则就象项目名称（Rule-controlled Environment For Evaluation of Rules, and Everything Else）所表达的：一切都在政策控制之下，包括证书收集和签名验证。按照[3]，信任管理系统评估请求的过程本身就是危险的：通过网络收集的证书会被篡改、冒充；因为证书是程序，执行证书和执行未知程序有相同的危险。REFEREE 并不试图消除危险，而是将危险置于政策控制之下，其信任管理系统是应用相关且不可形式描述和证明正确性的。

3 Keynote 信任管理系统[4]

3.1 一致性证据观点

KeyNote 沿用了 PolicyMaker 的一致性证据观点，其通用形式是：

输入：请求 R，证书集合$\{(f_0, POLICY), (f_1, s_1), \ldots, (f_{n-1}, s_{n-1}), (f_n, s_n)\}$，$(f_i, s_i)$是 PolicyMaker 的证书，$s_i$是发证者，$f_i$编码了持证者和权力。

问题：是否存在证书的有限序列 $i_1, i_2, \ldots, i_t$，其中 $i_j \in \{0,1,\ldots,n-1\}$且 i_j可以相同并无须穷尽$\{0,1,\ldots,n-1\}$，最终证明或拒绝了 R。显而易见，该问题的通用形式是不可判定的。

KeyNote 坚持单调的证据观点。单调性要求证书只能累加而不能去除证据，即没有撤消授权的观点。在此限制下，政策和信任关系可以表示成一个有向图（w, v, f）。W 是有向图结点的集合，每个结点代表一个或多个主体；v 是边的集合，边代表主体间的信任关系；权 f 给出信任条件。我们可以找到算法计算政策对请求的支持度。

3.2 KeyNote 语法

（1）行为属性

应用提交给信任管理系统评估的请求表示为一组名值对（类似于 shell 变量），被称为行为属性。行为属性的名、值是任意的字符串，其语义由应用解释，并在应用与使用它的证书间达成一致。_MIN_TRUST、_MAX_TRUST、VALUES 是系统的保留字，用于规定信任计算的返回值，_MIN_TRUST 给出请求与政策背离时的返回值，_MAX_TRUST 给出请求与政策一致时的返回值，_VALUES 是按照请求与政策的一致程度从高到低的返回值的序列。_ACTION_AUTHORIZERS 规定提交请求（直接支持请求）的主体名。_APP_DOMAIN 规定应用域的名称。

（2）证书和政策语言

在KeyNote系统中，证书和政策统称为声明。声明由若干域组成，其中：Authorizer 域规定授权者即声明的签发者；Licensees域规定被授权者，被授权者可以是一个或多个主体；Conditions域规定授权条件，Signature 域规定签名。

Licensees域的多个主体可以通过“&&”、“||”、“k-of(用“，”分隔的主体列表)”运算符连接，代表权力的不同分布。

Conditions是定义在行为属性集上的谓词，它利用“->”连接测试条件和返回值，测试条件是标准的正规表达式并可以嵌套，返回值缺省为_MAX_TRUST。政策声明和证书声明的区别是政策声明的授权者是“POLICY”，又因为政策是本地存储的无需签名保护。

3.3 信任的计算模型

首先给出以下定义：

定义1：政策支持度指请求服从本地政策的程度，它是名为“POLICY”的主体支持度。

定义2：主体支持度指主体支持请求的程度，它是主体的直接授权值与主体签发的声明的支持度的最大值。

定义3：直接授权值，提交请求的主体的直接授权值是_MAX_TRUST；否则_MIN_TRUST。

定义4：声明支持度是声明的持证者支持度与声明的条件支持度的最小值。

定义5：持证者支持度是声明的所有授权

对象的主体支持度按“||”(或)、“&&”（与）“k-of(<list>)”（列表中的任意k个主体）按运算符的语义计算所得。结果是“||”取较大值，“&&”取较小值，“k-of(<list>)”取列表中主体支持度的第k个最大者。

定义6：条件支持度是声明的条件域的所有测试条件的返回值的最大者。

3.4 举例

这里是引言中提到的电子银行的例子，本文再次利用它说明KeyNote信任管理系统的概念和方法。

A.政策规定金额在$10,000以下的贷款全权委托给RSA:dab212（主管行长）超出$10,000的开销本系统不于办理。声明中@dollars代表请求金额。

```
Authorizer: “POLICY”
Licensees: “RSA:dab212”
Conditions: (app_domain==”SPEND”) &&
(@dollars < 10000);
```

B.RSA:dab212 继续委派金额小于$7,500的贷款必须DSA:feed1234（部门经理）和职员列表中的任一公钥的共同签名,并当金额大于$2,500时要求记帐。

```
Authorizer: “RSA:dab212”
Licensees: “DSA:feed1234” && “RSA:abc123”
           ||”DSA:bcd987”
           ||”DSA:cde333” )
Conditions: (app_domain==”SPEND”) ->
{ (@(dollars) < 2500)-> _MAX_TRUST;
(@(dollars) <7500)->“ApproveAndLog”;};
Signature: “RSA-SHA1:9867a1”
```

C. 政策规定小于$1,000的贷款只要任意两个公钥的签名，它们是部门经理或职员的公钥。

```
Authorizer: “POLICY”
Licensees: 2-of(“DSA:feed1234”,
“RSA:abc123”, “DSA:bcd987”,
“DSA:cde333”)
Conditions: (app_domain==”SPEND”)
&&(@(dollars) < 1000);
```

我们看到KeyNote语言简练且表达能力强，以上政策和信任关系是ACL机制无法实现的。

假设现有DSA:feed1234、 DSA:978add共同签名、金额为$5，000的贷款请求处理：

```
_ACTION_AUTHORIZERS = “DSA:978add”
_ACTION_AUTHORIZERS = “DSA:978add”
app_domain = “SPEND”
_VALUES = {Approve, ApproveAndLog,
Reject}
```

dollars = “5000”, 代表请求金额，然而信任管理引擎无须理解语义。

信任的计算过程如下：

1. V(“DSA:978add”)=Approve；#直接授权者
2. V(“DSA:feed1234”)=Approve;#直接授权者
3. V(声明B)=ApproveAndLog; # 持证者支持度与条件支持度的最小值
4. V(“RSA:dab212”)=ApproveAndLog;
5. V(声明A)=ApproveAndLog;
6. V(“POLICY”)=ApproveAndLog;

4 结论

“信任管理”的概念出现以后，得到了安全研究领域的广泛关注。与传统的、基于ACL的授权机制相比，信任管理方法直接面向开放、分布式应用的授权问题，它具有灵活性、可扩展性和可靠性的特点。KeyNote作为第二代信任管理系统,它的最大贡献是给出了简练且富于表达能力的KeyNote语言，该语言与一致性检查算法紧密结合在一起。我们同时看到KeyNote只是一个原形系统，还有许多不足之处：（1）委托是任意的但应有限定的深度，而KeyNote语言不支持委托深度的规定，并且“||”、“&&”、“k-of”运算不足以表达更为复杂的授权对象；（2）在评估一个请求时系统做出的决策也许是不一致的，KeyNote没有给出冲突的解决方案；（3）KeyNote的形式证明建立在单调性证据的基础之上,系统需要类似于证书撤消的功能；（4）一个开放的、分布式的应用系统也许有很复杂的信任关系，如何收集、管理证书并确保在请求时提交足够的证书是系统必须考虑的问题。

参考文献

1. M.Blaze, J.Feigenbaum, J.Lacy, “Decentralized trust management”, *In Proc.of 17th Symposium on Security and Privacy*,pages 164-173, IEEE

Computer Society Press, Los Alamitos,1996.
2. M.Blaze, J.Feignbaum, M.Strauss, “ Compliance Checking in the PolicyMaker Trust Management System “, *In Proc. 2nd Financial Crypto Conference*,pages 251-265. Springer,Berlin,1998. LNCS 1465.
3. Y.-H.Chu, J.Feigenbaum, B.LaMacchia, P.Resnick, “ REFEREE:trust management for Web applications “, *Computer Networks and ISDN systems,29(8-13)*:953-964,Sept.1997.
4. M.Blaze,J.Feigenbaum,J. Ioannidis, and A. Keromytis, “The KeyNote Trust-Manage System “, RFC 2704.
5. S.Weeks, “Understanding Trust Management Systems “, *In Proceedings of IEEE Symposium on Security and Privacy,*2001,pages 94-101.
6. N.-H. Li,J.Feigenbaum, “A Logic-based Knowledge Representation for Authorization with Delegation “, HYPERLINK "http://www.research.ibm.com/" http://www.research.ibm.com/ .
7. 龚俭，陆晟，王倩，计算机网络安全导论，东南大学出版社,2000。