[1、 浏览器访问https网站的握手过程？ 2](#_Toc6641)

[1. 客户端发起HTTPS请求 4](#_Toc19863)

[2. 服务端的配置 4](#_Toc21728)

[3. 传送证书 4](#_Toc32519)

[4. 客户端解析证书 4](#_Toc3197)

[5. 传送加密信息 4](#_Toc7311)

[6. 服务段解密信息 4](#_Toc28268)

[7. 传输加密后的信息 5](#_Toc12362)

[8. 客户端解密信息 5](#_Toc19265)

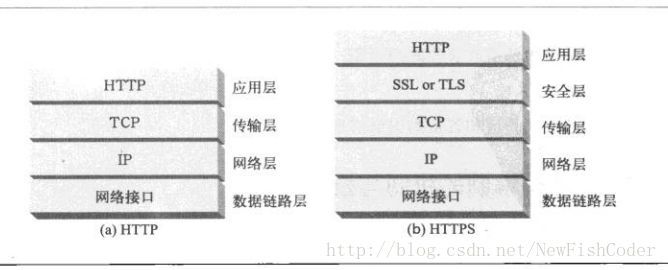
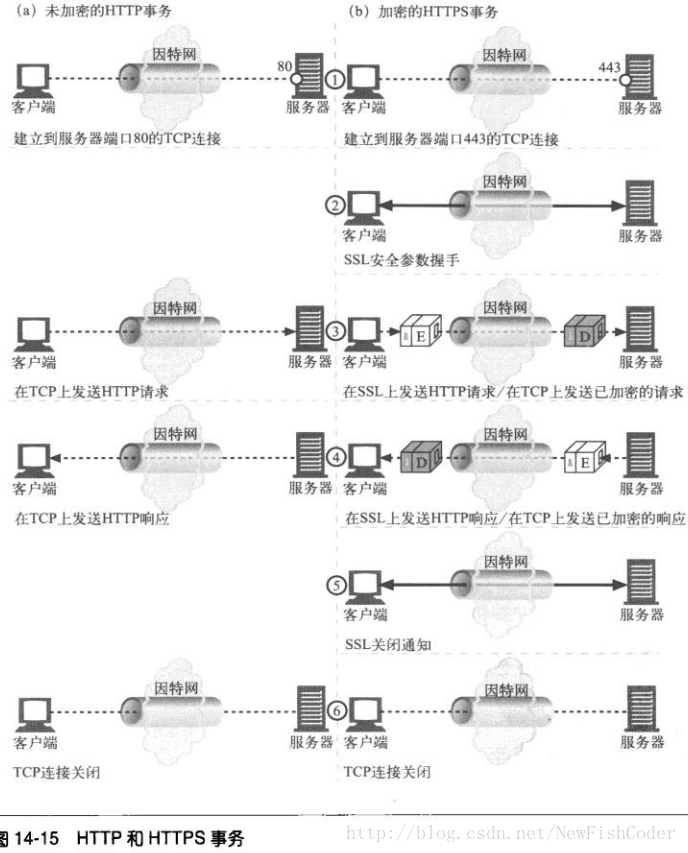
[2、 浏览器端的证书（客户端证书）是如何自动加载的？ 5](#_Toc984)

[3、 服务器需要做什么配置？ 6](#_Toc31755)

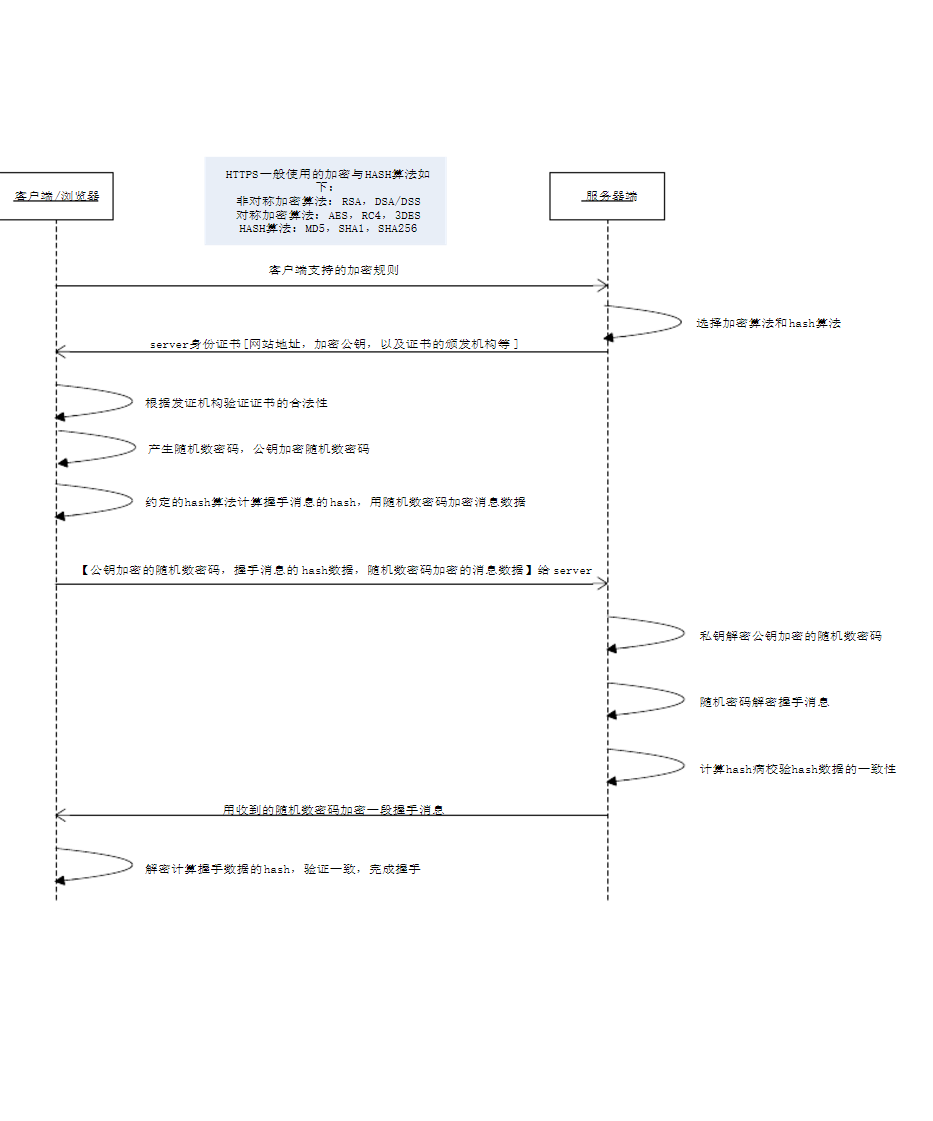
[附录： 6](#_Toc12854)

# 浏览器访问https网站的握手过程？

首先是http 和 https 的对比



详细过程详见下图



## 1. 客户端发起HTTPS请求

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

web通信中的SSL加密的公钥证书(受信任的第三方证书颁发机构签名颁发)常见的如

VeriSign

Thawte

GlobalSign

Symantec

## 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。

## 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

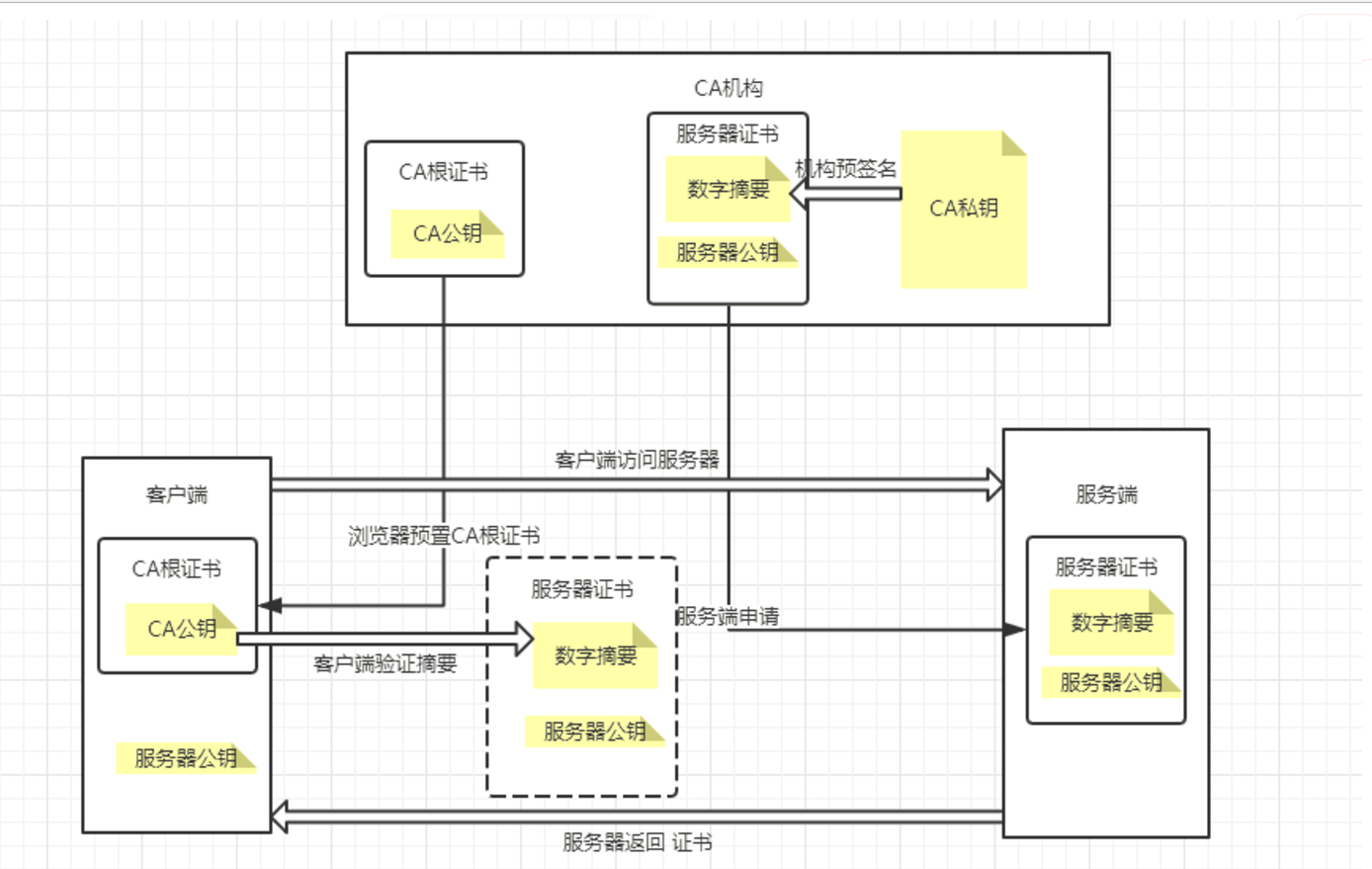
## 8. 客户端解密信息

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。

大概过程如下：

握手——协商加密算法——获得公钥证书——验证公钥证书——交换会话密钥——加密信息传输

# 浏览器端的证书（客户端证书）是如何自动加载的？



可以参考 ：

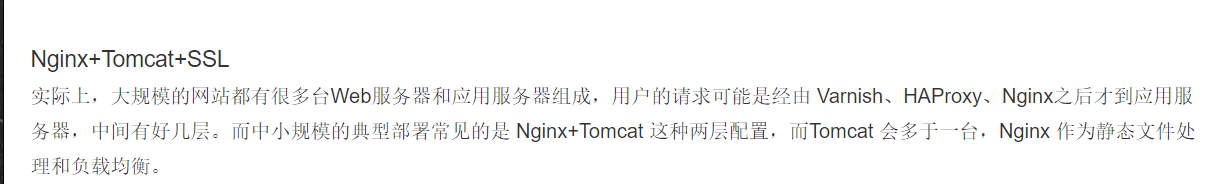
<http://www.jianshu.com/p/0d59d2216c64>

<http://blog.csdn.net/andy1219111/article/details/22716315>

# 服务器需要做什么配置？

<http://blog.csdn.net/andy1219111/article/details/22716315>

跟我们的方法很相像



# 附录：

