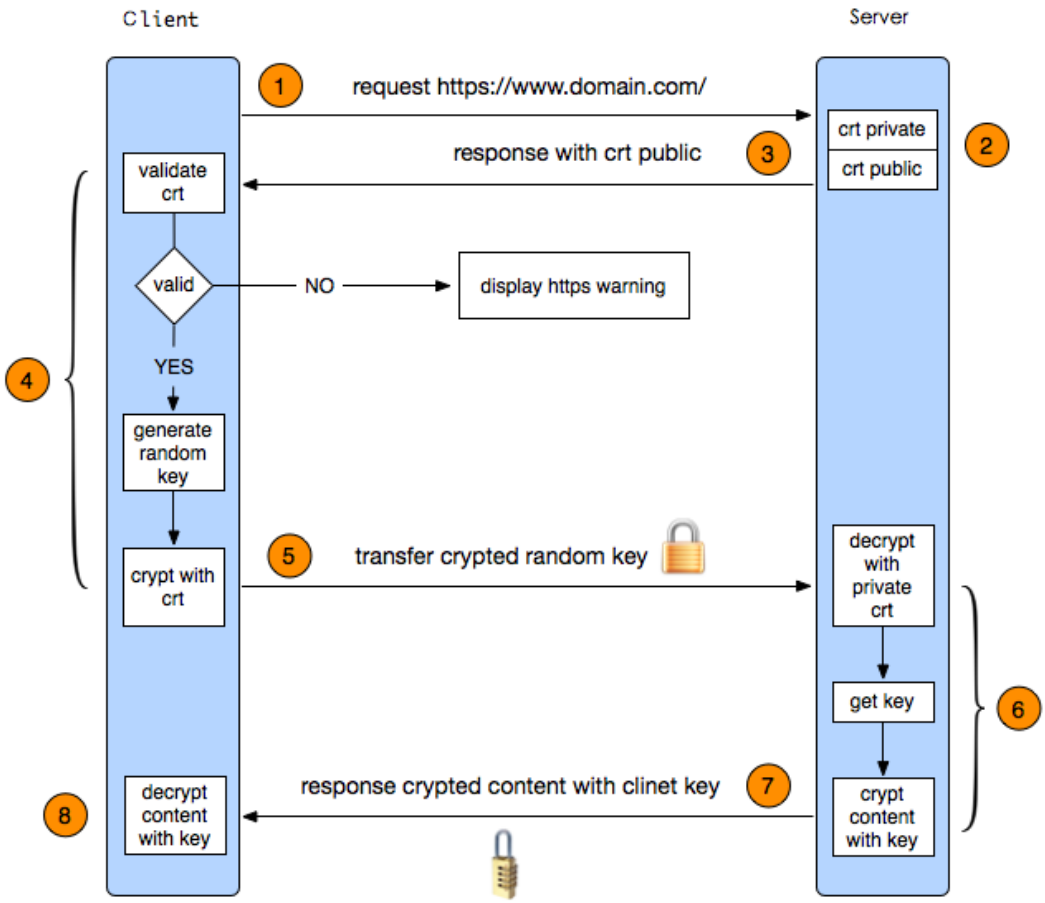
https 原理：证书传递、验证和数据加密、解密过程解析

我们都知道 HTTPS 能够加密信息，以免敏感信息被第三方获取。所以很多银行网站或电子邮箱等等安全级别较高的服务都会采用 HTTPS 协议。

HTTPS 简介

HTTPS 其实是有两部分组成：HTTP + SSL / TLS，也就是在 HTTP 上又加了一层处理加密信息的模块。服务端和客户端的信息传输都会通过 TLS 进行加密，所以传输的数据都是加密后的数据。具体是如何进行加密，解密，验证的，且看下图。



1. 客户端发起 **HTTPS** 请求

这个没什么好说的，就是用户在浏览器里输入一个 https 网址，然后连接到 server 的 443 端口。

2. 服务端的配置

采用 HTTPS 协议的服务器必须要有一套数字证书，可以自己制作，也可以向组织申请。区别就是自己颁发的证书需要客户端验证通过，才可以继续访问，而使用受信任的公司申请的证书则不会弹出提示页面(startssl 就是个不错的选择，有 1 年的免费服务)。这套证书其实就是一对公钥和私钥。如果对公钥和私钥不太理解，可以想象成一把钥匙和一个锁头，只是全世界只有你一个人有这把钥匙，你可以把锁头给别人，别人可以用这个锁把重要的东西锁起来，然后发给你 ,因为只有你一个人有这把钥匙 ,所以只有你才能看到被这把锁锁起来的东西。

3. 传送证书

这个证书其实就是公钥，只是包含了很多信息，如证书的颁发机构，过期时间等等。

4. 客户端解析证书

这部分工作是有客户端的 TLS 来完成的，首先会验证公钥是否有效，比如颁发机构，过期时间等等，如果发现异常，则会弹出一个警告框，提示证书存在问题。如果证书没有问题，那么就生成一个随即值。然后用证书对该随机值进行加密。就好像上面说的，把随机值用锁头锁起来，这样除非有钥匙，不然看不到被锁住的内容。

5. 传送加密信息

这部分传送的是用证书加密后的随机值，目的就是让服务端得到这个随机值，以后客户端和服务端的通信就可以通过这个随机值来进行加密解密了。

6. 服务段解密信息

服务端用私钥解密后，得到了客户端传过来的随机值(私钥)，然后把内容通过该值进行对称加密。所谓对称加密就是，将信息和私钥通过某种算法混合在一起，这样除非知道私钥，不然无法获取内容，而正好客户端和服务端都知道这个私钥，所以只要加密算法够彪悍，私钥够复杂，数据就够安全。

7. 传输加密后的信息

这部分信息是服务段用私钥加密后的信息，可以在客户端被还原。

8. 客户端解密信息

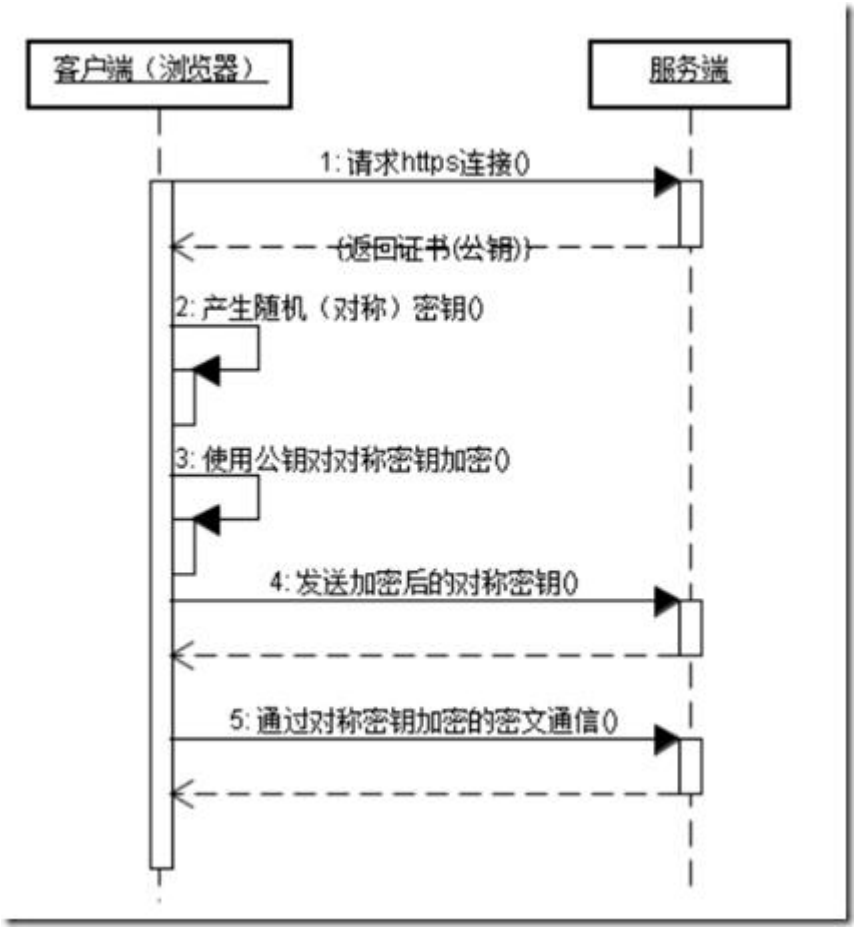
客户端用之前生成的私钥解密服务段传过来的信息，于是获取了解密后的内容。整个过程第三方即使监听到了数据，也束手无策。

SSL 的位置

SSL 介于应用层和 TCP 层之间。应用层数据不再直接传递给传输层，而是传递给 SSL 层，SSL 层对从应用层收到的数据进行加密，并增加自己的 SSL 头。如下图所示：

图 17-1　在互联网模型中 SSL 和 TLS 的位置

RSA 性能是非常低的，原因在于寻找大素数、大数计算、数据分割需要耗费很多的 CPU 周期，所以一般的 HTTPS 连接只在第一次握手时使用非对称加密，通过握手交换对称加密密钥，在之后的通信走对称加密。握手过程如下图所示：



HTTPS 在传输数据之前需要客户端（浏览器）与服务端（网站）之间进行一次握手，在握手过程中将确立双方加密传输数据的密码信息。TLS/SSL 协议不仅仅是一套加密传输的协议，更是一件经过艺术家精心设计的艺术品，TLS/SSL 中使用了非对称加密，对称加密以及 HASH 算法。握手过程的具体描述如下：

1.浏览器将自己支持的一套加密规则发送给网站。

2.网站从中选出一组加密算法与 HASH 算法，并将自己的身份信息以证书的形式发回给浏览器。证书里面包含了网站地址，加密公钥，以及证书的颁发机构等信息。

3.浏览器获得网站证书之后浏览器要做以下工作：

a) 验证证书的合法性（颁发证书的机构是否合法，证书中包含的网站地址是否与正在访问的地址一致等），如果证书受信任，则浏览器栏里面会显示一个小锁头，否则会给出证书不受信的提示。

b) 如果证书受信任，或者是用户接受了不受信的证书，浏览器会生成一串随机数的密码，并用证书中提供的公钥加密。

c) 使用约定好的 HASH 算法计算握手消息，并使用生成的随机数对消息进行加密，最后将之前生成的所有信息发送给网站。

4.网站接收浏览器发来的数据之后要做以下的操作：

a) 使用自己的私钥将信息解密取出密码，使用密码解密浏览器发来的握手消息，并验证 HASH 是否与浏览器发来的一致。

b) 使用密码加密一段握手消息，发送给浏览器。

5.浏览器解密并计算握手消息的 HASH，如果与服务端发来的 HASH 一致，

此时握手过程结束，之后所有的通信数据将由之前浏览器生成的随机密码并利用对称加密算法进行加密。

浏览器与网站互相发送加密的握手消息并验证，目的是为了保证双方都获得了一致的密码，并且可以正常的加密解密数据，为后续真正数据的传输做一次测试。另外，HTTPS 一般使用的加密与 HASH 算法如下：

1、非对称加密算法：RSA，DSA/DSS

2、对称加密算法：AES，RC4，3DES

3、HASH 算法：MD5，SHA1，SHA256