(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 1832395 B
(45)授权公告日 2011.06.08

(21)申请号 200610004474.9
(22)申请日 2006.02.13
(30)优先权数据
11/077,920 2005.03.11 US
(73)专利权人 微软公司
地址 美国华盛顿州
(72)发明人 C·F·罗斯三世 G·科斯塔尔
M·帕拉马斯万姆 R·N·潘迪亚
S·C·科特里勒 V·K·拉弗拉
V·亚尔莫莱恩克 钟昱晖
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 张政权
(51)Int.Cl.
H04L 9/08(2006.01)
H04L 9/32(2006.01)
(56)对比文件
US 6816965 B1,2004.11.09,
CN 1585326 A,2005.02.23,
US 6675296 B1,2004.01.06,
审查员 王加新

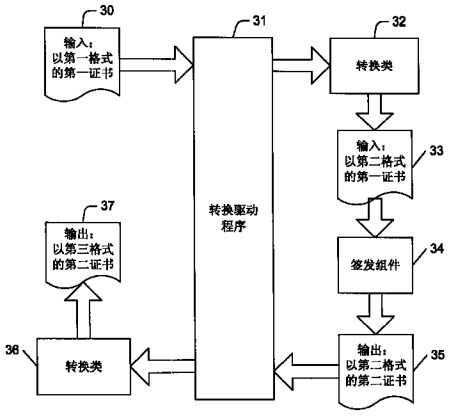
权利要求书 1 页 说明书 18 页 附图 7 页

(54)发明名称

用于签发证书的格式不可知的系统和方法

(57)摘要

一种经改进的证书签发系统可包括一证书转换引擎,该证书转换引擎用于将输入的证书和证书请求从第一格式转换成第二格式。证书签发引擎然后可用通用格式在输入请求上进行操作。签发引擎可根据其证书签发策略向客户机签发证书。该策略可用策略表达语言表达为可在运行时间消费的数据,它提供签发策略的灵活和有效改变。所签发的证书可被转换回由请求客户机消费的格式。这种转换可在将证书传送给请求客户机之前由转换引擎执行。



CN 1832395 B

1. 一种用于响应于客户机请求产生证书的证书签发系统，包括：

转换组件，用于接收第一格式的第一证书并将所述第一证书转换成第二格式，所述第二格式是通用格式；

签发组件，用于接收来自所述转换组件的经转换的第一证书，通过对来自所述经转换的第一证书的信息和证书签发策略作比较，来确定所述客户机是否有权得到第二证书，并用于在所述经转换的第一证书确定所述客户机具有所述证书签发策略所需的至少一个凭证时产生按所述第二格式的所述第二证书，并将所述第二证书发送到所述转换组件，

其中，所述转换组件将所述第二证书从所述第二格式转换为第三格式，并发送经转换的第二证书。

2. 如权利要求 1 所述的系统，其特征在于，所述签发组件包括用于与实现证书签发策略的一般表达的定制组件通信的接口。

3. 如权利要求 1 所述的系统，其特征在于，所述签发组件包括用于与实现证书签发策略的扩展的定制组件通信的接口。

4. 如权利要求 1 所述的系统，还包括用于向所述客户机传送所述证书签发策略的组件。

5. 如权利要求 1 所述的系统，其特征在于，所述客户机身份在所述第一证书确定所述客户机具有所述至少一个凭证时得到认证。

6. 如权利要求 1 所述的系统，其特征在于，所述客户机授权在所述第一证书确定所述客户机具有所述至少一个凭证时得到确定。

7. 如权利要求 1 所述的系统，其特征在于，所述证书签发策略包括对所述第二证书的形式的要求，且其中所述签发组件根据所述形式产生所述第二证书。

8. 一种用于向客户机签发安全证书的方法，包括：

接收按第一格式的第一证书并将所述第一证书转换成第二格式，所述第二格式是通用格式；

通过对来自所述经转换的第一证书的信息和证书签发策略作比较，来确定所述客户机是否有权得到第二证书；以及

如果所述经转换的第一证书确定所述客户机具有证书签发策略所需的至少一个凭证，则产生按所述第二格式的所述第二证书，

将所述第二证书从所述第二格式转换为第三格式，并发送经转换的第二证书。

9. 如权利要求 8 所述的方法，其特征在于所述第三格式和所述第一格式是相同的格式。

10. 如权利要求 8 所述的方法，还包括与定制软件组件通信，以实现所述证书签发策略的扩展。

11. 如权利要求 8 所述的方法，其特征在于，客户机身份在所述第一证书确定所述客户机具有至少一个凭证时得到认证。

12. 如权利要求 8 所述的方法，其特征在于，产生所述第二证书包括从所述证书签发策略中确定对所述第二证书的形式的至少一个要求。

用于签发证书的格式不可知的系统和方法

[0001] 相关申请

[0002] 本申请涉及申请号为＿＿＿＿于2005年2月28日提交的[律师案号为MSFT-4736/311539.01]题为“用于签发证书的可扩展的数据驱动系统和方法”(ExtendableData-Driven System and Method for Issuing Certificates)的美国专利申请。

技术领域

[0003] 本发明涉及计算安全，尤其涉及使用数字证书来向服务器认证客户机并确定客户机许可，还特别涉及证书签发系统中这些证书与通用格式之间的转换。

背景技术

[0004] 证书是证明某物或某物所有权的真实性的文档。在计算的世界中，数字证书提供各种功能。例如，数字证书可通过确定某实体实际上是所声称之物来认证该实体数组证书可通过确定－实体有权访问受限源而授权该实体。数字证书还可用来以防伪方式捕捉“策略”，例如授权策略、信任策略等。

[0005] 证书是非常有用的，并且在当前更为有用。安全策略的表达和实现是一种越来越重要的企业能力。证书格式的数量也在激增。当前可用的一些较流行证书格式是X.509、安全性断言标记语言(SAML)安全令牌、XrML 1.2和MPEG-REL。注意MPEG-REL具有众多变体并有众多名称，包括XrML 2.x、MPEG ISO-REL和ISO-REL。首字母简略词MPEG-REL在此使用时指向以上所列变体的至少全部。

[0006] 为了说明以上示例性证书的各种功能和格式，X.509证书遵守它们自己的格式且通常表示身份。SAML证书遵守它们自己的XML模式并广泛用于联合身份解决方案。XrML 1.2和MPEG-REL表达资源的使用策略并遵守它们自己的XML模式。

[0007] 现在存在生产并消费证书的服务和产品。然而，当新类型的证书开始流行时问题产生了。目前，消费特定格式证书的证书签发系统可能与其它格式的证书并不兼容。在最佳情况下，这可导致客户机尝试获取适当格式化的证书时的无效，或者需要客户机事先确定服务器要求哪种证书格式。在最差情况下，它导致互操作性失败。

[0008] 可实现的一种可能解决方案是维持能处理不同格式证书的多个并排的证书签发服务器。不幸的是，该解决方案使证书签发系统的实现和更新更为困难。实现并维持多个系统所需努力随着所使用的每个附加证书签发器倍增。

[0009] 现有证书签发系统的另一弱点是难以更改可签发证书的情形，即“证书签发策略”。在现有系统中，策略用证书签发系统二进制码表达为经过编译的算法，或者表达为配置参数的经具体建模的“脆弱”集合。改变该实现策略需要重新编码、重新编译、并重新部署新的证书签发系统。因而，为实践起见，证书签发策略受限于证书签发系统编程人员所预想的内容。为了改变该策略，证书签发系统可能不得不全部重新编码。要实现它会花去产品开发组大量的时间和精力。

[0010] 因此，本行业中有未满足的提供证书签发中的更多互操作性、以及便于改变证书签发策略的需要。

发明内容

[0011] 考虑本领域中以上所述的缺点，本发明提供一种经改进的证书签发系统和由这种经改进系统实现的方法。

[0012] 在此提供的证书签发系统可包括一转换组件，用于将输入证书转换成通用格式并用于将输出的生成证书转换成任何支持格式。因而，该系统可被描述为格式不可知的。实现通用证书签发策略的单个证书签发组件可对以各种格式到达的证书进行操作。由签发组件签发的证书还可在将这些证书传送给请求客户机之前转换成各种格式。

[0013] 该系统还可包括用于表达证书签发策略的全新设备。策略可用标记策略表达语言表达，并存储于例如由证书签发系统在运行时间消费的文件中。策略因而可通过改变该文件来简便地改变。还可提供用于扩展证书签发系统的能力的某些技术，因此该系统可应用并实现新的签发策略。

[0014] 本发明的其它优点和特征如下所述。

附图说明

[0015] 根据本发明的用于签发证书的系统和方法参照附图作进一步描述，在附图中：

[0016] 图 1 是广泛表示适于结合本发明各方面使用的示例性计算设备的基本特征的框图。该计算设备可访问计算机可读介质上的指令，并以适当顺序执行那些指令以执行证书签发系统的各个功能。

[0017] 图 2 示出证书签发系统可在其中操作的示例性网络化计算环境。签发器可驻留于例如设备 271 上。设备 277 上的客户机过程可从 271 上的签发器中请求证书，随后使用该证书与设备 275 上的服务器过程进行通信。

[0018] 图 3A 示出证书签发系统的转换组件部分。转换组件或转换引擎用来将多种格式的输入证书转换成单个通用格式。签发组件 34 产生的用于传送给客户机的通用格式证书可按需转换成任何格式。

[0019] 图 3B 示出输入证书请求如何由签发器处理的视图。首先，请求到达服务器入口点 301。例如 302 的原始格式的证书－或其它请求信息－被传送给转换层 303。它被转换成第二格式 304。然后该证书和／或关联于请求 304 的其它信息由服务器证书引擎 305 进行处理。

[0020] 图 3C 示出本发明的另一视图，其中客户机 310 向服务器 313 发送对证书的请求。类型 1 证书关联于客户机 310 的请求。该类型 1 证书可通过证书转换引擎 (CTE)311 转换成类型 2 证书。类型 2 证书可被签发引擎 312 用来确定是否要许可客户机 310 的请求。当签发引擎 312 以第二格式签发证书时，CTE 311 可将其转换成客户机 310 的第一格式。

[0021] 图 4 示出证书签发系统的签发组件部分。认证 41、授权 42、以及证书格式化 43 是签发组件可执行的示例性功能。作为执行这些功能的一部分，通用策略语言解析和实现引擎 44 可应用证书签发策略 45。因而在要实现的策略不需要在引擎 44 中表达而在由引擎 44 在运行时间消费的签发策略 45 中表达的意义上，签发组件是数据驱动的。

[0022] 图 5 示出一种系统和方法，用于扩展签发组件以应用和实现策略表达语言和 / 或策略语言解析和实现引擎 56 不会自然支持的新签发策略。当策略语言语法在例如 59 中得到扩展时，处理扩展后策略语言结构的插件逻辑 55 可被添加到实现引擎 56 中。当使用自然策略语言通配符时，处理扩展通配符替换模式的插件逻辑 57 可被添加到实现引擎 56 中。

具体实施方式

[0023] 某些特定细节在以下说明书和附图中陈述以提供对本发明各实施例的完整理解。然而，为了避免不必要地混淆本发明的各个实施例，通常关联于计算和软件技术的某些众所周知的细节并未在以下揭示中陈述。此外，本领域技术人员将理解，无需以下所述的一个或多个细节他们可实践本发明的其它实施例。最后，尽管各种方法是参照以下说明书中的步骤和顺序来描述的，但是这样的描述是用来提供本发明各实施例的清晰实现的，且各步骤和步骤的顺序不应视为是实践本发明所必须的。

[0024] 在此所述的设备和方法通常涉及签发数字证书。术语“证书”在此用作“数字证书”的缩写形式。如在背景技术中所述，证书是证明某物或某物所有权的真实性的文档。术语“证明”表示确认是正确、真实或真的。因而在此将被称为客户机的第一实体可使用证书来向第二实体 - 服务器 - 确认有关它自己的一些事实。证书通常（尽管并非必须）由受信任的第三方签发。当在此使用时，证书的范围可以从自己产生的文档或令牌到由受信任第三方签发的具有许多安全特性的可高度信任的数字文件，诸如根据一种或多种公钥或私钥技术等的加密。由证书证明的“某物”可以是任何东西。通常，可证明客户机的身份和 / 或对客户机花去或访问某些资源的授权，但也可证明任何其它东西。

[0025] 受信任的第三方在此被称为证书签发系统。术语“证书签发系统”在此且在本行业中也可称为“证书签发服务”，并且为简便起见可简称为术语“签发器”。证书签发系统确定是否向特定客户机授权一证书。如果是，则向客户机签发证书并可使用该证书向服务器进行认证。

[0026] 尽管客户机和服务器可被视为两个完整的计算设备，每一个都包括硬盘、总线、系统存储器等，但是本领域技术人员以更广泛含义来理解这些术语。客户机和服务器实际上可以是单个计算设备内、或在分布式计算装置中的多个计算机上存在的两个实体。这样，证书签发系统也可存在于包括一个或多个客户机以及一个或单个服务器实体的计算机内，并还可存在于多个设备之上。

[0027] 参照图 1，概述了适于结合证书签发系统使用的示例计算设备 100。在其最基本配置中，设备 100 通常包括处理单元 102 和存储器 103。取决于计算设备的准确配置和类型，存储器 103 可以是易失性存储器 103A（诸如 RAM）、非易失性存储器 103B（诸如 ROM、闪存等等）、或两者的某些组合。此外，设备 100 还可具有诸如磁性或光学的盘或带的大容量存储设备（可移动的 104 和 / 或不可移动的 105）。类似地，设备 100 可包括诸如键盘和鼠标的输入设备 107，和 / 或诸如将 GUI 表示为对计算设备 100 的功能的图形辅助访问的显示器的输出设备 106。设备 100 的其它方面可包括使用有线或无线介质与其它设备、计算机、网络、服务器等的通信连接 108。

[0028] 易失性存储器 103A、非易失性存储器 103B、可移动大容量存储 104 和不可移动大容量存储 105 是计算机可读介质的示例。计算机可读介质可包括通信介质以及计算机存储

介质。通信介质通常体现为计算机可读指令、数据结构、程序模块、或其它诸如载波或其它传送机制的已调制数据的信号的其它数据，并包括任何信息传送介质。术语“已调制数据信号”意指具有以这种把信息编码到信号中的方式来设置或改变的一个或多个特征的信号。作为示例，而非限制，通信介质包括诸如有线网络或直接有线连接的有线介质，以及诸如声学、RF、红外和其它无线介质的无线介质。

[0029] 计算机存储介质可以用来存储诸如计算机可读指令、数据结构、程序模块、或其它数据的信息的任何方法或技术实现。计算机存储介质包括，但不限于，RAM、ROM、EEPROM、闪存或其它存储器技术、CD-ROM、数字多功能光盘 (DVD) 或其它光学存储器、磁盒、磁带、磁盘存储器或其它磁性存储设备、或任何其它介质。

[0030] 本发明至少部分地可通过诸如由计算机 100 执行的程序模块的计算机可执行指令实现。通常，程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等。

[0031] 计算机可执行指令通常体现为某些形式的计算机可读介质上的对计算机 100 可用的数字信息。例如在图 1 中，系统存储器 103 可存储操作系统和应用程序以及其它程序模块和程序数据。应用程序等可绑定于操作系统，或可单独存在并利用操作系统服务来发挥作用。

[0032] 应当理解，尽管在此所述的本发明的各个实施例可以是软件实现，但在此所述的各种技术也可通过用硬件组件替换至少一些程序模块来实现。因而，尽管本发明的方法和装置、或其某些方面或部分可采取高级过程或面向对象编程语言的程序代码的形式，但（各）程序也可用汇编或及其语言实现。在任一情形中，语言可以是编译或解释语言，并可与硬件实现相结合。

[0033] 本发明可在许多通用或专用计算系统环境或配置中操作。众所周知的适用于本发明的计算系统、环境和 / 或配置的示例包括，但不限于，个人计算机、服务器计算机、手持式或膝上型设备、多处理器系统、基于微处理器的系统、机顶盒、可编程消费电器、网络 PC、小型计算机、包括任何以上系统或设备的分布式计算环境等。

[0034] 图 2 提供一示例性网络化计算环境。该网络包括计算设备 271、272、276 和 277，对象 273、274 和 275，以及数据库 278。这些实体 271、272、273、274、275、276、277 和 278 的每一个都可包括或使用程序、方法、数据存储、可编程逻辑等。实体 271、272、273、274、275、276、277 和 278 可跨越诸如 PDA、音频 / 视频设备、MP3 播放器、个人计算机等的相同或不同设备的各个部分。每个实体 271、272、273、274、275、276、277 和 278 可通过通信网络 270 与另一实体 271、272、273、274、275、276、277 和 278 进行通信。

[0035] 网络上的各个设备利用（各）协议层提供的功能来彼此通信。例如，超文本传输协议 (HTTP) 是结合万维网 (WWW) 或“web”使用的通用协议。通常诸如互联网协议 (IP) 地址的计算机网络地址、或诸如全球资源定位器 (URL) 的其它引用可用来彼此标识服务器或客户机计算机。网络地址可被称为 URL 地址。通信可在通信介质上提供，例如（各）客户机和（各）服务器可通过高容量通信的 TCP/IP 连接来彼此耦合。

[0036] 网络本身可包括向图 2 系统提供服务的其它计算实体，并可表示多个互连网络。每个实体 271、272、273、274、275、276、277 和 278 可包含离散的功能程序模块，这些程序模块可使用 API、或其它对象、软件、固件和 / 或硬件来请求其它实体 271、272、273、274、275、

276、277 和 278 的一个或多个的服务。

[0037] “客户机”是使用与之不相关的另一类或组的服务的一个类或组的成员。在计算中，客户机可以是请求由另一程序提供的服务的进程，即粗略地为一系列指令或任务。这种服务可以是例如由证书签发系统的证书签发。客户机进程无需“知道”任何有关其它程序或服务本身的工作细节就可使用所请求的服务。在客户机 / 服务器体系结构特别是网络化系统中，客户机通常是访问由例如服务器的另一计算机提供的共享网络资源的计算机。在图 2 示例中，取决于各情形，任何实体 271、272、273、274、275、276、277 和 278 可被视为客户机、服务器或两者。

[0038] 服务器通常（尽管并非必须）是在诸如因特网的远程或局域网上可访问的远程计算机系统。该客户机进程可在第一计算机系统内活动，而服务器进程可在第二计算机系统内活动，彼此经通信介质通信，从而提供分布式功能并允许多个客户机利用服务器的信息收集功能。任何软件对象都可在多个计算设备或对象上分布。

[0039] 因而本发明的各个实施例可解决请求证书的客户机实体驻留于网络中的例如第一计算设备 277 的情形。具有为客户机实体所需的一些资源的服务器实体可驻留于例如第二计算设备 275 中。证书签发系统可驻留于例如另外的第三计算设备 271。

[0040] 设备 277 上的客户机可确定它需要一证书来向设备 275 上的服务器证明一些客户机凭证。277 上的客户机因而在网络总线 170 上向 271 上的签发器提交请求。请求本身可包括一个或多个先前签发的证书。271 上的签发器继续确定客户机是否有权得到所请求的证书。271 上的签发器通过应用证书签发策略来完成它。如果 277 上的客户机具有策略所需的凭证，则所请求证书可由 271 上的签发器签发给 277 上的客户机。然后客户机可在它与 275 上服务器的通信中使用所签发证书以及任何数量的其它证书。

[0041] 图 3A 示出证书签发系统的转换组件部分。签发组件 34 更详细地在图 4 和 5 中示出。在本发明的各个实施例中，转换组件实际上可以是接收输入证书的第一个组件，也可以是在证书传送给客户机之前可操纵证书的最后一个组件。在其它实施例中，转换组件可按需用于证书签发系统中的任何地方。

[0042] 当对证书的请求到达证书签发系统时，诸如第一格式 30 的第一证书的任何附带证书可被路由到转换驱动器 31。转换驱动器 31 可操作以进行各证书与用于签发组件 34 的操作的通用格式之间的转换。因而，第一格式 30 的第一证书可被转换成第二格式。这种转换的结果是第二格式 33 的第一证书。

[0043] 相反，当证书由签发组件 34 签发时，它可从签发组件 34 产生的格式转换成客户机所需的任何格式。因而，第二格式 35 的第二证书可被转换成第三格式。这种转换的结果是第三格式 37 的第二证书。该第三格式可以是任何证书格式，包括达到对证书的客户机请求的一个或多个证书（例如 30）的格式。

[0044] 证书转换组件最好被设计成转换尽可能多的证书格式。这样，它可转换 X.509 证书、安全性断言标记语言 (SAML) 安全令牌证书、XrML 1.2 证书、和 MPEG-REL 证书（仅列出一些较流行的示例性证书格式）。然而，对每种可能的证书类型设计转换装置并非总是经济可行的。本发明并不限于转换组件能够转换的证书类型的数量或特定格式。

[0045] 因为在本领域中新的证书格式在持续产生，所以将转换组件设计成能进行扩展以容纳其它证书格式是有利的。驱动器 31 可基于诸如 32 的一个或多个类所提供的指令，来

管理将特定证书 30 的各个元素转换成通用格式证书 33。图 3A 的特定装置是有利的，因为它允许转换组件可扩展以容纳新的证书格式。然而，驱动器 31 和诸如图 3A 中 32、36 装置的类仅仅是示例性的，并且本领域技术人员将理解各种装置可用来完成证书从一种格式到另一种格式的转换。

[0046] 在图 3A 中，通用格式被称为第二格式。通用格式的选择涉及确定一种格式，该格式尽可能稳健以容纳各种格式的输入证书中可包含的全部各种类型信息。尽管可选择任何证书来用作通用格式，包括 X.509 证书、安全性断言标记语言 (SAML) 安全令牌证书、XrML 1.2 证书、和 MPEG-REL 证书（仅列出一些较流行的示例性证书格式），但已选择 MPEG-REL 来实现本发明，并被视为是有利于此目的的。MPEG-REL 目前被视为最丰富、最稳健和可扩展的证书格式。然而，随着技术的进步，可开发其它更多有利的证书。

[0047] 使用转换组件的另一优点源于这样的事实，各种证书格式的每一种都用它自己的方式来表达策略。现有证书签发器互操作性的障碍是格式不能兼容，因为要消费任何特定格式需要遍及签发器的定制算法。我们不能期望说服所有现有的证书制造商采用一通用格式，因此我们必须假设这些格式将继续长期存在。用于将不同的证书格式及其语义映射或转换成通用语言的技术减少了多种格式的系统性影响，并且是在解决证书互用性问题的方向上的一个步骤。

[0048] 正确的转换应满足语法和语义要求。前者需要转换后格式具有有效格式。后者需要转换后证书传送与原始证书相同的信息。然而，有源格式具有比目标格式更多信息的情形，这使得转换中的信息损失不可避免。因而实现本发明的目标是确保信息正确转换同时尽力保留其它信息。

[0049] 证书转换算法 32、26 可基于其功能分成两类：

[0050] 语法级：在结构之间映射

[0051] 语义级：在证书之间转换

[0052] 在一实施例中，该算法集合被实现为一个类集。该类集在图 3A 中被示为证书转换驱动器 31。转换驱动器 31 的各个实施例可包括三个组件：驱动器类、证书转换类和配置类。驱动器类可执行转换过程的整体协调并调用转换类。转换类可执行实际转换过程，而配置类可包含转换驱动器 31 进行操作所必须的配置数据。

[0053] 以下简单示例被包括为演示从第一证书格式 XrML 1.2 到第二证书格式 XrML2.0 的转换的示例性操作。如本领域技术人员可以理解的，MICROSOFT ® 的 RightsManagement Server（权限管理服务器）产品的第一版本使用 XrML 1.2 格式的证书来进行其策略评估。然而，权限管理服务器产品的将来发行版本将使用 XrML 2.0 格式的证书。因而，以下提供转换组件的操作和对在此提供的本发明的增长需要的较好示例。情形如下：客户机将证书请求发送给假定的实现本发明的权限管理服务器产品。该请求本身包含 XrML 1.2 格式的证书，例如图 3A 中的证书 30。XrML 2.0 是证书签发系统 34 所使用的通用格式。

[0054] 为了接收并理解两种格式的证书，实现一转换类集。假设这已经完成，在接收来自客户机的具有 XrML 1.2 格式的证书 30 的请求之后，转换驱动器 31 插件相应的证书转换类 32、将配置数据传送给它并调用转换方法。证书转换类 32 确认 XrML 1.2 证书 30 的签名和有效期，并及其转换成 XrML 2.0 证书 33，并将转换特定信息返回给转换驱动器 31。然后转换驱动器 31 使用 XrML 2.0 格式的证书 33 来调用权限管理服务器 34。在该示例中，权限

管理服务器 34 自然地理解 XrML 2.0 证书、执行证书签发操作、并将结果发送回转换驱动器 31。在接收由权限管理服务器 34 签发的 XrML 2.0 证书 35 之后，转换驱动器 31 创建将 XrML 2.0 证书 35 转换成 XrML 1.2 证书 37 的证书转换类 36。

[0055] 图 3B 示出输入证书请求如何由签发器处理的视图。首先请求到达服务器入口点 301。原始格式的证书－或其它请求信息 -302 被传送给称为转换层 303 的转换组件。注意，证书并非总是附于证书请求的。通常作出某些形式的证明，并且在证书是确认要证明的事实的较好方式时，它们并非总附于证书请求。各种类型的其它信息也可包括于请求中。当使用一些其它信息时，信息的特定数据类型可以有些像证书格式可不同的方式而不同。因而，证书转换层 303 可被配置成将诸如 302 的证书从一种格式转换成另一种格式，或从各种格式转换成单一格式，但也可配置成处理附于证书请求的任何数据。像输入证书的这种数据可被转换成可由单个签发组件或（在此称作）证书引擎 305 消费的通用格式。

[0056] 图 3B 中的输入信息可被转换成第二格式 304。如果输入信息已是第二格式，则当然无需转换。然后证书和 / 或关联于请求 304 的其它信息可由服务器证书引擎 305 来处理。

[0057] 图 3C 示出本发明的另一视图，其中客户机 310 向服务器 313 发送对证书的请求。类型 1 证书关联于客户机 310 的请求。类型 1 证书可被证书转换引擎 (CTE)311 转换成由签发器 312 消费的某些其它格式，诸如类型 2 证书。类型 2 证书可被签发引擎 312 用来确定是否要许可客户机 310 的请求。当签发引擎 312 签发第二格式的证书时，CTE 311 可将其转换成用于客户机 310 的第一格式。或者，签发引擎 312 可简便地产生用于客户机 310 的适当格式的证书，而无需再使用 CTE 311 来从第二格式证书转换成第一格式证书。注意在本发明的许多实施例中，CTE 311 可被配置成处理多个输入证书类型。每种类型最好被配置成将来自签发引擎 312 的第二格式证书转换成如客户机 310 所请求的各种证书类型的任一种。

[0058] 在各示例性情形中，可获益于诸如图 3C 所示系统的使用的是客户机设备和 / 或进程的证明，来许可客户机许可证颁发证书 (CLC) 请求、并特许对数字内容的权限。

[0059] 当使用以第一格式证书操作的客户机（“v1 客户机”）的用户首次尝试打开由权限管理系统保护的电子邮件时，产生一示例性认证情形。在该情形中，v1 客户机可向证书签发系统 313 发送表示客户机身份的目前是 XrML 1.2 证书的 v1 机器帐户证书 (MAC)，以及 WINDOWS ® 域凭证。CTE 311 可将 MAC 转换成目前是表示客户机身份的 MPEG-REL 证书的安全处理器证书 (SPC)，并将该 SPC 给予签发引擎 312。然后签发引擎可签发第二格式证书，例如目前是表示客户机上用户身份的 MPEG-REL 证书的权限帐户证书 (RAC)，并将该 RAC 送给 CTE 311。CTE 311 然后可将 RAC 转换成第一格式证书，例如目前是表示客户机上用户身份的 XrML1.2 证书的组身份证书 (GIC)。该签发系统 313 然后可用 GIC 向客户机 310 作出回应。

[0060] 一示例性 CLC 请求情形是，客户机 310 向签发系统 313 发送请求以获取授权用户公布对离线受保护内容的许可证的证书。在该情形中，CTE 311 可将关联于该请求的输入 RAC 转换成可由签发引擎 312 处理的 CLC。或者，客户机 310 可发出由 CTE 311 转换成 SPC 的 GIC，且签发器 312 可返回由 CTE 311 转换成 CLC 的 RAC。

[0061] 诸如图 3C 的系统可用于特许数字内容中权限的示例性情形如下：为了消费权限受保护内容，v1 客户机 310 可作出对签发系统 313 的特许请求。客户机 310 可向签发系统

313 发送 v1 发布许可证－签发许可证 (IL)，例如描述对特定保护内容的作者指定使用权限的 XrML 1.2 证书，和 GIC。然后 CTE 311 可转换输入的 IL 和 RAC，并将它们给予签发引擎 312。如果签发引擎 312 向客户机 310 授予使用许可证 (UL)，例如授权特定用户访问特定受保护内容的 MPEG-REL 证书，则 CTE311 也可将所签发的 UL 转换成终端用户许可证 (EUL)，诸如授权特定用户访问特定受保护内容的由 v1 客户机 310 消费的 XrML 1.2 证书。签发系统 313 然后可将该 UL 发回客户机 310。

[0062] 尽管众多软件设计可用来实现 CTE 311，但如本领域技术人员所理解的，示例性 CTE 设计可包括三个组件 :CTE 驱动器、证书转换类和配置类。

[0063] 在该设计中，CTE 驱动器与服务器入口点 301 和服务器证书引擎 305 交互。在接收证书请求之后，驱动器创建相应的证书转换类、将配置数据传送给转换类、并调用转换方法。然后证书转换类例如使用诸如 a) 签名确认，b) 证书有效期间隔过期时间以及 c) 对该签发器和受信任签发器集作比较的技术，来确认所处理证书的签名和有效期。证书转换类还可将证书转换成通用格式证书对应体，并向 CTE 驱动器返回通用格式证书字符串、以及其它转换－特定信息。

[0064] 然后 CTE 驱动器使用通用格式证书来调用证书引擎 305。在接收到由证书引擎 305 签发的合成通用格式证书之后，CTE 驱动器创建一证书转换类，该类将所产生的通用格式证书转换成第三格式的对应证书，例如原始输入证书的格式。

[0065] 因而，CTE 的示例性工作流可如下进行 :

[0066] 获得来自客户机的证书请求

[0067] 按需确认并解密所附证书

[0068] 将解密后的证书转换成通用格式证书

[0069] 可替换某些信息 (密钥、SPC 等)

[0070] 保存不能在通用格式证书中表示的任何信息

[0071] 调用证书签发引擎

[0072] 获得来自证书签发引擎的已签发通用格式证书

[0073] 按需解密已签发的通用格式证书

[0074] 将解密后的已签发通用格式证书转换成客户机格式证书

[0075] 可使用在前一步骤中保存的信息

[0076] 加密并签名客户机格式证书

[0077] 将客户机格式化的证书发送给客户机

[0078] 图 4 示出证书签发系统的签发组件部分。一旦参照图 3A 所述转换完成，转换后的证书表示可使用图 4 所示的各种功能组件来处理。转换组件未在图 4 或图 5 中示出以避免使示图模糊。对于各个组件的示例性组合，参见图 3A。

[0079] 客户机 40 发送一个或多个声明、及其对证书的请求。声明是实体所作的任何断定，以用来确定该实体是否有权得到证书。如果声明经校验为真，则客户机实体 40 显示它具有凭证。在向客户机 40 签发证书之前证书签发策略最终会需要一个或多个凭证。这些凭证本身可由一个或多个证书证明。

[0080] 认证 41、授权 42 和凭证格式化 43 是可由签发组件执行的示例性功能。如图 4 所示，这些功能可按认证 41、授权 42 和凭证格式化 43 的顺序连续地执行。该顺序在所有实施

例中并非是必需的。取决于满足客户机请求的所需，41、42 和 43 可独立执行，可执行某些子组合，或者一个或多个这些功能可结合图 4 中所示的有些其它功能来执行。

[0081] 当连续执行 41、42 和 43 时，认证过程 41 首先可确定声明支持证书请求的客户机 40 实际上是否是客户机 40 所声明的实体。如果这得到了证实，则授权过程 42 可确定客户机是否可获得接收所请求证书的授权。或者，授权过程 42 可简便地确定客户机 40 的授权级别，并将其记录在所创建的证书中。最后，当证书代表客户机 40 产生时，列示于证书中的客户机 40 凭证可由 43 在所产生的证书中格式化。以下提供了可由证书签发引擎 44 应用的示例性算法：

[0082] 签发证书（用户输入证书，服务器签发策略）

[0083] 基于签发策略

[0084] 认证用户输入证书

[0085] 授权用于签发证书的服务器

[0086] 如下构建结果证书

[0087] 创建客户机引擎需要认证的

[0088] 创建签发策略所允许的许可

[0089] 授权许可签发

[0090] 调用扩展以按需授权

[0091] 产生许可

[0092] 调用扩展以按需产生许可的一部分

[0093] 构建作为许可的用户权限

[0094] 签名所产生的证书

[0095] 返回该证书

[0096] 作为执行认证 41、授权 42 和凭证格式化 43 功能的一部分，通用策略语言解析和实现引擎 44 可应用证书签发策略 45。签发组件在要实现的策略不在引擎 44 中表达而在由引擎 44 在运行时间消费的签发策略 45 中表达的意义上，签发组件是数据驱动的。

[0097] 尽管现有技术证书签发系统在产生证书时应用并实现一策略，但该策略在现有技术签发器中可用证书签发系统二进制码表达为经过编译的算法，或者表达为配置参数的经具体建模的“脆弱”集合。结果，改变该实现策略需要重新编码、重新编译、并重新使用新的证书签发系统。换言之，所传送的签发器受限于实现证书签发系统编程人员所预想的策略集。

[0098] 证书签发引擎 44 应几乎不或不包括预想策略结构。相反，引擎 44 应包含元数据驱动策略实施引擎，该引擎兑现它在运行时间遭遇的来自 45 的特定策略数据。该策略数据 45 使用通用的为引擎 44 使用而设计的可扩展策略表达语言。

[0099] 引擎 44 最好用单一的通用策略表达语言来操作，并基于 45 中的可用策略和数据作授权判定。通过在同类策略表达语言格式上执行该处理，引擎 44 的逻辑更简单更为有效，并可优化成选定的策略表达语言。通过数据驱动，引擎 44 可估算所表达策略的广泛范围，而不必改变引擎 44 的逻辑以容纳新的策略、语义或结构。如附图所示的引擎 44 包括用于解析和实现策略 45 的功能组件，以及用于产生证书的功能组件。引擎 44 所产生的证书的形式可由策略 45 以及签发策略的其它方面来管理。

[0100] 用于表达策略 45 的策略表达语言可取各种各样形式的任一种。所使用语言可以是诸如可扩展标记语言 (XML)、超文本标记语言 (HTML)、或某些其它标记语言的标记语言。如本领域技术人员所理解的，可读字和标记的集可在这些语言中组合以精确地指定所需操作。诸如引擎 44 的机器进程可被配置成在运行时间消费该形式的文件并实现所需操作。设计用于本发明的任何策略表达语言应当是稳健的、可扩展的和灵活的，以按需容纳策略中的变化和语言语义的添加。标记指字符或其它符号的序列，它们可插入文本或字处理文件中的某些地方以描述文档的逻辑结构。标记指示符常常称为“标签”。标记可通过键入符号或通过使用编辑器并选择预先打包的标记符号（来保存键击）而直接由文档创建者插入。

[0101] XML 是“可扩展”的，因为不像 HTML，标记符号是非限制和自定义的。XML 实际上是标准通用标记语言 (SGML) 的较简单和便于使用的子集，SGML 是如何创建文档结构的标准。可预期 HTML 和 XML 将在许多 Web 应用程序中一起使用。例如，XML 标记可在 HTML 页面内显现。这样，用于本发明的特定语法可包括标记语言的组合。

[0102] 在图 4 中，引擎 44 应用以策略表达语言表达的并以数字格式存储的策略 45。策略 45 可显现为一个或多个数字文件、或数据库、或任何其它存储的数据格式。本领域技术人员认为数字文件可转换成任何形式：其各方面可被插入数据库的各个字段，或者文件可从一格式转换成另一格式。因而，尽管期望至少在开始时策略最好由人用文本编辑器可使用的数字格式，诸如通用文本 (.txt) 或文档 (.doc) 格式的文件来创建，但这种初始数字文件可在存储到 45 中之前转换成任意数量的形式。不管数据的格式如何，如果客户机有权得到所请求的证书，则在此表达的签发策略 45 必须由客户机 40 满足。策略 45 还可管理所产生证书的格式，即它可包括用于证书格式化的策略。

[0103] 签发策略 45 最好由至少以下组成：

[0104] 客户机认证要求

[0105] 客户机授权要求

[0106] 证书签发服务的授权要求

[0107] 授权实现指令

[0108] 为了再次将流行的 MICROSOFT WINDOWS ® Rights Management Server 签发器用作证书签发系统的示例，本领域技术人员认为该签发器可用来实现用于受保护文档和电子邮件的“信息权限管理”特征，例如 MICROSOFT ® Office 2003 的信息权限管理特征。作为方案的一部分，WINDOWS ® Rights Management Server 的现有版本使用包括预想的硬编码的脆弱签发策略定义的签发器。只可实现固定的众所周知的签发策略集 - 例如：

[0109] 什么是受信任的应用程序？

[0110] 什么用户是具体排除的？

[0111] 什么实体受信任来签发用户标识证书？

[0112] 权限管理软件的什么版本必须由用户在其桌面上运行？

[0113] 相反，WINDOWS ® Rights Management Server 的现有版本不能实现新的签发策略，诸如：

[0114] 什么是受用户公司部门信任的应用程序？

[0115] 哪类用户是具体排除的（例如其网络密码将在少于 7 天内过期的所有人）？

[0116] 什么特定证书是证书签发系统信任要产生的?

[0117] 通过重新结构化WINDOWS® Rights Management证书签发服务的现有版本来实现在此所述的系统和方法,产品可用灵活的策略表达语言来理解和实现策略,且无需更改所使用的签发器就可容纳以上列示的新签发策略以及任何其它可能签发策略。只有45中的表达策略需要进行改变。

[0118] 证书签发系统可发布其签发策略45以及可用签发证书格式集,以便于客户机40发现过程、辨析等。以下是为进行说明的一示例性签发策略:<r:license licenseId="f34e026a-836c-4557-97db-4368b3eddd14"xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS">

[0119] <r:title>RightsAccountCertificateIssuancePolicyRoot-Public</r:title>

[0120] <r:inventory>

[0121] <r:forAll licensePartId="LP0"varName="ValidityInterval">

[0122] <r:anXmlExpression>/r:validityInterval</r:anXmlExpression>

[0123] </r:forAll>

[0124] <r:forAll licensePartId="LP1"varName="AccountEncryptionPublicKey"/>

[0125] <r:keyHolder licensePartId="LP2">

[0126] <r:info>

[0127] <KeyNamexmlns="http://www.w3.org/2000/09/xmldsig#">RACIssuancePolicyRoot</KeyName>

[0128] <KeyValue xmlns="http://www.w3.org/2000/09/xmldsig#">

[0129] <RSAKeyValue><Modulus>rrREhbeyebCOsKWeVh7KSc6oFJj6zZX8vJQQDKWxpDjwm7EvbSSgwt/3/ZVN5QJa8vclZ061gkp5hRGCslVvJzSVs+du0caz519uQXTCXft0tVuQkv7LCktbT5aKOpUuoDs26Hs/Vw4Cg4IJwbMAmyuAZ27o6ngd1LlVm7o/rr0=</Modulus>

[0130] <Exponent>AQAB</Exponent>

[0131] </RSAKeyValue>

[0132] </KeyValue>

[0133] </r:info>

[0134] </r:keyHolder>

[0135] <r:forAll licensePartId="LP3"varName="PrivateKey">

[0136] <r:anXmlExpression>/tm:privateKey/xkms:RSAKeyValue</r:anXmlExpression>

[0137] </r:forAll>

[0138] <r:forAll licensePartId="LP4"varName="Licensor">

[0139] <r:propertyPossessor>

[0140] <trustedLicensor xmlns="tm"/>

[0141] <r:trustedRootIssuers>

[0142] <r:keyHolder licensePartIdRef="LP2"/>

[0143] </r:trustedRootIssuers>

[0144] </r:propertyPossessor>

[0145] </r:forAll>

[0146] <r:forAll licensePartId="LP5"varName="SidPrincipal">

[0147] <r:anXmlExpression>/tm:sidPrincipal</r:anXmlExpression>

[0148] </r:forAll>

[0149] </r:inventory>

[0150] <r:grant>

[0151] <r:forAll licensePartIdRef="LP4"/>

[0152] <r:forAll licensePartIdRef="LP1"/>

[0153] <r:forAll licensepartIdRef="LP0"/>

[0154] <r:forAll varName="AccountInfo">

[0155] <r:anXmlExpression>/tm:account/tm:identity[@type="urn:msft:tm:identity:rfc822" andcontains("microsoft.com")]</r:anXmlExpression>

[0156] </r:forAll>

[0157] <r:principal varRef="Licensor"/>

[0158] <r:issue/>

[0159] <r:grant>

[0160] <r:keyHolder varRef="AccountEncryptionPublicKey"/>

[0161] <r:possessProperty/>

[0162] <account varRef="AccountInfo"xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0163] <r:validityInterval varRef="ValidityInterval"/>

[0164] </r:grant>

[0165] <r:validityInterval>

[0166] <r:notBefore>2004-05-20T09:48:49Z</r:notBefore>

[0167] <r:notAfter>2004-05-20T09:48:49Z</r:notAfter>

[0168] </r:validityInterval> </r:grant> <r:grant>

[0169] <r:forAll licensePartIdRef="LP4"/>

[0170] <r:forAll licensePartIdRef="LP1"/>

[0171] <r:forAll licensePartIdRef="LP0"/>

[0172] <r:forAll varName="KeyUsageInfo">

[0173] <r:anXmlExpression>/tm:keyUsage[@uri="urn:msft:tm:keyUsage:encryption" or@uri="urn:msft:tm:keyUsage:signing"]</r:anXmlExpression>

[0174] </r:forAll>

[0175] <r:principal varRef="Licensor"/>

[0176] <r:issue/>

[0177] <r:grant>

[0178] <r:keyHolder varRef="AccountEncryptionPublicKey"/>

[0179] <r:possessProperty/>

[0180] <keyUsage varRef="KeyUsageInfo"xmlns="http://www.microsoft.

com/DRM/XrML2/TM/v2"/>

[0181] <r:validityInterval varRef="ValidityInterval"/>

[0182] </r:grant> </r:grant> <r:grant>

[0183] <r:forAll licensePartIdRef="LP4"/>

[0184] <r:forAll licensePartIdRef="LP1"/>

[0185] <r:forAll licensePartIdRef="LP0"/>

[0186] <r:forAll varName="BindingPrincipalInfo">

[0187] <r:anXmlExpression>/tm:bindingPrincipals/r:allPrincipals/tm:sidPrincipal</r:anXmlExpression>

[0188] </r:forAll>

[0189] <r:principal varRef="Licensor"/>

[0190] <r:issue/>

[0191] <r:grant>

[0192] <r:keyHolder varRef="AccountEncryptionPublicKey"/>

[0193] <r:possessProperty/>

[0194] <bindingPrincipals varRef="BindingPrincipalInfo"xmlns=" http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0195] <r:validityInterval varRef="ValidityInterval"/>

[0196] </r:grant> </r:grant> <r:grant>

[0197] <r:forAll licensePartIdRef="LP4"/>

[0198] <r:forAll licensePartIdRef="LP1"/>

[0199] <r:forAll licensePartIdRef="LP0"/>

[0200] <r:forAll varName="CertificationPrincipalInfo"><r:anXmlExpression>/tm:ceerificationPrincipals/r:allPrincipals/tm:sidPrincipal</r:anXmlExpression>

[0201] </r:forAll>

[0202] <r:principal varRef="Licensor"/>

[0203] <r:issue/>

[0204] <r:grant>

[0205] <r:keyHolder varRef="AccountEncryptionPublicKey"/>

[0206] <r:possessProperty/>

[0207] <certificationPrincipals varRef="CertificationPrincipalInfo" xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0208] <r:validityInterval varRef="ValidityInterval"/>

[0209] </r:grant>

[0210] </r:grant>

[0211] <r:grant>

[0212] <r:forAll licensePartIdRef="LP4"/>

[0213] <r:forAll licensePartIdRef="LP1"/>

```
[0214]      <r:forAll licensePartIdRef="LP0"/>
[0215]      <r:forAll varName="TrustedSecurityProcessor">
[0216]           <r:propertyPossessor>
[0217]                <trustedSecurityProcessor xmlns="tm"/>
[0218]                <r:trustedRootIssuers>
[0219]                     <r:keyHolder licensePartIdRef="LP2"/>
[0220]                </r:trustedRootIssuers>
[0221]           </r:propertyPossessor>
[0222]      </r:forAll>
[0223]      <r:principal varRef="Licensor"/>
[0224]      <r:issue/>
[0225]      <r:grant>
[0226]           <r:keyHolder varRef="TrustedSecurityProcessor"/>
[0227]           <r:possessProperty/>
[0228]           <holdsPrivateKey xmlns="http://www.microsoft.com/DRM/XrML2/
TM/v2">
[0229]                <r:keyHolder varRef="AccountEncryptionPublicKey"/>
[0230]           </holdsPrivateKey>
[0231]           <r:validityInterval varRef="ValidityInterval"/>
[0232]       </r:grant>
[0233]  </r:grant>
[0234]  <r:grant>
[0235]       <r:forAll licensePartIdRef="LP4"/>
[0236]       <r:forAll licensePartIdRef="LP1"/>
[0237]       <r:forAll licensePartIdRef="LP0"/>
[0238]       <r:principal varRef="Licensor"/>
[0239]       <r:issue/>
[0240]       <r:grant>
[0241]            <r:forAll varName="TrustedSecurityProcessor">
[0242]                 <r:propertyPossessor>
[0243]                      <trustedSecurityProcessor xmlns="tm"/>
[0244]                      <r:trustedRootIssuers>
[0245]                           <r:keyHolder licensePartIdRef="LP2"/>
[0246]                      </r:trustedRootIssuers>
[0247]                 </r:propertyPossessor>
[0248]            </r:forAll>
[0249]            <r:principal varRef="TrustedSecurityProcessor"/>
[0250]            <receivePrivateKey xmlns="tm"/>
[0251]            <r:keyHolder varRef="AccountEncryptionPublicKey"/>
```

[0252] <r:validityInterval varRef="ValidityInterval"/>

[0253] </r:grant>

[0254] </r:grant>

[0255] <r:grant>

[0256] <r:forAll licensePartIdRef="LP4"/>

[0257] <r:forAll licensePartIdRef="LP3"/>

[0258] <r:forAll licensePartIdRef="LP5"/>

[0259] <r:forAll licensePartIdRef="LP0"/>

[0260] <r:principal varRef="Licensor"/>

[0261] <r:issue/>

[0262] <r:grant>

[0263] <r:allPrincipals>

[0264] <sidPrincipal varRef="SidPrincipal"xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0265] </r:allPrincipals>

[0266] <decryptWithPrivateKey xmlns="tm"/>

[0267] <privateKey varRef="PrivateKey"xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0268] <r:validityInterval varRef="ValidityInterval"/>

[0269] </r:grant>

[0270] </r:grant>

[0271] <r:grant>

[0272] <r:forAll licensePartIdRef="LP4"/>

[0273] <r:forAll licensePartIdRef="LP3"/>

[0274] <r:forAll licensePartIdRef="LP5"/>

[0275] <r:forAll licensePartIdRef="LP0"/>

[0276] <r:principal varRef="Licensor"/>

[0277] <r:issue/>

[0278] <r:grant>

[0279] <r:allPrincipals>

[0280] <sidPrincipal varRef="SidPrincipal"xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0281] </r:allPrincipals>

[0282] <signWithPrivateKey xmlns="tm"/>

[0283] <privateKey varRef="PrivateKey"xmlns="http://www.microsoft.com/DRM/XrML2/TM/v2"/>

[0284] <r:validityInterval varRef="ValidityInterval"/>

[0285] </r:grant>

[0286] </r:grant>

[0287] <r:issuer xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS">

[0288] <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">

[0289] <SignedInfo>

[0290] <CanonicalizationMethod Algorithm="http://www.microsoft.com/xrml/lwc14n"/>

[0291] <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>

[0292] <Reference>

[0293] <Transforms>

[0294] <Transform Algorithm="urn:mpeg:mpeg21:2003:01-REL-R-NS:licenseTransform"/>

[0295] <Transform Algorithm="http://www.microsoft.com/xrml/lwc14n"/>

[0296] </Transforms>

[0297] <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>

[0298] <DigestValue>ZYG/bsCMxs5FhAT/atoXYGRuQ6Y=</DigestValue>

[0299] </Reference>

[0300] </SignedInfo><SignatureValue>Jd72Kt3h1fTYigxaYlrjaES+RLzmMZjr7bJBb9236GD7tty90zmZQxpYqTrA9D/qmrca5k84BGzefXodP8uLokDxUkpdXEI4aVurCDjP7cHwtOnZdMR5ATMvSgPn4kLHZ6E0g2pX7gjAm8jItvmD49Sa2D9CKjOtORq5zEkQMLc=</SignatureValue>

[0301] <KeyInfo>

[0302] <KeyValue>

[0303] <RSAKeyValue><Modulus>rrREhbeyebCOsKWeVh7KSc6oFJj6zZX8vJQQDKWxpDjwm7EvbSSgwt/3/ZVN5QJa8vc1Z061gkp5hRGCs1VvJzSVs+du0caz519uQXTCXft0tVuQkv7LCktbT5aKOpUuoDs26Hs/Vw4Cg4IJwbMAmyuAZ27o6ngd1Ll Vm7o/rr0=</Modulus>

[0304] <Exponent>AQAB</Exponent>

[0305] </RSAKeyValue>

[0306] </KeyValue>

[0307] </KeyInfo>

[0308] </Signature>

[0309] <r:details>

[0310] <r:timeOflssue>2004-05-20T09:48:49Z</r:timeOflssue>

[0311] </r:details>

[0312] </r:issuer>

[0313] <r:otherInfo>

[0314] <tm:infoTables xmlns:tm="http://www.microsoft.com/DRM/XrML2/TM/v2">

[0315] <tm:infoList tag="#LicenseType">

[0316] <tm:infoStr name="licenseRole">RightsAccountCertificateIssuancePolicy</tm:infoStr>

[0317] <tm:infoStr name="licenseVersion">1.0</tm:infoStr>

[0318] <tm:infoStrname="licenseType">RightsAccountCertificateIssuancePolicy-Public</tm:infoStr>

[0319] <tm:infoStr name="licensorUrl">http://rms.microsoft.com/rms/certification</tm:infoStr>

[0320] </tm:infoList>

[0321] </tm:info Tables>

[0322] </r:otherInfo></r:license>

[0323] 图 5 示出一种系统和方法，用于扩展签发组件以应用和实现策略表达语言和 / 或策略语言解析和实现引擎 56 不会自然支持的新签发策略。当策略语言语法通过组合附加数据 59 和签发策略数据 58 而得到扩展时，处理扩展后策略表达语言结构的语义的插件逻辑 55 可被添加到实现引擎 56 中。当例如通过将通配符替代模式数据 60 添加到签发策略数据 58 来使用自然策略语言通配符时，处理扩展通配符替换模式 60 的插件逻辑 57 可被添加到实现引擎 56 中。

[0324] 数据驱动的策略评估引擎 56 可被设计成预测它不理解或不打算处理的证书的语义。这些证书语义的可扩展性机制可被呈现，从而可询问用于该证书的诸如 55 或 57 的定制逻辑，以用良好定义的方式来提供值、执行定制处理、或以其它方式处理未知语义。该处理的结果可反馈到策略估算引擎 56 中并可能影响确定对证书权利的最终结果。

[0325] 估计至少有两种机制可用于扩展签发组件。首先通过在上面建立可扩展策略表达语言（例如发挥 XML 可扩展性的语言），并提供适当的诸如 55 的插件机制，证书签发系统可支持对其客户的定制可扩展性。其次，通过将通配符替换模式的概念包括在其策略表达语言中并提供诸如 57 的适当插件机制，签发器可支持对其客户的定制可扩展性。

[0326] 从用于扩展签发组件的第一所述选项开始，策略表达语言语法在较佳实施例中是可扩展的，策略实现引擎 56 可预先配置成“知道”如何兑现策略表达语言的原始方面的语义含义。然而，如果策略表达语言的语义得到扩展，则必须也扩展引擎 56。

[0327] 策略表达语言的扩展可在引擎 56 可访问的数字数据 59 中陈述。这种扩展可表现为一个或多个文件、或数据库、或任何其它已存储的数据格式。尽管扩展最好由人用文本编辑器可使用的格式，诸如通用文本 (.txt) 或文档 (.doc) 格式的文件来创建，但这种初始文件可在存储之前转换成任意数量的形式。不管数据的格式如何，在此表达的签发策略扩展 59 可由定制逻辑 55 访问，以使用扩展后的策略表达语法 59 来应用并实现所表达策略。

[0328] 为了完成它，引擎 56 可被配置成允许诸如 55 的扩展后逻辑（“插件”）进行登记。插件 55 可提供对策略表达语言的任何新语法扩展 59 的语义支持。因而非预期的客户要求无需检查引擎 56 本身或策略表达语言的原始使用语法就可得到解决。策略表达语言被理想地设计成语法支持扩展。

[0329] 一示例可有助于阐明使用插件 55 的引擎的可扩展性。假设证书签发系统与包含表示用户 PKI 身份的元素的 XML 策略表达语言一起发送……可能它看起来如下所示：

[0330] <user>

[0331] <name>George Washington</name>

[0332] <publickey>1234567890</publickey>

[0333] </user>

[0334] 证书签发系统可能想要扩展用户的概念，以包括用户的企业低权重目录访问协议 (LDAP) 身份。包括 LDAP 语法的策略语言扩展可在需要时用对证书签发系统数据库执行 LDAP 查询的代码来支持。在该情形中，扩展后的策略语言结构看起来如下所示：

[0335] <user>

[0336] <name>George Washington</name>

[0337] <publickey>1234567890</publickey>

[0338] <extension:ldapid>georgewashington</extension:ldapid>

[0339] </user>

[0340] 此外，称为插件 55 的用 LDAP 查询校验用户的代码将进行编码，并向证书签发系统进行登记。该插件 55 将在遇到扩展后策略表达语言语法时由引擎 56 调用。注意，在某些实施例中，使用扩展后语法而不登记插件是可能的。这可通过将系统的扩展后语法添加到输入 / 输出证书以及由引擎揭示的签发策略中来完成。

[0341] 现在参看用于扩展签发组件的第二部分，所使用的策略表达语言可包括通配符替代参数。通配符替代参数可在原始签发策略 58 中陈述，或者通过使附加数据 60 可用于补充原始策略 58 来添加到策略中。

[0342] 通配符替换模式 60 可表现为一个或多个文件、或数据库、或任何其它已存储的数据格式。尽管初始替换模式最好由人用文本编辑器可使用的格式，诸如通用文本 (.txt) 或文档 (.doc) 格式的文件来创建，但这种初始文件可在存储到 60 中之前转换成任意数量的形式。不管数据的格式如何，在此表达的通配符替换模式 60 可由定制逻辑 57 访问，以使用定制逻辑 57 指示的方式来应用并实现通配符。

[0343] 如果策略表达语言包括其语法内的通配符定义 60，且引擎 56 提供一机制来登记选择特定所需值的定制逻辑 57，则这向证书签发系统提供可扩展性的另一途径。

[0344] 再一次，一示例可有助于阐明。示例性证书签发系统可包含定义已签发格式的策略。例如，服务可包含表述服务可向客户机签发“受信任雇员证书”的证书签发策略。因为签发者必须对客户机的动态世界作出响应，通配符证书签发策略可被结构化为：

[0345] 证书签发系统可向它认为适当的任何客户机签发“受信任雇员证书”。

[0346] 然后证书签发系统所有人可定义并登记例如 57 的逻辑，它填入客户机“认为适当”的说明书中。逻辑 57 可确定在特定服务调用期间应向哪些客户机签发“受信任雇员证书”。该理解 57 可在遇到证书签发策略中更一般的通配符定义从句时由证书签发系统来调用。

[0347] 按照可根据图 1 和 2 中提供的一般框架建立的不同计算环境，在此提供的系统和方法不能解释为以任何方式受限于特定计算框架。相反，本发明不应解释为限于任何单个实施例，而相反应根据所附权利要求在宽度和范围内进行解释。

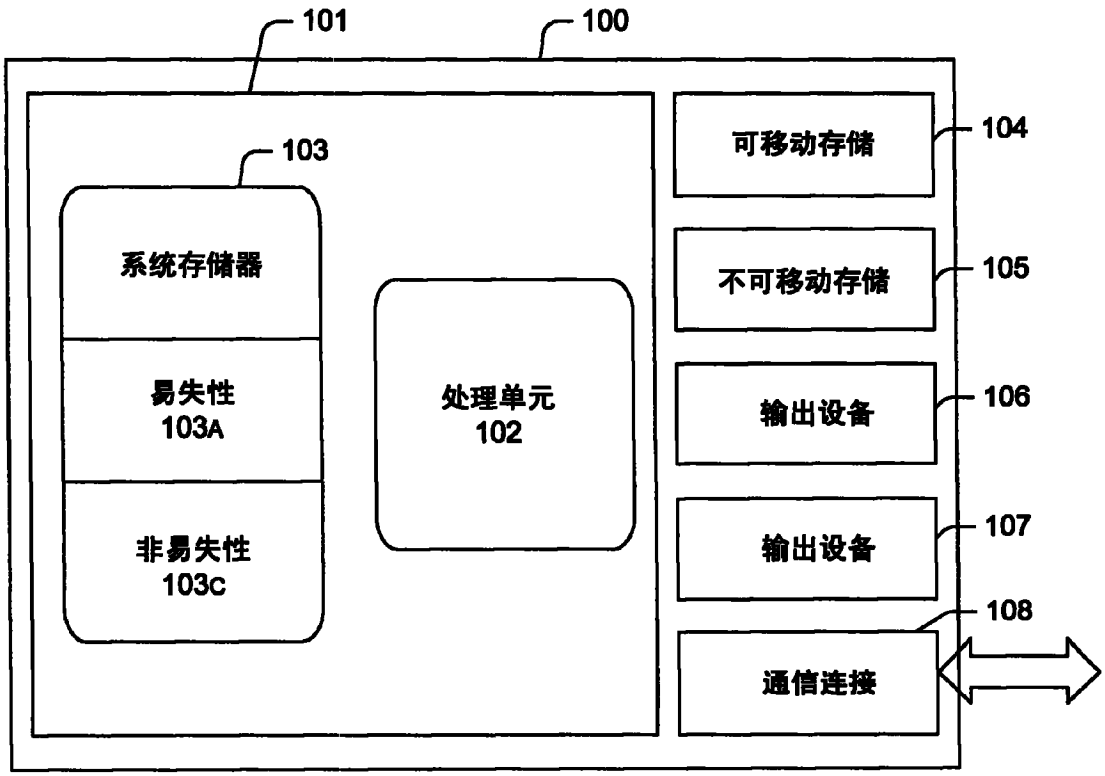

101
100
103
系统存储器
易失性
103A
非易失性
103C
处理单元
102
可移动存储
104
不可移动存储
105
输出设备
106
输出设备
107
108
通信连接

图 1

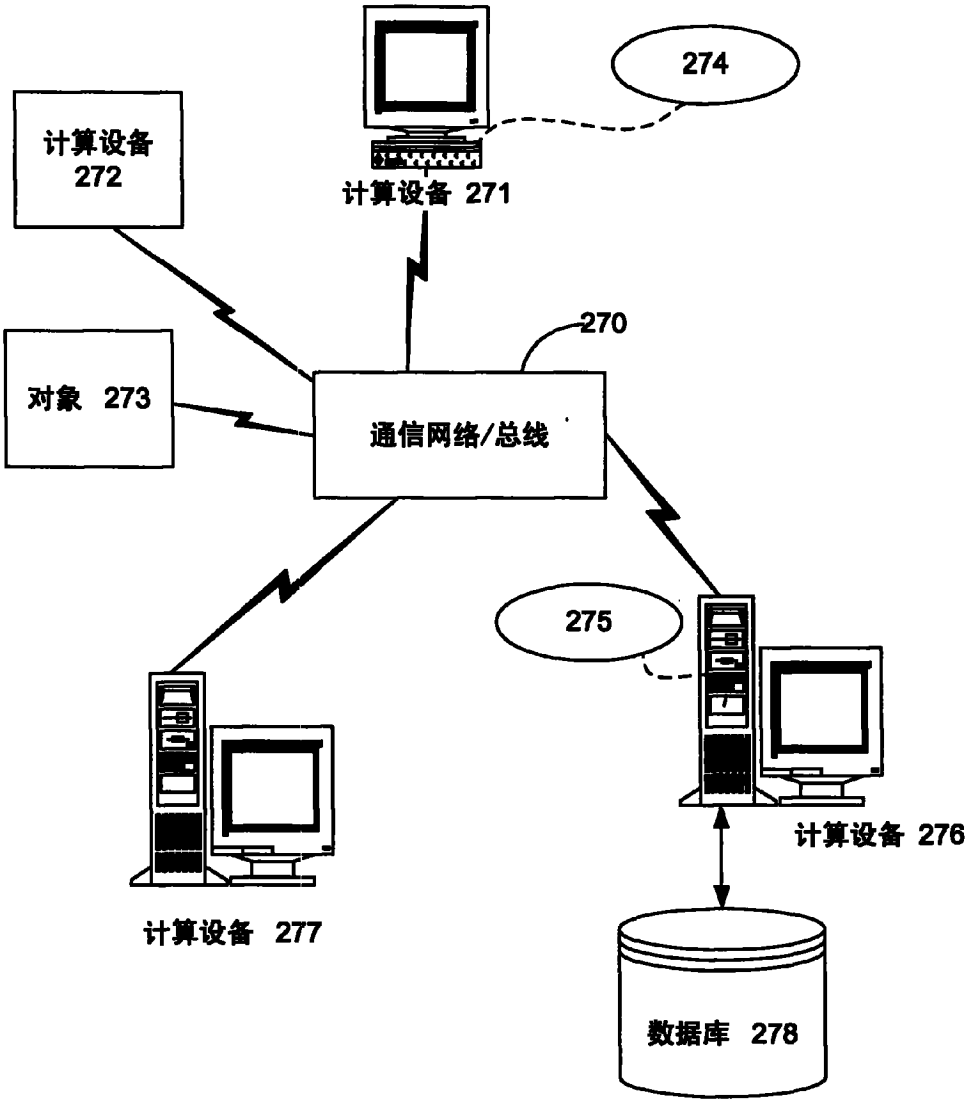

274
计算设备
272
计算设备 271
270
对象 273
通信网络/总线
275
计算设备 276
计算设备 277
数据库 278

图 2

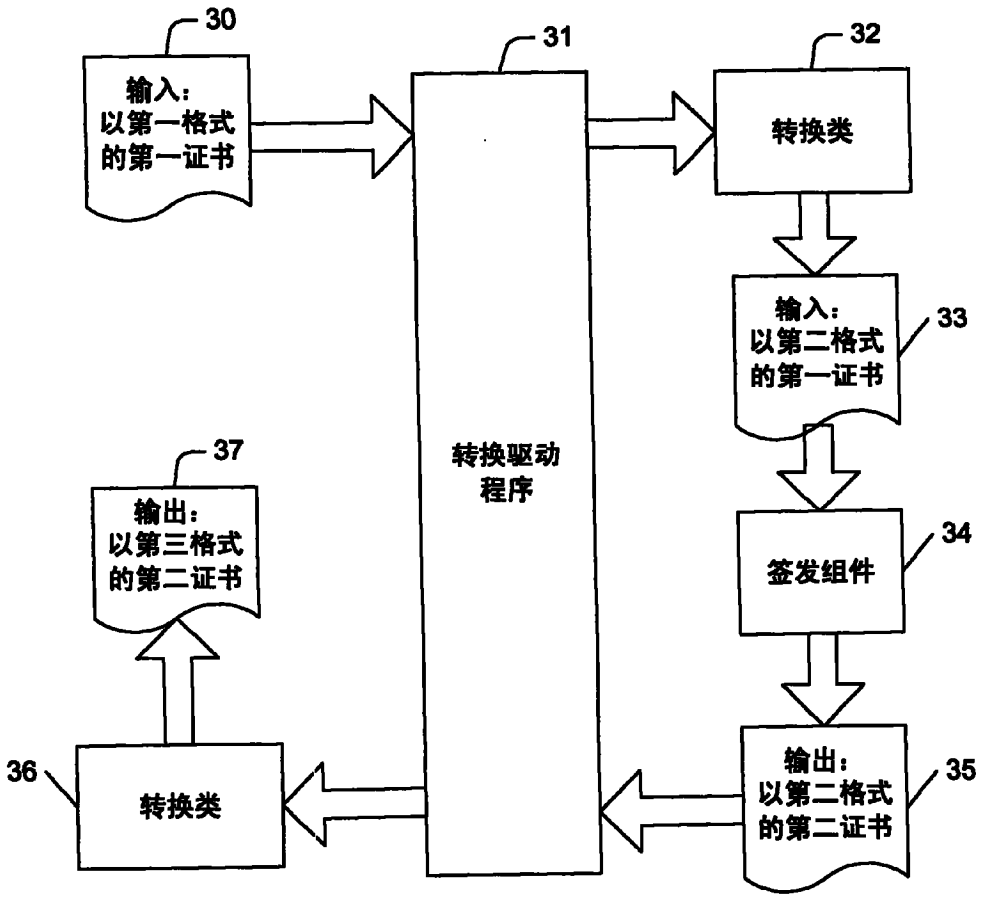

30
输入：
以第一格式
的第一证书
31
转换驱动
程序
32
转换类
输入：
以第二格式
的第一证书
33
签发组件
34
输出：
以第二格式
的第二证书
35
转换类
36
输出：
以第三格式
的第二证书
37

图 3A

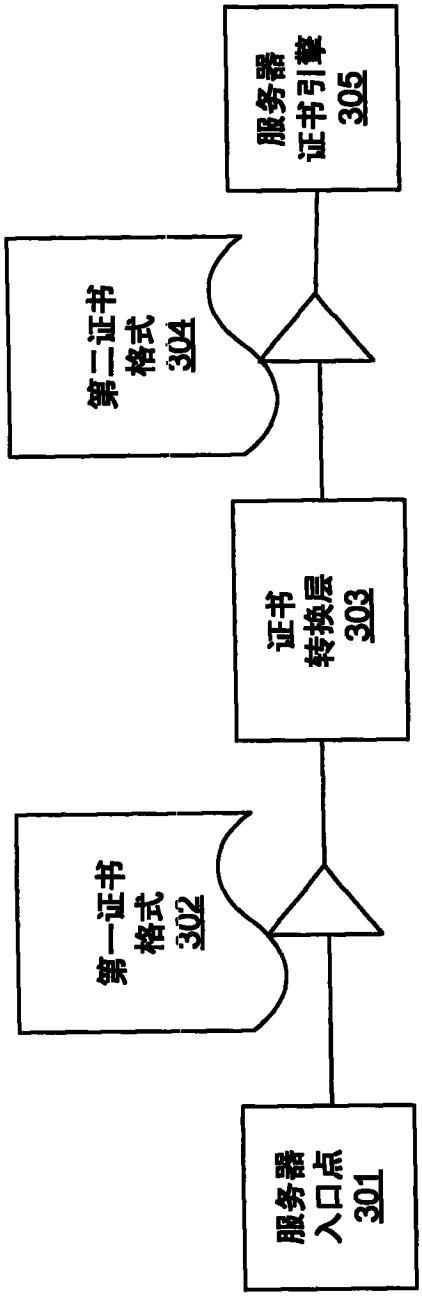


图 3B

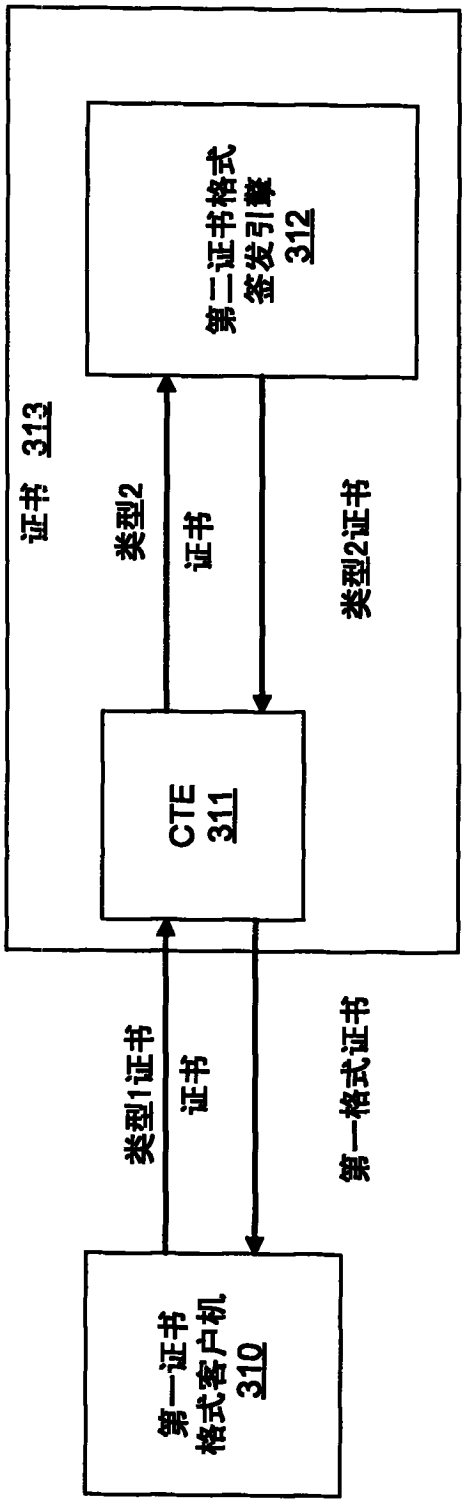

第二证书格式
签发引擎
312
证书 313
类型2
证书
类型2证书
CTE
311
类型1证书
证书
第一格式证书
第一证书
格式客户机
310

图 3C

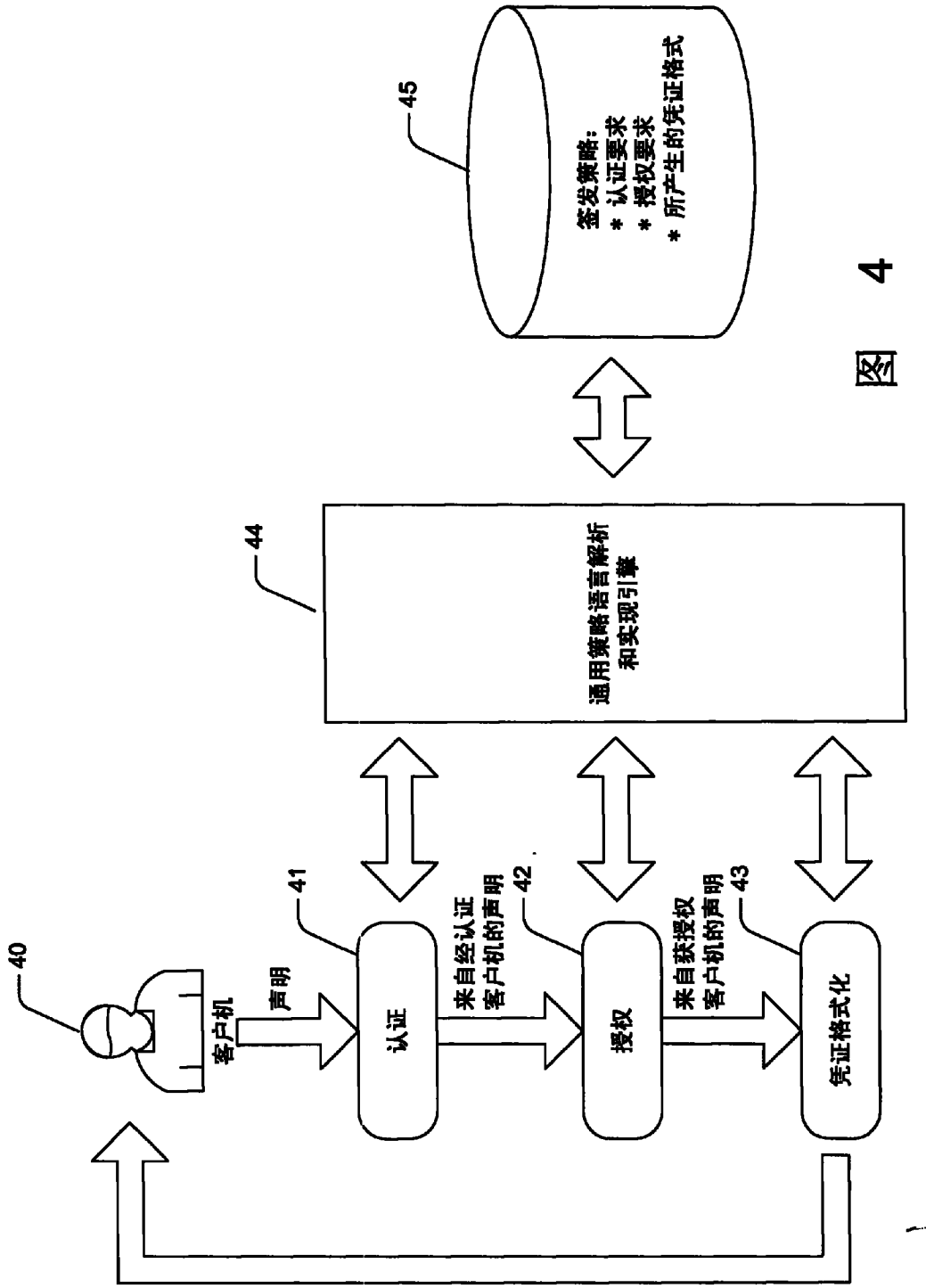

45
签发策略：
* 认证要求
* 授权要求
* 所产生的凭证格式
44
通用策略语言解析和实现引擎
40
客户机
声明
41
认证
来自经认证客户机的声明
42
授权
来自获授权客户机的声明
43
凭证格式化

图 4

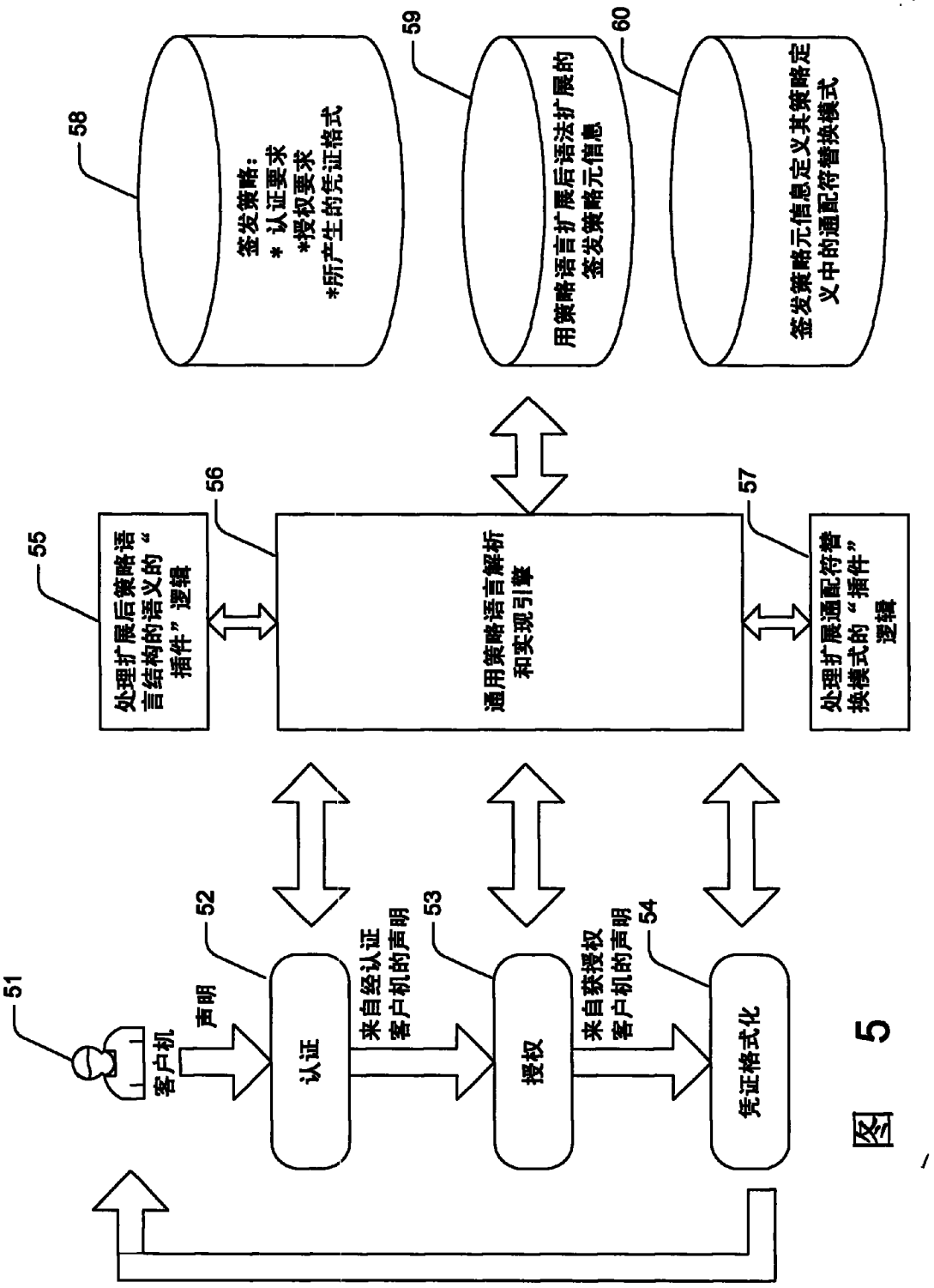

58
签发策略：
* 认证要求
*授权要求
*所产生的凭证格式
59
用策略语言扩展后语法扩展的签发策略元信息
60
签发策略元信息定义其策略定义中的通配符替换模式
55
处理扩展后策略语言结构的语义的“插件”逻辑
56
通用策略语言解析和实现引擎
57
处理扩展通配符替换模式的“插件”逻辑
51
客户机
声明
52
认证
来自经认证客户机的声明
53
授权
来自获授权客户机的声明
54
凭证格式化

图 5