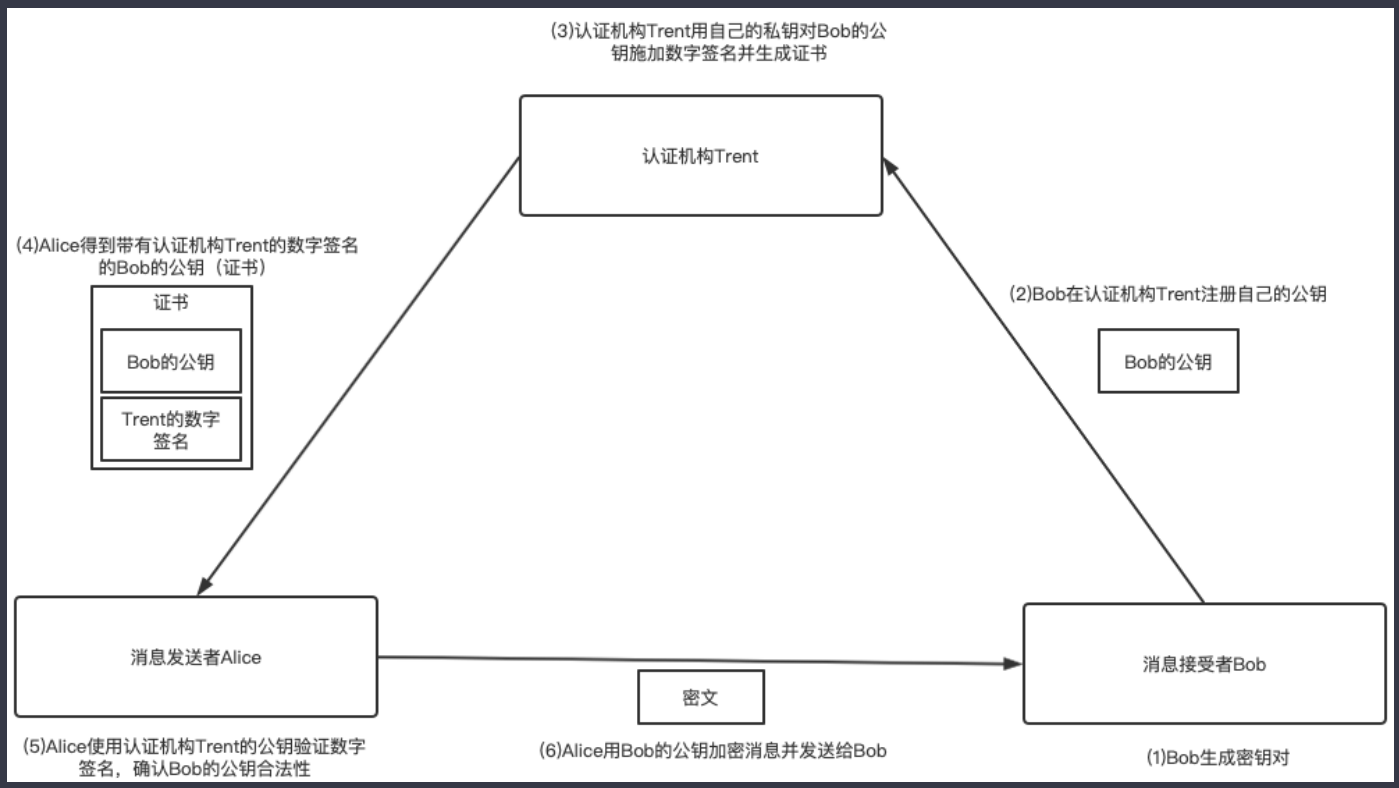
1.证书

> 公钥证书（Public-Key Certificate，PKC）其实和驾照很相似，里面记有姓名、组织、邮箱地址和个人信息，以及属于此人的公钥，并由认证机构（Certification Authority、Certifying Authority,CA）施加数字签名。是要看到公钥证书，我们就可以知道认证机构认定该公钥属于此人。公钥证书也简称为证书（certificate）。

1.1 证书的应用场景



> 但是注意一点，就是因为https是无状态的，一般来说Alice会通过公钥对对称加密的密钥进行加密然后发送给接受者Bob，后续消息加解密都是基于这样的对称加密的（更高效）。

1.2 证书规范和格式--x509

X509是一种非常通用的证书格式。所有的证书都符合ITU-T x509国际标准，因此理论上为一种应用创建的证书可以用于任何其他符合x.509标准的应用。x.509证书的结构是用ASN1（abstract syntax notation one）进行表述数据结构，并使用ASN.1语法进行编码。

x.509规范中一般推荐使用PEM（privacy enhanced mail）格式来存储证书相关文件。

- 证书文件的文件名后缀一般为.crt或.cer
- 对应私钥文件的文件名后缀一般为.key
- 证书请求文件的文件名后缀为.csr
- 有时候也统一用pem作为文件名后缀

证书里面一般的内容：

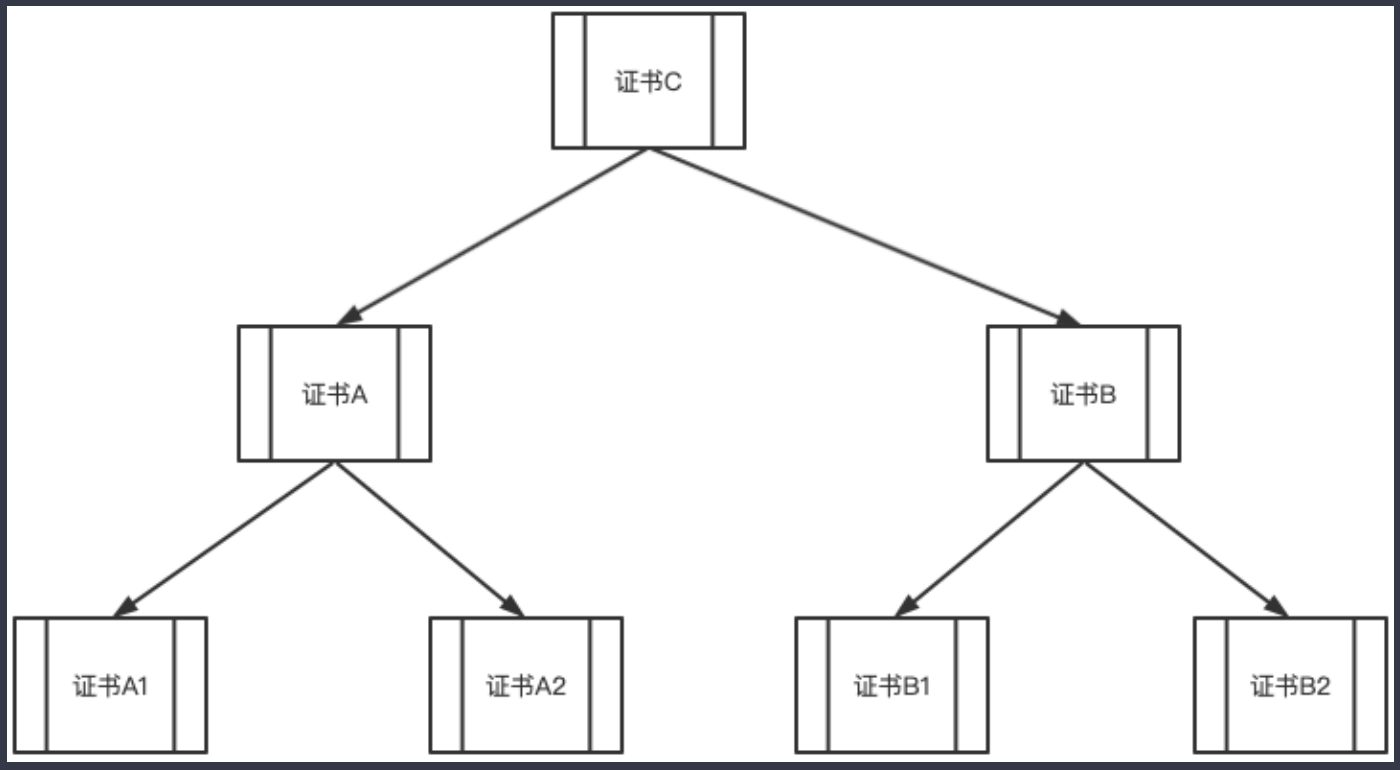
> 目前使用x509的v3版本规范（RFC5280）其中定义了如下证书信息域：
>
> - 版本号（version number）：规范的版本号

- 序列号（Serial Number）：由CA维护的为它所发的每个证书分配的唯一的序列，用来追中和撤销证书。只要拥有签发者的信息和序列号，就可以唯一来标识一个证书，最大不能超过20个字节；
- 签名算法（Signature Algorithm）：数字签名所采用的算法，如：
  - SHA256-with-RSA-Encryption
  - ccda-with-SHA256
- 颁发者（Issuer）：发证书单位的标识信息，
- 有效期（Validity）：证书的有效期，包括起止时间。
- 主题（Subject）：证书拥有者的标识信息。如“C=CN，ST=Beijing”.....

.....

1.3 证书的信任链

证书之间是可以有信任关系的。通过一个证书可以证明另一个证书也是真实可信的。实际上，证书之间的信任关系是可以嵌套的。比如，C信任A1，A1信任A2，A2信任A3....这个叫做证书的信任链，只要你信任链上头一个一个证书，那么后续证书，都被视为可信任。



处于最顶上的树根位置的那个证书，就是“根证书”。除了根证书，其他证书都要依靠上一级的证书，来证明自己。那谁来证明根证书的可靠性呢？实际上，根证书是靠自己证明自己是可靠的（即根证书不需要被证明）。

1.4 CA证书

CA证书是由Certification Authoriy机构发布的数字证书。要对CA证书完全理解其作用，首先需要理解SSL。SSL（security socket layers，安全套接层）是为网络通信提供安全及数据完整性的一种安全协议。SSL3.0版本以后有被称为TLS。SSL位于TCP与个应用层之间。是操作系统向外提供的API。SSL如何保证网络通信的安全和数据的完整性呢？就是采用了两种手段：身份认证和数据加密。首先的身份认证就是需要用到CA证书了。

- 证书的获取和身份认证

  客户端与服务端需要经过一个握手过程才能完成身份认证，建立一个安全的连接。握手的过程如下：

  - 客户端访问服务器（比如：https://www.12306/cm），发送ssl版本、客户端支持的加密算法等消息。
  - 服务器向客户端发送ssl版本、加密算法、证书等消息；
  - 客户端收到消息后，判断证书是否可信，若可信，则继续通信，发送消息；客户端生成一个随机数，从证书中首获取服务器端的公钥，对随机数进行加密。随后信息都将使用双方协定的加密方法和密钥发送，客户端的握手结束；
  - 服务器端对数据解密得到随机数，使用协商好的加密算法和密钥进行通信；

1.5 公钥基础设置（PKI）

仅仅指定证书的规范还不足以支持公钥的实际运用，我们还需要很多的其他规范，例如证书应该由谁来颁发，私钥泄露时应该如何作废证书，计算机之间的数据交换采用什么样的格式

1.5.1 什么是公钥基础设施

公钥基础设施（Public-Key Infrasture）是为了能够更有效地运用公钥而指定的一系列的规范和规格的总称。公钥基础设施一般根据器英语缩写为PKI。

PKI只是一个总称，而非指某一个单独的规范和规格。例如，RSA公司所制定的PKCS系列规范也是PKI的一种。而互联网规格RFC中也有很多与PKI相关的文档。此外，x509这样的规范也是PKI的一种，在开发PKI程序时所使用的由各个公司所编写的API和规格设计数也可以算是API的相关规格。

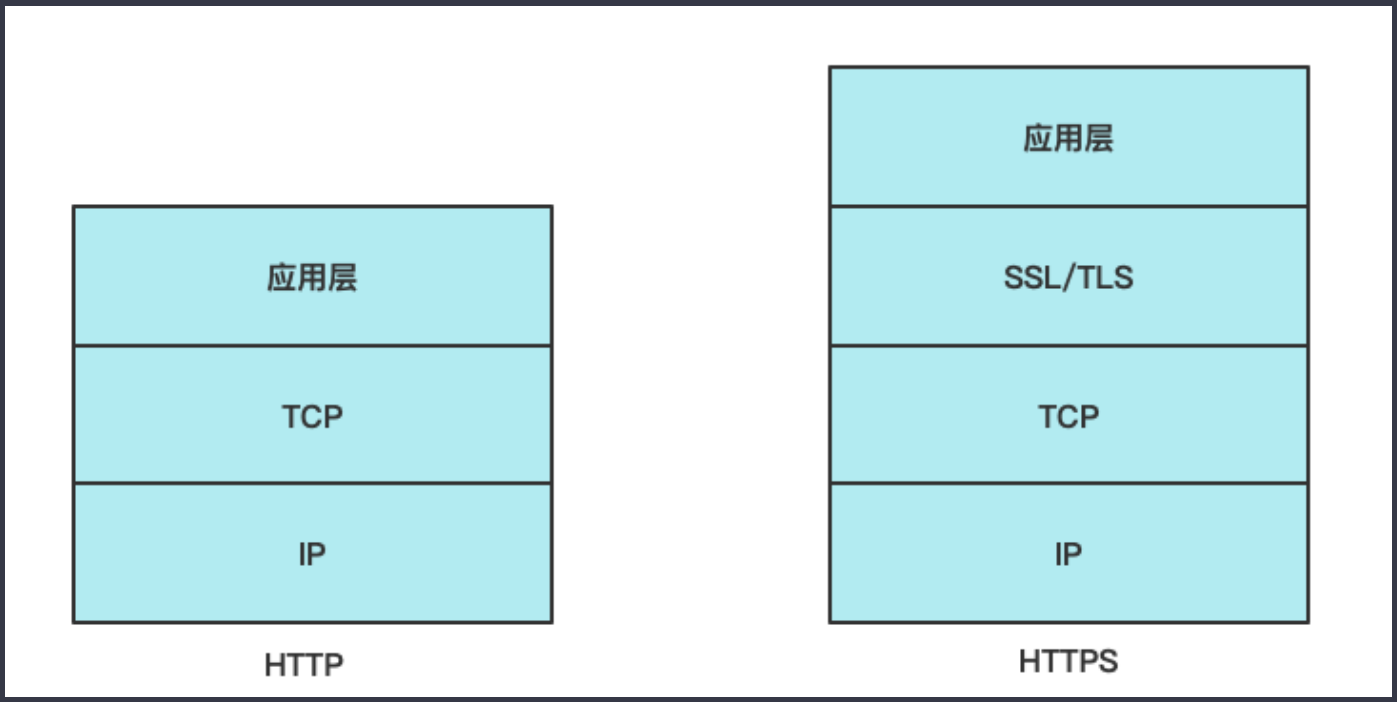
因此，根据具体所采用的规格，PKI也会有很多变种，这也是很多人难以理解PKI的原因之一。

1.5.2 PKI的组成要素

PKI的组成要素主要有以下三个：

- 用户——使用PKI的人
  - 申请证书的人——> web服务器
    - 申请证书
      - 生成密钥对，或者委托CA生成
      - 将公钥发送给CA
      - CA使用自己的私钥对得到的公钥签名
      - 将证书发送给用户
    - 发送证书
      - 当客户端访问服务器的时候发送证书给客户端
    - 注销证书
      - 当发现私钥泄露后会注销证书
  - 使用证书的人——>客户端
    - 接受证书
    - 验证对方的身份信息
- 认证机构——颁发证书的人
  - 可以生成密钥对
  - 对公钥进行签名
  - 注销证书
- 仓库——保存证书的数据库

2.SSL/TLS



> **SSL**：（Secure Soket Layer，安全套接字层），为Netscape所研发，用以保证在Internet上数据传输之间的安全，利用数据加密技术（Encryption），可确保数据在网络上的传输不会被截取。SSL协议位于TCP/IP协议与各种应用层之间，为数据通讯提供安全支持。SSL协议分为两层：SSL记录协议（SSL Record protocol），它建立在可靠的传输协议（如TCP）之上，为高层协议提供数据封装，压缩，加密等基本功能的支持。SSL握手（SSL Handshake Protocol）：它建立在SSL记录协议之上，用于在实际数据传输开始前，通讯双方进行身份认证，协商加密算法，交换加密密钥等。
>
> **TLS**：（Transport Layer Security，传输层安全协议），用于两个应用程序之间提供保密性和数据完整性。TLS 1.0是IETF制定的一种新的协议，它建立在SSL3.0协议的规范之上，是SSL 3.0的后续版本，可以理解为SSL 3.1，它是写入RFC的，该协议仅仅由两层组成的：TLS记录协议和TLS握手协议。较低的层为TLS记录协议，位于某个可靠的传输协议（例如TCP）上面。

SSL/TSL协议提供的主要服务有

- 认证用户和服务器，确保数据发送到正确的客户机和服务器；
- 加密数据以防止数据中途被窃取；
- 维护数据的完整性，确保数据在传输过程中不被改变；

3.HTTPS通信过程

3.1 HTTP通信的缺点

- HTTP不验证身份，导致身份可能被伪装；
- 明文传输，可能导致数据泄露；
- 无法验证数据的完整性，数据可能被篡改；

> HTTP不验证身份，导致身份可能被伪装可是使用证书的方式来解决明文传输的问题可以使用对称加密或者非对称加密的方式对数据进行加解密；而数据被篡改的问题可以是用数字签名的方式解决

3.2 HTTPS通信流程

- **第一步**：客户端向服务端发起建立HTTPS的请求
  - 客户端会生成一个随机数R1并发送给服务端；
  - 告诉服务端自己支持哪些加密算法；
- **第二步**：服务器向客户端发送数字证书
  - 服务端生成一个随机数R2发送客户端；
  - 从客户端所支持的加密算法中选取一个双方都支持的算法用于后续的会话加密算法；
  - 服务端把证书、R2、会话密钥算法一同发给客户端；
- **第三步**：客户端验证数字证书
  - 验证证书的可靠性。先用CA的公钥解密被加密过的证书，能解密则说明证书没有问题，然后通过证书里提供的摘要算法对数据进行摘要，然后通过自己生成的摘要与服务端发送的摘要对比。
  - 验证证书的合法性，包括证书是否吊销、是否到期、域名是否匹配，通过后则进行后续流程
  - 获得证书的公钥，会话加密算法、随机数R2
  - 生成一个随机数R3
  - 根据会话密钥算法使用R1、R2、R3生成会话密钥
  - 用服务端证书的公钥加密随机数R3并发送给服务端；
- **第四步**：服务器得到会话密钥
  - 服务端获取到客户端发送过来的随机数R3，并基于之前协商好的算法使用R1、R2、R3生成会话密钥

- **第五步**：客户端与服务端进行加密会话
  - 客户端发送加密数据给服务端，服务接受加密数据，并使用密钥对加密数据进行解密，处理之后在使用密钥对响应数据进行加密，然后发送给服务端
  - 客户端接受到服务端发送过来的加密数据，使用密钥进行解密获取到响应数据。

4. 自签名证书

- 先创建一个目录，进入该目录
- 启动openssl

```
openssl
```

- 使用openssl工具生成一个RSA私钥，注意：生成私钥需要提供密码

  Server.key

```
genrsa -des3 -out server.key 2048
```

- 生成CSR（证书签名请求）

```
req -new -key server.key -out server.csr
```

- 删除私钥中的密码[可选]

```
rsa -in server.key -out server_no_password.key
```

- 生成自签名证书

```
x509 -req -days 365 -in server.csr -signkey
server_no_password.key -out server.crt
```

> 如果现在需要部署，则`server_no_password.key`以3des这种加密方式保存了公钥和私钥。以及证书`server.crt`即可。

5.HTTPS优缺点

5.1 HTTPS优点

- 使用HTTPS协议认证用户和服务器，确保数据发送到正确的客户端和服务器；
- HTTPS是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，要比http协议安全，可防止数据在传输过程中不被窃取和改变，确保数据的完整性；
- HTTPS是现行架构下最安全的解决方案，虽然不是绝对安全，但它大幅度增加了中间人攻击的成本；

5.2 HTTPS缺点

- HTTPS协议握手阶段比较费时，会使页面的加载时间延长50%，增加10%～20%的耗电；
- HTTPS连接缓存不如HTTP高效，会增加数据开销和功耗，甚至已由的安全措施也会因此而受到影响；
- SSL/TLS证书需要钱，功能越强大的证书费用越高，个人网站和小说网站没有必要一般也不会使用；
- SSL/TLS证书通常需要绑定IP，不能在同一IP绑定多个域名，IPv4资源不可能支撑这个消耗；
- HTTPS协议的加密范围也比较有限，在黑客攻击、拒绝服务攻击、服务器劫持等方面几乎起不到什么作用。更重要的是SSL证书的信用链体系并不安全，特别是某些国家可以控制CA根证书的情况下，中间人的攻击仍然是可行的。