- **Okamoto身份识别方案**
- **Okamoto对Schnorr方案进行了改进，保证在假设Z_p中一个特定的离散对数问题是难解的条件下，改进方案是安全的。**
- **TA在Okamoto方案中除了要确定p,q,t和散列函数、签名方案等参数外，还要选择两个阶为q的元素α_1， $\alpha_2 \in Zp^*$，**
- **Sender在计算v时将选择两个秘密参数a_1和$a_2(0 \le a_1, a_2 \le q-1)$，且$v=\alpha_1^{-a1}\alpha_2^{-a2}(\text{mod } p)$。**
- **身份识别时，随机数k的选择也由一个变为两个：k_1和$k_2(1 \le k_1, k_2 \le q-1)$，且$\Gamma=\alpha_1^{k1}\alpha_2^{k2}(\text{mod } p)$。**
- **在Receiver选定随机数r后，Sender将计算两个Y：$Y_1=(k_1+a_1r)\text{mod } q$，$Y_2=(k_2+a_2r)\text{mod } q$送给Receiver，Receiver最后的验证相应地变为：$\Gamma=\alpha_1^{Y1}\alpha_2^{Y2}v^r(\text{mod } p)$。**
- **Okamoto将一个秘密参数a扩展成两个a_1和a_2，这样的改进看似不大，但却从根本上解决了安全性的证明。**

- 定理：假设Cheater知道一个值Γ，对这个Γ成功模仿Sender的概率$\varepsilon \geq (2^t - 1)^{-1}$，那么Cheater和Sender合伙以概率$1-1/q$能够在多项式时间计算$\log_{a_1} a_2$
- Sender选择了两个秘密参数而不是一个，所有可能的秘密参数有序对构成一个集合，这个集合中有q个对，它与Sender的(a_1, a_2)是“等价的”。
- 知道这个集合中两个不同的对就提供了计算离散对数c的有效算法。
- Sender知道其中一个对，如果Cheater能够冒充Sender，那么就能计算出集合中的一个对，并且与Sender的对不同的概率几乎为1。
- 因而Sender与Cheater一起能够找到两个不同的对，从而计算出c，也就产生了与TA假设的矛盾。
- 尽管Okamoto方案是安全的，但与Schnorr方案相比，执行时间几乎翻倍。

4.5 密钥交换协议

- 1.Blom 协议
- p 是素数，k是一个整数，k是使得网络中的任何k个用户合伙，方案仍是安全的最大值．即安全性条件为：至多k个不同用户的任何用户集必须不能确定两个用户间的密钥.
- 下面给出k=1的方案

- (1)每个用户任选一个公开的$r_u \in Z_p$,这些元素互不相同.
- (2)可信中心选择三个随机元素$a,b,c \in Z_p$,
- $f(x,y)=(a+b(x+y)+cxy) \bmod p$
- (3)对每个用户U，可信中心计算
- $g_u(x)=f(x, r_u)=(a+b(x+r_u)+cr_ux) \bmod p$
- 并在安全信道上送给U
- (４)如果U和V要通信，只要计算
- U计算$K_{uv}=g_u(r_v)$，V计算$K_{vu}=g_v(r_u)$
- $K_{uv}=K_{vu}$.
- 定理：k=1时的Blom方案对任何单个用户是无条件安全的

- 下面推广为k个情况
- 把函数改为 $\sum_{i=0}^{k}\sum_{j=0}^{k} a_{ij}x^i y^j$

这里$a_{ij}=a_{ji}$,对任意的i,j

- 2.Differ-Hellman 协议
- Diffie-Hellman密钥交换协议：
- 设p 为素数,g是Z_p的生成元，都公开；
- （1）A随机选择数x，向B发送信息（$g^x \bmod p$）；
- （2）B随机选择数y，向A回送信息（$g^y \bmod p$）；
- A完成计算$(g^y \bmod p)^x$, B完成计算 $(g^x \bmod p)^Y$
- 显然，$(g^y \bmod p)^x = (g^x \bmod p)^y = g^{xy} \bmod p$便是共享密钥；
- 对于入侵者，他可以截获以上信息，但是无法利用p, g, $g^x \bmod n$ 计算x，即计算上不可行；但此算法仍有缺点，考虑以下情况：
- A,B是通信双方，T是入侵者，x为A的随机数, y为B的随机数, z为T的随机数,
- (1) A向B发送信息（$g^x \bmod p$）；
- (2)入侵者T截获了信息并向B发送 ($g^z \bmod p$);
- (3) T接着将此随机数$g^z \bmod p$ 发送给A，A与T产生共享密钥$g^{xz} \bmod p$ ；
- (4) B向A发送的$g^y \bmod p$也被T截获并与B产生共享密钥$g^{yz} \bmod p$;
- 由于A、B都把T看作为合法的B和A, 因此, A用$g^{xz} \bmod p$ 作为与B通信的共享密钥，而B把$g^{yz} \bmod p$作为与A通信的共享密钥，
- 实质上, A与B的通信结果都是通过T完成的,这样,A和B之间的通信总会被T捕获，这种攻击被称为中间人拦截攻击

- 增加一个签名方案Sig, Ver
- 引进可信中心.
- 每个用户I有自己的证书$C(V)=(ID(I),Ver_U, Sig_{TA}(ID(U),Ver_U)$
- (1)A随机选择数x，向B发送信息（$g^X \bmod p$）；
- (2)B随机选择数y，计算$(g^X)^Y \bmod p$和
- $Y_B=Sig_B(g^X,g^Y)$, 向A回送信息（$C(B),g^y \bmod p, Y_B$）
- (3)A计算$(g^Y)^X \bmod p$, $Y_A=Sig_A(g^X,g^Y)$,使用Ver_B验证Y_B，使用Ver_{TA}验证C(B), 通过验证后A向B回送信息（$C(A), Y_A$）；
- (4)B使用Ver_A验证Y_A，使用Ver_{TA}验证C(A)

4.6 CA与数字证书

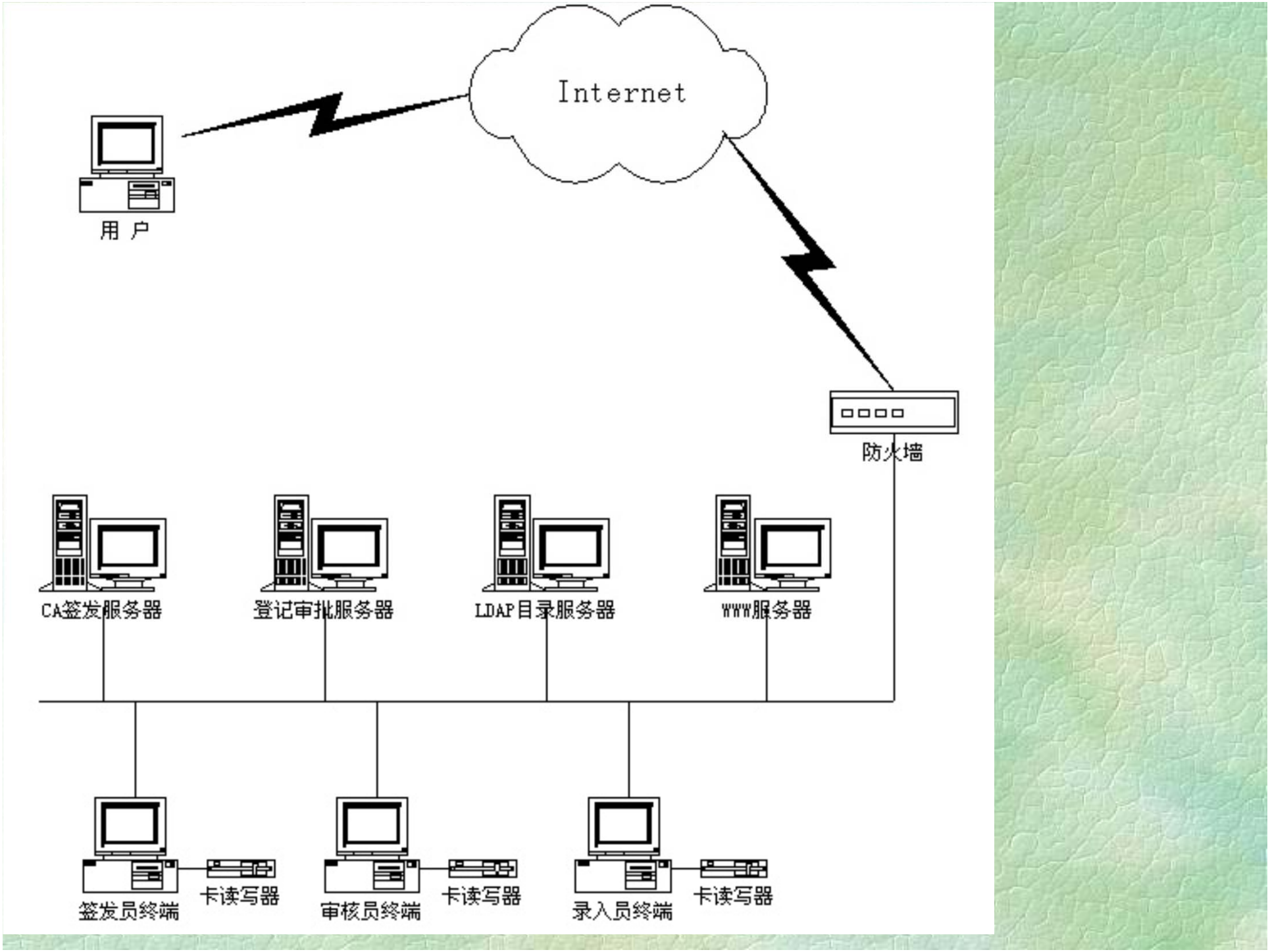
- 在网络安全和电子商务等实际应用中，通常用公钥密码算法来协商会话密钥
- 每次通信时临时产生密钥
- 而用对称密钥密码算法来加密要秘密传输的数据
- 数字信封

- 例1:A要与B通信
- A的动作：
- 1)A从公钥库中得到B的公钥PK_B；
- 2)A随机产生一个会话密钥K；
- 3)A用B的公钥PK_B对K加密产生C_1；
- 4)A用K加密要传输的数据DATA，得到C_2；
- 5)A将$C_1||C_2$发送给B
- B的动作：
- 1)B接收到$C_1||C_2$；
- 2)B用自己的私钥SK_B对C_1解密，得到K；
- 3)B用K对C_2解密得到DATA。

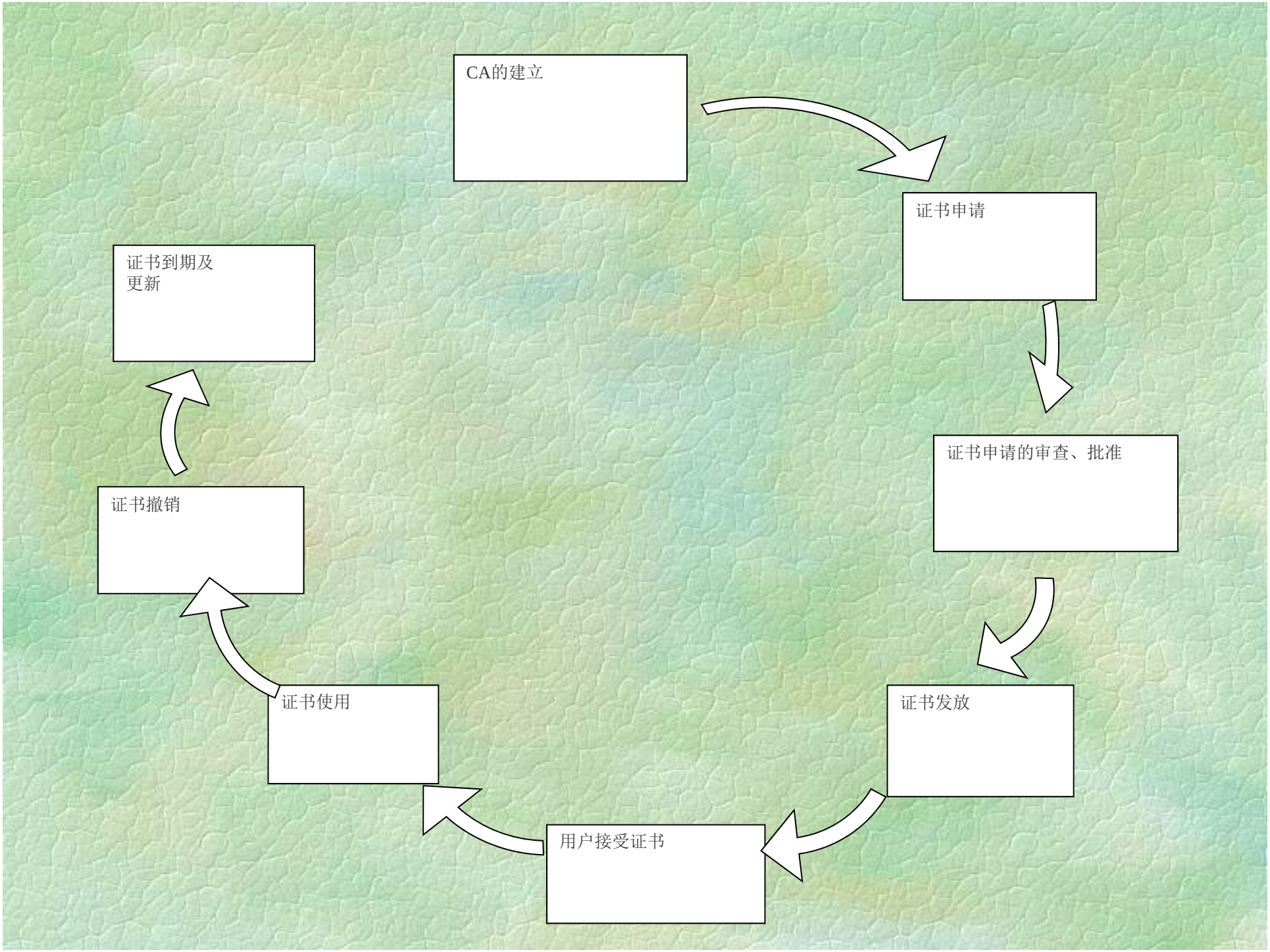
- 为了防止这种攻击，必须有一个大家都信任的第三方机构将公钥库中用户名字与用户的公钥绑定起来。
- 大家都信任的第三方结构就是CA
- 绑定后的公钥就构成了用户的数字证书，简称证书
- 绑定的途径就是数字签名技术。

- **4.6.1 基本概念**
- CA又叫证书授权中心，是用来为用户发放证书的权威机构。
- 功能是：接收用户的申请，为用户发放证书，并维护用户的证书库。
- 所谓证书实际上就是用户在电子世界中的身份证，用户可以通过它来识别通信对方的身份，用它来协商会话密钥等。证书的格式有一定的国际标准。
- X.509证书的格式如下。
- 版本号，序列号，签名算法，签发者，有效期，用户名，公钥信息，扩展域，签名算法，CA签名 。

- 证书吊销列表(CRL)，即证书黑名单，存放不再使用的用户证书信息。
- CA认证中心包括管理服务器、登记审批服务器、证书签发服务器、LDAP目录服务器、录入员终端、审核员终端、签发员终端。总体架构图如图1所示。
- 根据系统的容量，将签发员终端、审核员终端、录入员终端单独划出来，构成受理点。一个CA认证中心可以有多个受理点。


Internet
用 户
防火墙
CA签发服务器
登记审批服务器
LDAP目录服务器
WWW服务器
签发员终端
卡读写器
审核员终端
卡读写器
录入员终端
卡读写器

- 建设一个具体的可运作的CA包括以下几个基本部分：物理环境的建立；硬件设施的配备；安全管理策略的制定；软件模块的开发；人员的培训。
- 其中软件模块的开发包括：CA认证中心的建设；RA受理点的建设；CA服务网站的建设
- 按照用户的不同类型，可以将证书分为：个人身份证书；企业身份证书；服务器身份证书；信用卡身份证书（符合SET协议）


CA的建立
证书申请
证书申请的审查、批准
证书发放
用户接受证书
证书使用
证书撤销
证书到期及
更新

- 4.6.2申请签发证书流程
- 1.证书申请
- 证书申请方式包括网上申请和亲自到认证机构的受理点当面申请。
- 2.证书审批
- 审核员到中心登记审批服务器抽取未审核的证书申请者信息，根据证书申请者提供的信息进行审核
- 包括身份确认、物理材料的正确性、资料完整性、申请证书使用的合理性等方面的审核。
- 审查通过后，提交给登记审批服务器
- 登记审批服务器将经过审核的证书申请表信息发送给CA中心证书签发服务器

- 3.签发证书
- 签发员根据证书签发服务器上的经过审核的证书申请表信息，进行证书的签发。
- 首先签发员验证证书申请表的签名，确认证书申请表没有被篡改。若签发操作员发现签名不正确，则拒绝证书申请。若签名验证通过，则同意用户的证书请求
- 采集系统的随机信号产生随机数，由此按照RSA公钥算法算出公钥和私钥。
- 签发员签发证书并同时更新LDAP服务器和CA中心数据库。
- 证书的内容包括证书申请表的全部内容和发证机关的信息，证书的用途在扩展域中标明。
- 证书的最后用发证机关的私钥签名。
- 产生的证书及证书申请者的私钥(加密存放，按照PKCS#5)制作成介质(IC卡、磁盘)，由证书申请者带走。

- 4.使用证书
- 用户带走证书介质(IC卡、磁盘)后，利用IC卡读写器或软驱与相关安全应用软件(如SSL协议软件、SET协议软件和其它要使用证书的软件)使用证书。

- 4.6.3申请撤消证书流程
- 1.撤消条件
- 证书主体受到损坏、更改或非授权使用；私钥或用户口令受到损坏、更改、泄漏或非授权使用；
- 持证人不再需要证书；现有证书不能满足持证人的需要；
- 法律法规或其它法令及政府行为的改变。

- 2.撤消流程
- 1)撤消申请
- 申请者来到CA的某个受理点RA申请撤消证书，向录入员提供证明其资格的相关证件和证明材料以及一些必要的证明信息。录入员将申请者提供的信息输入录入员终端，并传给登记审批服务器。
- 2)撤消审核
- 审核员到中心登记审批服务器抽取未审核的证书撤消申请者信息，根据申请者提供的信息进行审核，包括身份确认、物理材料的正确性、资料完整性、申请证书撤消的合理性等方面的审核。资料审查通过后，提交给登记审批服务器。此时登记审批服务器将经过审核的证书撤消信息发送给CA中心证书签发服务器。

- 3)证书撤消
- 签发员根据证书签发服务器上的经过审核的证书撤消信息，进行证书的撤消。首先签发员验证证书撤消申请的签名，确认证书撤消申请没有被篡改。如果正确的话，则同意用户的证书撤消申请，同时更新LDAP服务器(包括CRL列表)和CA中心数据库。

- **4.6.4 证书的编码方法**
- **有多种编码方法将对证书的信息进行编码。**
- **常见的有DER编码、BASE64编码等。**

- **4.6.5 SSL协议简介**
- **安全套接层(SSL)最早是由Netscape公司开发的安全协议。**
- **该协议的第3个版本已经进行了公开的评论，并在业界的参与下设计，作为Internet的文档出版。**
- **IETF形成了TLS工作组，制定作为Internet标准的SSL。**
- **TLS的第1个版本实际上是SSLv3.1，和SSLv3十分接近，并且保持了向后兼容性。**
- **介绍SSLv3。**

- **1.SSL体系结构**
- **SSL是在TCP层之上的一个安全协议，发信方和收信方在建立TCP连接之后，通过它可以协商建立安全通信环境，具体包括身份认证(通过对对方的证书的验证来实现)、密码算法和密钥的协商、MAC秘密的协商等。**
- **安全环境建立起来之后，就可以收发SSL数据(SSL记录)。**
- **SSL不是一个单独的协议，而是两层协议**

<table>
<tr><td>SSL 握手协议</td><td>SSL 更改密码规格协议</td><td>SSL 警报协议</td><td>HTTP</td></tr>
<tr><td colspan="4">SSL 记录层协议</td></tr>
</table>

- SSL是一个包括两层共四个协议的安全套接层协议。
- 它的下层是SSL记录层协议，负责数据的分段、压缩、计算MAC码、加密等操作。
- 上层共三个协议（握手协议、改变密码规范协议和警告协议），这三个协议起着建立安全环境，更改密码参数和通信告警的作用，同时为SSL记录层提供需要的密码学参数。

- 连接：连接是指能提供合适服务类型的传输(在OSI分层模型中的定义)。

  对SSL，这样的连接是对等关系。

  连接是暂时的，每个连接都和一个会话相关。
- 会话：SSL会话是指在客户机和服务器之间的关联。

  会话由握手协议创建。

  会话定义了一组可以被多个连接共用的密码安全参数。

  对于每个连接，可以用会话来避免对新的安全参数进行代价昂贵的协商。

- 会话状态：包括
- 会话标识符(Session identifier)——用来标识会话；
- 对等实体证书(Peer certificate)——对等实体的X.509v3证书；
- 压缩算法(Compression method)——压缩数据的算法；
- 加密规格(Cipher spec)——指定数据加密算法、用于MAC计算的散列算法(MD5，SHA-1等)和密码属性，如hash_size等;主秘密(Master secret)——由服务器和客户端共享的48位密钥
- 是否可恢复(Is resumable)——用来确定会话是否可用于初始化新连接的标志。

- 连接状态：包括
- 服务器和客户机随机(Server and client random)——每个连接中，服务器和客户机选择的字节流；
- 服务器写MAC秘密(Server write MAC secret)——对服务器写的数据进行MAC操作时使用的秘密；
- 客户端写MAC秘密(Client write MAC secret)——对客户端写的数据进行MAC操作时使用的秘密；
- 服务器写密钥(Server write key)——服务器加密、客户端解密用的分组密码算法的密钥；
- 客户端写密钥(Client write key)——客户端加密、服务器解密用的分组密码算法的密钥；
- 初始化向量(Initialization vectors)——CBC加密模式中所用的初始向量；
- 序列号(Sequence number)——对于每次连接，每一方为传送和接收的消息保留的单独的序列号。当一方发送或接收变化的密码规格消息时，将适当的序列号置0。

- 2．SSL握手协议
- SSL握手协议位于SSL记录层协议之上，用来会话状态的密码学参数。
- 当SSL客户机和服务器第一次通信时，他们要对协议的版本号、选择的密码算法、可选的双方身份认证进行协商，并使用公钥加密技术来产生共享的秘密。握手协议由一系列客户机与服务器的交换信息组成。
- 所有消息都具有如下的格式：
- 1字节　　3字节　　>=1字节
- 类 型　　长 度　　内 容

SSL 握手协议的消息类型

消息类型	参数
hello_request	Null
Client_hello	版本，随机数，会话 id，密码组，压缩模式
Server_hello	版本，随机数，会话 id，密码组，压缩模式
Certificate	x.509v3 证书
server_key_exchange	参数，签名
certificaterequest	类型，授权
Server_hello_done	Null
certificate_key_exchange	签名
client_key_exchange	参数，签名
Finished	哈希值

- SSL握手协议概括如下：
- 客户机发送给服务器一个client_hello消息，服务器必须用server_hello对这个消息做出响应，否则会产生一个严重的错误，连接就会失败。
- 客户机和服务器使用client_hello和server_hello来建立客户机和服务器之间的安全通信能力。
- client_hello和server_hello建立下面的属性：
- Protocol Version, Session ID, Cipher Suite, and Compression Method。而且，产生和交换两个随机数：ClientHello. random和 ServerHello.random。

- 在hello消息之后，如果客户要对服务器进行认证，服务器就发送他的certificate。
- 如果服务器被认证了，它可能要求客户机发送它的证书。
- 服务器可以发送server_hello_done消息了,表示握手协议的hello消息阶段已经结束。
- 服务器等待客户机的响应。如果服务器已经发送了一个certificate_request消息，客户机必须发送certificate或者一个no_certificate警告. 接着客户端发送一个client_key_exchange消息，这个消息中的内容取决于hello消息阶段协商的公钥算法。

- 客户端发送一个change_cipher_spec消息，同时将pending Cipher Spec拷贝到current Cipher Spec。
- 然后，客户端立刻在新算法、新密钥和新秘密下发送一个finished消息。
- 服务器也发送它自己的change_cipher_ spec消息，将pending Cipher Spec拷贝到current Cipher Spec，在新的Cipher Spec 下发送finished。
- 完成了整个的握手过程。服务器和客户机就可以交换应用层数据。

- 改变密码规格协议是使用SSL记录层协议的3个特定协议之一，也是其中的最简单的协议。
- 该协议由单个消息组成，该消息只包含一个值为1的单个字节。
- 该消息的唯一作用就是使未决状态(peding state)拷贝为当前状态(current state)，更新用于当前连接的Cipher suite

- 警告协议用来为对等实体传递SSL的相关警告。当其它应用程序使用SSL时，根据当前状态的确定，警告消息同时被压缩和加密。
- 警告协议的每条消息有两个字节。第1个字节有两个值，warning(1)和fatal(2)来表示消息的严重性。
- 如果是fatal，SSL立即终止该连接。
- 同一会话的其他连接也许还能继续，但在该会话中不会再产生新的连接。
- 第2个字节包含了指示特定警告的代码。

- SSL记录层协议为SSL连接提供了两种服务：
- 机密性:使用握手协议协商好的密码算法对数据加密。
- 完整性:使用握手协议协商好的MAC密钥进行MAC操作

- 从通信量分析、主动攻击、重放攻击、密码组回滚攻击以及中间人攻击这五种常见的安全攻击入手，分析SSL的抵御攻击能力。
- 1 通信量分析
- 通信量分析是一种被动攻击，但是恶意攻击。
- 目标在于，通过检查包中未加密的域及属性，以获得受保护会话的机密信息。
- 例如，通过检查未加密的源和目标的IP地址（甚至TCP端口号），或者检查网络通信流，一个通信量分析专家可以确定通信各方的通信内容，使用的服务类型，有时甚至还可以获得有关商业或个人关系的信息。
- 实际上，用户大都认为通信量分析的威胁相对来说是无害的，因此SSL不会试图阻止这种攻击。忽略通信量分析看来是一种可行的设计方案。

- 在SSL体系结构中，通信量分析仍会造成一些威胁。
- 检查密文的长度会暴露出有关SSL或被SSL加密的网站通信中用到的URL请求的信息。当一个Web浏览器通过加密传输（如SSL）连接到一台Web服务器时，包含有URL信息的GET请求会以加密方式进行传输。
- 浏览器要下载哪一个页面，这种信息是保密的——显然，具有URL方面的知识，对于一个想下载整个网站的人来说，这已经足够了——而且，通信量分析可以获得Web服务器的身份信息，请求的URL长度，以及从Web服务器所返回的html数据的长度。这个漏洞往往使得窃听者能够发现用户访问了哪些Web页面。
- 上述弱点之所以出现，是因为密文长度暴露了明文的长度。SSL在块加密模式中支持随机填充，而在流加密模式中却不支持。

- 2 主动攻击
- SSL能抵御主动攻击以安全地保护机密数据。
- IPSEC中的一个重要的主动攻击是剪贴攻击。要获得机密性，光对链接进行加密是不够的——接收端也必须防止敏感数据在无意中被泄露出来。剪贴攻击的攻击方式就是谋求接收端的信任，使之对敏感数据进行解密并将这些数据泄露给它们。
- SSL 3.0能够阻止剪贴攻击。阻止这种攻击的一种局部性防御策略是为每一个不同的上下文使用独立的会话密钥，这样便能在不同的连接之间，以及连接的不同方向之间阻止剪贴操作。SSL已经为每个连接方向使用了独立的密钥。
- 但是，这种机制无法阻止传输中一个方向上的剪贴操作。应用最广泛的防御策略是在所有加密包上使用强大的认证机制，以阻止敌方对密文数据的修改。SSL记录层确实采用了这种防御策略，因此剪贴操作被完全阻止住了。

- 3 重放攻击
- 光靠使用报文鉴别码（MAC）不能防止对方重复发送过时的信息包。SSL通过在生成MAC的数据中加入隐藏的序列号，来防止重放攻击。这种机制也可以防止被耽搁的，被重新排序的，或者是被删除数据的干扰。
- 序列号的长度是64bit，因此打包不会有问题。另外，序列号由每个连接方向分别维护，而且在每一次新的密钥交换时进行更新，所以不会有明显的弱点。

- 4 密码组回滚攻击（CipherSuite Rollback attack）
- SSL 2.0密钥交换协议中有一个严重的缺陷：主动攻击者能够在暗地里迫使一个家庭用户使用功能被削弱的出口加密算法，即使通信双方都支持并首选了较高等级的算法。这就是密码组回滚攻击（CipherSuite Rollback attack），它通过编辑在hello报文中发送的所支持密码组的明文列表来达到自身的目的。
- SSL 3.0 修正了这个缺陷，它使用一个master_secret来对所有的握手协议报文进行鉴别，这样一来，便可在握手结束时检查出敌方的上述行为，如果有必要，还可结束会话。

- SSL 3.0中防止修改握手协议报文的机制。
- 所有初始的握手协议报文在传送时都是未保护的，此时密钥交换协议会将当前会话状态改为未决的会话状态，而不是修改当前使用中的各个参数。
- 在协商完成之后，通信的每一方都发送一个短的更改密码规格报文（change cipher spec message），该报文仅仅是警告对方将当前状态升级为未决的会话状态。虽然change_cipher_spec报文未受保护，但是新的会话状态还是将以下一个报文为开始。紧跟在change_cipher_spec报文之后的是结束（finished）报文，它包含了一个报文鉴别码（MAC），此MAC由被master_secret加密过的所有握手协议报文计算得出。（基于特殊的非安全性因素，change_cipher_spec报文和alert报文在finished报文中没有进行鉴别。）48字节长的master_secret从未被泄露出去，而且会话密钥由它产生。这就保证了即使会话密钥被人截获，master_secret仍可安然无恙，所以握手协议报文能够安全的得到鉴别。
- Finished报文使用新建的密码组（ciphersuite）对自身进行保护。通信各方只有在收到对方的finished报文并对其进行核实后，才会接收应用层的数据。

- 5 中间人攻击（man_in_the_middle attack）
- SSL 3.0中包含了对Diffie-Hellman密钥交换进行短暂加密的支持。Diffie-Hellman是一种公开密钥算法，它能有效地提供完善的保密功能，对于SSL来说是一个有益的补充。在SSL 3.0 Diffie-Hellman密钥交换系统中，服务器必须指定模数和原始根（这两个数均为素数），以及Diffie-Hellman的指数。为了防止服务器端产生的陷门，客户端应该对模数和原始根进行仔细的检查，看它们是否为固定公共列表上的可靠数值。在SSL 3.0中，通过对服务器端的Diffie-Hellman指数的鉴别，可以抵御众所周知的中间人（man-in-the-middle）攻击。（匿名客户不必拥有证书。）另外，在SSL 3.0中并不支持具有较高性能的Diffie-Hellman变量，如较小的指数变量（160bit）或椭圆曲线变量。

- 6.传输层安全
- “传输层安全”(Transport Layer Security,TLS) 建立在Netscape所提出的SSL3.0协议规范基础上；
- 传输层安全性的标准协议，TLS和SSL3.0,两者之间存在着显著的差别。
- 版本号
- TLS记录格式于SSL记录格式是相同的，并且首部的字段也具有相同的含义。不同点在于版本的值。对于TLS当前的草案，主要版本是3，次要版本是1。
- 报文鉴别码
- SSLv3和TLS的MAC模式在实际的算法上略有不同。虽然双方都采用RFC2104定义的HMAC算法，但是其填充字节的处理方式有所区别。SSLv3采用将填充字节和密钥连接起来达到块长度，而TLS是采用异或的方式。两种情况下的安全程度应该是相同的。
- 伪随机函数
- 为了生成或验证密钥，TLS使用了称为PRE的伪随机函数来将密钥扩展成数据块，目的是为了使用相对小的共享密值，但却以一种对于在散列函数和MAC上的攻击类型是安全的方式来生成更长的数据块。

- 告警代码
- 除了没有证书的告警之外，TLS支持SSLv3中定义的所有告警代码。TLS定义了很多补充代码。
- 密文族
- SSLv3和TLS可用的密文族之间存在几个小的差别：
- 密钥交换：TLS支持除了Fortezza之外的SSLv3的所有的密钥交换技术。
- 对称加密算法：TLS包括了SSLv3建立的多有除了Fortezza之外的对称加密算法。
- 填充
- 在SSL，在对用户数据加密前增加的填充自己，使得被加密的整个数据长度达到密文块长度整数倍的最小数目。在TLS中，填充可以是任意使得整个长度达到密文块长度整数倍的数目，最大达到255个字节。

- **4.6.6 SET协议**
- **1995年，信用卡国际组织、资讯业者及网络安全专业团体等开始组成策略联盟，共同研究开发电子商务的安全交易系统。**
- **1996 年6月，由IBM，Master Card International，Visa International，Microsoft，Netscape，GTE，ViriSign，SAIC，Terisa 共同制定的标准SET（Secure Electronic Transaction）正式公告，涵盖了信用卡在电子商务交易中的交易协定、信息保密、资料完整即数字认证、数字签名等。**
- **该协议由若干协议形成，它们是：STT(Visa/Microsoft)、SEPP(MasterCard) 和 iKP 协议族(IBM)。**

- 在1997年就推出了SET1.0版，但它的推广应用较缓慢，原因是昂贵、互操作性差和难以实施，
- 它提供了多层次的安全保障，复杂程度显著增加；
- 另一个原因是由于SSL的广泛应用。
- 此外，银行的支付业务不光是卡支付业务，而SET支付方式适应于卡支付，对其他支付方式是有所限制的。
- SET协议支持"B to C"类型的电子商务模式，即消费者持卡在网上购物与交易的模式，而不能支持B to B模式。

1.SET 协议的运作流程

SET 主要用于消费者与商店、商店与收单银行（付款银行）之间。其运作方式如图所示：

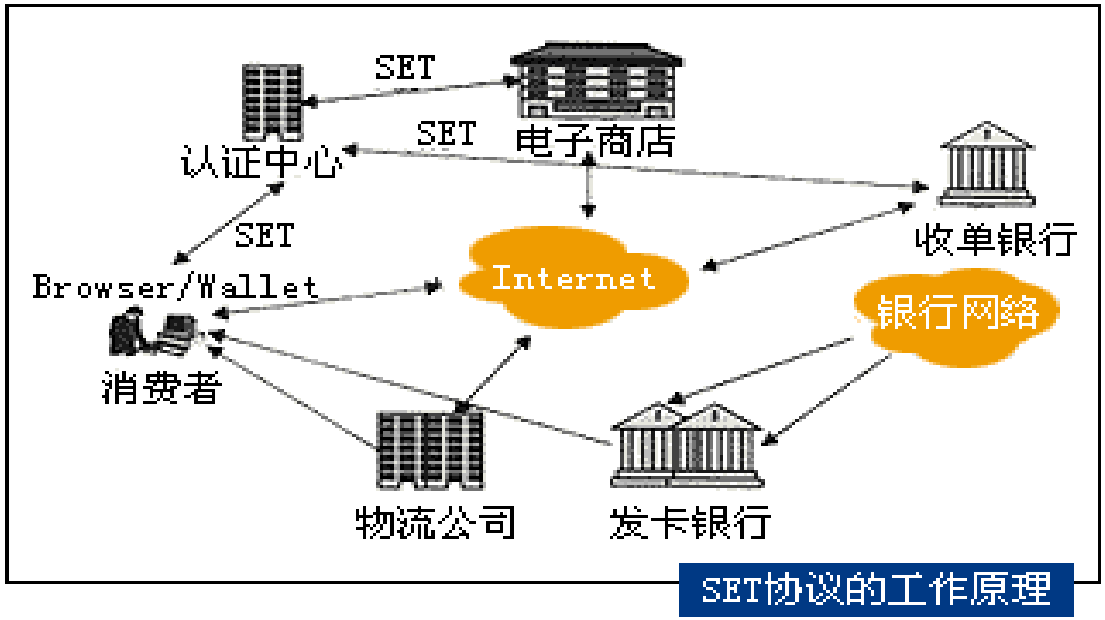


图 5 SET 的简易流程

- 在这个过程中，CA扮演了系统的很重要的角色。SET标准着重的是其交易安全及隐密性。
- 数字证书为其核心，它提供了简单的方法来确保进行电子交易的人们能够互相信任。
- 信用卡组织提供数字证书给发卡银行，然后发卡银行再提供证书给持卡人；同时，信用卡组织也提供数字证书给收单银行，然后收单银行再将证书发给特约商店。
- 在交易时，持卡人和特约商店符合SET的规格软件会在资料交换前分别确认双方的身份，也就是检查由授权的第三者所发给的证书。
- 在SET协定中，有持卡人证书、特约商店证书、支付网关证书、收单银行证书和发卡银行证书等五种证书。

- 持卡人的证书必须由发卡银行来颁发。
- 在首次上网购物之前，持卡人必须先通过一个客户端程序将包括姓名、卡号、卡片有效期、邮寄地址等可以证明持卡人身份的基本资料发给发卡银行。这些资料使用银行的公钥加密，可安全地送至银行。
- 发卡银行确认此帐户正确无误后，便发给持卡人一张具有电子安全数字签章的证书。
- 持卡人只要将证书储存在电脑上，即可进行电子购物。
- 同样，商店也必须取得收单银行的电子证书才可以接收SET方式的支付。商店要将它的基本资料发送给收单银行，收单银行在确认无误后发出一张数字证书，允许它从事电子交易。

- 2. SSL和SET协议的比较
- SET和SSL除了都采用RSA公钥算法以外，二者在其他技术方面没有任何相似之处。
- 而RSA在二者中也被用来实现不同的安全目标。
- SET是一个多方的报文协议，它定义了银行、商家、持卡人之间必需的报文规范
- SSL只是简单地在两方之间建立了安全连接。
- SSL是面向连接的，而SET允许各方之间的报文交换不是实时的。
- SET报文能够在银行内部网或者其他网络上传输，而SSL之上的卡支付系统只能与Web浏览器捆绑在一起。

- SET与SSL相比具有如下优点：
- ①SET为商家提供了保护自己的手段，使商家免受欺诈的困扰，使商家的运营成本降低。
- ②对消费者而言，SET保证了商家的合法性，并且用户的信用卡号不会被窃取，SET替消费者保守了更多的秘密使其在线购物更加轻松。
- ③银行和发卡机构以及各种信用卡组织来说，因为SET可以帮助它们将业务扩展到Internet这个广阔的空间，从而使得信用卡网上支付具有更低的欺骗概率，这使得它比其他支付方式具有更大的竞争力。
- ④SET对于参与交易的各方定义了互操作接口，一个系统可以由不同厂商的产品构筑。

- 提供这些功能的前提是：
- SET要求在银行网络、商家服务器、顾客的PC上安装相应的软件。这些阻止了SET的广泛发展。
- SET还要求必须向各方发放证书，这也成为阻碍之一。所有这些使得SET要比SSL昂贵得多。
- SET的另外一个优点在于：它可以用在系统的一部分。例如，一些商家正在考虑在与银行连接中使用SET，而与顾客连接时仍然使用SSL。这种方案既回避了在顾客机器上安装钱夹软件；同时又获得了SET提供的很多优点。
- 在SET中，交易凭证中有客户的签名，银行就会有客户曾经购物的证据。提供这种能力的特性在密码学中称之为认可。它所提供的可靠性的前提是客户必须保证私人签字密钥的安全。

4.7 公开密钥基础设施

- 网络环境下广泛开展的电子商务、电子政务等社会化应用已成为现实，怎样奠定网络空间的运作次序成为必须解决的问题。
- PKI就是一个颁发用户公开密钥的系统，所有的PKI都应该执行以下两个基本操作：
- 发放证书：将公开密钥与个人、组织、或其它实体、甚至类似于许可证或凭据等信息捆绑在一起。
- 确认：判断一个证书是否可以被正确使用。

- # 4.7.1 证书的类型以及所包含的内容
- 发放证书是所有PKI的基本功能。首先就是定义一种格式证书，PKI以这种格式传播公钥、传播与公钥有关的信息或两种均传播。
- 证书依靠所包含的信息的类型分成两类：
- 一类是身份证书，该证书能简单地鉴别一个实体（称为证书的主体（subject）），证书中列出了实体的公钥。
- 另一类是凭据证书，包含了实体属性的证书，属性可以是：成员关系、角色、许可证或其他访问权限。使用凭据证书可以鉴别许可证（例如“有权访问计算机xyz”）、凭据（例如，“一个合法的股票代理人”）或其它属性（例如“是A公司的市场部“)。

- 身份证书提供了认证、数据完整性和秘密性。
- 虽然认证、数据完整性和秘密性解决了安全传输问题，但是在许多应用中，这些服务还不足以（或者管理起来很烦琐）提供授权，因为在授权中，实体的身份考虑得相对少，更多的是考虑属性（如角色、帐户限制），这些属性决定授权。
- 从管理方便、安全、互操作性等方面的考虑，通常把这些信息从身份中分离出来，这些授权信息也需要象类似于公钥证书方式那样保护起来，这就是凭据证书。

- 证书的用户就是依赖于证书包含的信息的实体。
- 证书用户信任颁发机构颁发的真实的证书，
- 对于身份证书，证书能鉴别主体（subject）和它的公钥的对应关系；
- 对于凭据证书，证书正确描述主体（subject）的属性。
- 证书的颁发者CA。

- **4.7.2 PKI的各个CA、证书的主体(subject)、证书的用户三者之间的关系**
- **在PKI的各个CA、证书的主体(subject)、证书的用户两两之间的关系中，有可能CA不同于主体(subject)和用户，也可能用户和主体(subject)本身就是CA。**
- **当三个实体不同时，要明确这三者之间相互的认识程度如何以及运作的PKI要求的最小的相互之间通晓程度是多少。**

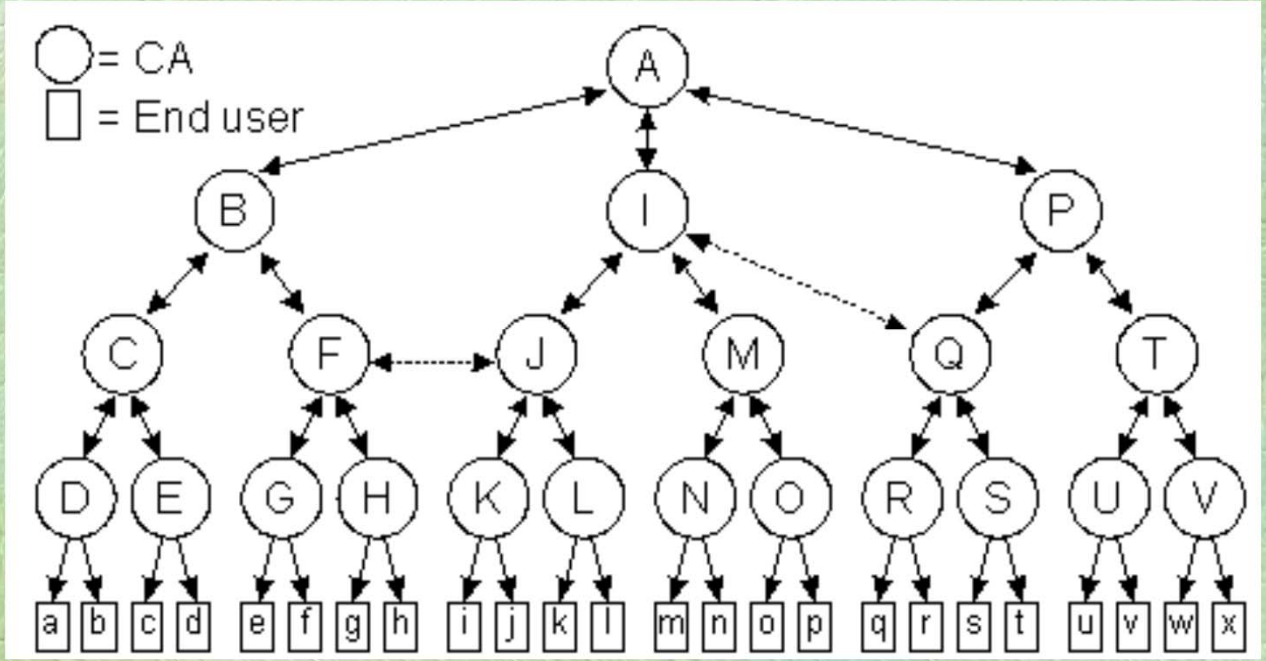
- 一些PKI要求用户只在一个CA上建立或代理信任，还有一些PKI允许用户信任多个CA。
- 有时CA必须将他们的信任放在其它CA或放在这些CA的主体(subject)，
- 一些PKI允许一个 CA 可以只信任另一个CA的某些证明。
- 例如，第一个CA可以有一个策略：“我只信任另一个CA颁发的将email地址与公钥捆绑在一起的证书”，而且该策略以一种自动要遵守的方式表达。

- 处理信任问题的同时也产生了责任问题，当PKI用于保护敏感信息的安全性时，明确谁负责那些重要的信息是重要。
- 信任问题机制，不是单靠设计PKI来解决。一般都是将某些而不是全部信任给某人，没有一个预先定义好的模型能包括所有信任分支，一个PKI只是一个表达信任关系的工具。任何试图做更多事的PKI均缺少灵活性。
- 一个实体能够信任的证书是怎样确定的
- 信任是怎样建立的
- 怎样限制或控制信任

4.7.3 CA的排列

- 只有一个CA显然不实际，因此，大多数的PKI允许一个CA给其它CA发放证书。
- 一个CA 可以告诉它的用户：可以信任另一个CA所颁发的证书的内容。
- 例如，用户A由CA_A颁发证书，用户B由CA_B颁发证书，用户A和用户B通信时，用户A需要CA_B的公钥和CA_A对CA_B的公钥颁发的证书，这样用户A得到CA_B的公钥就可以获得用户B的公钥。
- 在这里，用户A和用户B的证书是终端用户证书，CA_A颁发给CA_B的证书称为交叉证书。

- 一般地，用户A到用户B的路径中可以有任意个CA。
- 为得到用户B的公钥，用户A通过某种手段直接获得用户B的公钥，或者要依次验证证明路径（certification path valediction）中每个CA的证书都是正确的才能获得用户B的证书，这个过程就是证明路径的验证。
- 证明路径的长度是用户A和用户B之间CA的个数，或用户A获得用户B的密钥需要验证的CA的个数。



- **PKI中的CA关系控制了PKI的可扩展性。一个 PKI要运作于全局，它的功能必须达到上亿的用户才说明它是可行的，同时必须很容易找到证书的路径并且路径不要太长，**
- **“web信任”就不能很好地扩展，尽管在一个全局web 中，证书路径的平均长度不会很长，但在一个自由的web中寻找路径相对困难，同时还有如何代理信任和如何获得代理的问题。**

- 虽然自上而下层次模型能很好地扩展，但也产生很多问题
- 所有用户必须把顶层的CA作为他们的根CA，这需要所有用户在使用PKI之前，先获得顶层CA的拷贝，而且任何时候都要绝对信任顶层CA，这种自上而下的层次不适合全局PKI。
- 普通层次方式允许任一CA作为一个证书路径的根，然而结构仍然依赖于上层CA，特别是顶层CA，在普通层次方式中大量的证书路径经过顶层CA，这要求PKI的用户默认信任顶层CA。
- 如果顶层CA的私钥泄露，那么普通层次方式中两个实体之间的信息就可以被伪造，从而这个系统成为一个易受攻击的目标。
- 普通层次方式也面临可能证书路径过长的危险，从而导致与“web信任”的同样问题即信任差异问题。
- 交叉证书有助减少认证路径的长度，但又有增加路径复杂性的危险。

- 信任计算
- 付钱期限，送货期限，货物质量，价格
- 评价的权重
- 研究建立一定的评价机制

- 4.7.4 证书确认的方法
- PKI的第二个基本过程是证书的确认。证书中的信息会变化，证书的用户确保证书的正确有两种基本的确认方法：
- 1)在线确认：用户每次使用证书时要求CA直接对证书确认。
- 2)离线确认：CA是否在证书中包括确认日期？一对日期定义了证书中信息有效的时间范围。
- PKI可以用其中的一种方法或两种都用。

- 4.7.5 证书废止
- 与确认方法密切相关的是废止证书。证书的废止就是让用户知道一个证书中的信息不再是正确的。
- 如果每次使用证书都要让CA在线确认，那么废止问题变得很一般，因为CA能简单地告诉证书不再有效。
- 但是当使用确认日期进行离线确认时，证书废止问题就显得非常重要(特别是在私钥泄露时)。

- 当没有在线方式时，最普通的废止方法是使用证书废止列表(CRL)。
- 一个 CRL是一列CA签名并定期颁发的废止证书列表
- 用户在确认过程中要求用户检查最新的CRL，以确保用户要使用的证书不在废止证书列表之列。
- 访问CRL所关心的问题之一是：
- 在CA接收到废止一个证书的通知与CA颁布下一次CRL这一段时间里发生了什么。
- 废止的证书将只能出现在下一次的 CRL中，因此任何检查CRL的用户只能根据CRL列表检查一个证书是否被废止，如果CRL没有列出某一个证书，其实该证书已被废止，那也只能假定这个证书是正确的，并继续使用，CRL的时间间隔问题。

- 访问CRL所关心的另一个问题是CRL的大小。
- 一个CA需要证明上千甚至数百万的主体(subject)，即使是一个给定人群，其废止速率也是不可预测时，因此CA的CRL就可能变得很大。
- 当一个CRL很大时，用户就很难检索。
- 对CRL也要有签名，所以当使用CRL前要先验证签名，而且在一个大CRL中确认签名和处理实体所需的时间显得很重要。

- 这些问题导致了几种改进访问CRL的方法。
- 第一种办法是：根据不同的目标建立不同的CRL。
- 例如，为不同的废止原因(或不同的主体(subject))发布不同CRL。
- CA可以发布一个用于日常废止的CRL(如一个证书的主体(subject)的鉴别信息发生变化)，一个由于安全泄露而废止证书的CRL，还可以为终端实体主体(subject)发放一个CRL，再建立与该CA进行交叉证明的其它CA的CRL。
- 这些措施有效地将一个大的CRL分成几个小的CRL。不关心日常废止的用户只需检查安全泄露的CRL。
- 同样在处理一个证书链时，用户只要检查某一个CA的CRL(直到路径的头)。
- 尽管这些步骤有利于减少CRL的大小，但仍不能解决CRL的时间间隔问题。

- 另一种改进访问CRL的方法是：delta-CRL。
- delta-CRL采用存储废止信息而不是存储CRL结构的方式来提高处理应用的时间，忽略本地数据库中没有发生变化的信息，只处理变化的信息 。
- 一个delta-CRL是一列在上次完整CRL颁发完后发生变化的CRL（CA签名）的列表。
- delta-CRL 允许频繁地公布废止通知，减少一个废止的证书被当作合法的证书使用的可能性。
- delta-CRLs也有利于解决CRL 大小的问题，证书验证系统开始时将完整的CRL拷贝下来，这样便于检查，以后不断地取回delta-CRL，通过更新变化的部分CRL达到更新完整的CRL拷贝。
- 在线废止和确认的方法还有待于进一步研究。
- 尽管在线访问方法可以避免CRL管理问题，但这种访问方式的带宽和处理请求的问题还有待于解决。

- 4.7.6 强身份认证和非否认
- 认证是使用 PKI的过程，也就是一个CA确认了一个实体和一个用户，或者说确认了证书，或者说实体已经被证实。
- 用户对证书信息的信任程度和对正确性的信任程度是衡量认证强度的方法。
- 在任何时候用PKI离线或在线做认证，都称之为带内认证。
- 使用传统方式，类似于电话或见面，叫做带外认证。

- 每个PKI都在力图减少对带外认证的需求
- 完全取消带外认证是不可能的，至少一个想使用PKI的人第一次需要CA确认身份或凭据时，不能使用PKI进行初始的确认，因为那里没有任何的CA担保这个人的身份或凭据，因此开始的过程需要带外认证。
- 根据身份信息和凭据信息的变化以及信息的更新，不同的 PKI使用不同的带外认证
- PKI对带外认证要求的程度部分取决于 PKI的设计者对非否认要求的程度。

- 如果PKI作为电子签名取代纸上签名的基础，那么就要着重考虑非否认。
- 用户A与她的CA的带外联系越密切, 她从事这种欺骗就越少。
- PKI提供的非否认程度还取决于许多非技术的因素如运行的PKI所在的法律和社会机构。
- 在PKI所控制的范围外还有许多影响非否认的因素。
- 实体如何管理他们的私钥。

- ## 4.7.7 匿名
- 一个PKI所给予的不可否认程度产生了另一个问题：匿名。
- 匿名是指用户只想获得某一特定情况下的信息时所需具备的PKI的条件。
- 一个不可否认的身份签名足以鉴别用户身份，但是在许多环境中，一个用户在行使某种权限时，不希望暴露自己的完全身份(当然不影响该权限的行使)，因为全部信息的暴露会导致别人利用这些信息进行阴谋活动。
- 一个PKI要求依据用户名、地址、电话号码、身份证号、工作单位、头衔来鉴别一个人的身份，这些信息对一个用户去银行开户是必须的，
- 但购物，不必要将这些信息提供给商家。
- 理想的PKI是既提供非否认的认证又通过匿名提供强保密性，凭据可以满足这些要求。

- 量子计算机时代的密码技术
- 2007 年2 月，加拿大D-Wave 公司成功研制出世界上第一台16 位商用量子计算机“猎户”（Orion），其量子计算芯片由铝和铌元素组成的超导材料制成，被液氦冷冻在-273.145℃温度下。
- 该量子处理器在基础构造和生产工艺上借鉴了现有半导体产业的成果，包含16 qubit。“猎户”采用一种混合型平台：使用普通的硅处理器和平台，而将量子处理器作为运算加速器或协处理器。整套系统由D-Wave 设计，而量子处理器芯片由美国宇航局喷气推进实验室下的微型设备实验室制造。

- 2007 年11 月，D-Wave 宣布研制成功28 qubit 量子计算机系统。2008 年5 月，D-Wave 宣布研制成功48 qubit 量子计算机系统。按照D-Wave 发布的路线图，到2008 年底，将达到1024 qubit（或512 qubit）。但这个目标后来没有实现。
- 2008年12 月19 日，D-Wave 宣布研制成功128 qubit 量子处理器
- D-Wave 还制定了一个量子计算机发展规划：量子计算软件系统，提供SQL、Prolog、Lisp等程序界面，将通过一个安全的互联网连接提供量子计算机的对外访问和使用，

- 目前针对密码破译的量子算法有两种：
- 一是由贝尔实验室的Grover 在1996 年发明的Grover 算法。这是一种针对所有密码（包括对称密码）的通用的搜索破译算法，其计算复杂度相当于把密钥长度减少到原来的一半。
- 从破译的角度，虽然这种算法使现有的计算能力提高了数亿倍，但对于目前使用的绝大多数对称密码和公钥密码来说还没有受到致命威胁。
- 二是由贝尔实验室的Shor 在1994 年发明的Shor 算法。这是一种专用的搜索破译算法，其扩展算法能以多项式时间攻破所有的能够转换成广义离散傅立叶变换的公钥密码——包括目前广泛使用的RSA、DH 和ECC。
- 由于量子并行运算的内在机制，即使不断增加这类密码的密钥长度，也只不过给破译工作增加了很小的代价。对于椭圆曲线离散对数问题，Proos 和Zalka 指出在N qubit 的量子计算机上可以容易地求解k 比特的椭圆曲线离散对数问题，例如利用1448qubit 量子计算机可以破译256 位的椭圆曲线密码。但Shor算法不能用来破译其它类型的公钥密码。

- “抗量子计算的公钥密码”是信息安全领域的下一个技术制高点，
- 公钥密码应对量子计算机的挑战，有三种基本对策：
- （1）用量子密码替代公钥密码。其优点：安全性能在理论上保证绝对不可能用数学方法来破译。问题在于：建立量
- 子密码系统需要昂贵的量子信道，量子密码也不具备数字签名的功能，如何建立网络信任体系值得探讨。

- （2）用对称密码的签名取代公钥密码的签名，例如用Hash 函数实现Merkle 签名方案。其优点：其安全性等价求解单向函数的困难性，大大高于目前所有的公钥密码体制；
- 与量子密码相比实现的代价很低。问题在于，这种签名采用一次一密体制，即产生一次签名就要用掉一个密钥，难以在开放环境下大量使用。
- （3）采用不能转换成离散傅立叶变换的数学难题来建立公钥密码体制。其优点：功能与现有公钥密码一致，能满足开放性的使用环境，成本低，具备工程实现的可行性。问题在于：其安全性不如前两种那么强，不排除将来有可能会发明出能破译这些密码的新的量子算法。
- 在目前条件下，只有第三种对策是现实可行的。

- 一旦量子计算机来了，用什么公钥密码来替代正在大量使用的RSA 和ECC？当前国外主要有以下几类互相竞争的技术解决方案：
- 一是NTRU 公钥密码体制。由美国数学家Hoffstein、Pipher 和Silverman 在1996 年发明的NTRU (Number Theory Research Unit)，基于在一个大维数的格中寻找一个很短向量的数学难题。这是目前工程指标最成熟的抗量子计算公钥密码，其算法简洁、计算速度快、占用存储空间小。
- NTRU 公司有强大的资本和美国政府的支持，开发了一系列示范性产品（IC 卡、手机、3G、无线互联网、电子商务、可信计算等），

- 安全性还没有获得充分认同，其签名的安全性远低于加密的安全性，不适合用在网络信任体系；
- 其次是知识产权的障碍，NTRU 公司在多个国家注册了基础性专利
- 二是OTU2000 公钥密码体制。由日本NTT公司Okamoto 实验室推出的“量子公钥密码体制”，基于改进的背包问题。
- 缺点是产生密钥需要计算离散对数，即需要用量子计算机解决大量密钥的生成问题。
- NTT 已用软件模拟方法完成了概念验证。

- 三是McEliece 公钥密码体制。该方案的安全性基于纠错编码问题，由McEliece 在1978 年提出。该方案的安全性很强，
- 缺点是密钥量太大、计算效率较低，实现大规模工程应用的代价较大。
- 四是MQ 公钥密码体制。即多变元二次多项式公钥密码体制（Multivariate Quadratic Polynomials in Public Key Cryptosystem），基于有限域上的多变元二次多项式方程组的难解性。与RSA、DH、ECC 相比，MQ 的安全性很难被证明等价于一个已知的可简单表述的数学难题。
- MQ 缺陷是加密算法过于简单；只能签名、不能加密，其加密的安全性远低于签名的安全性。