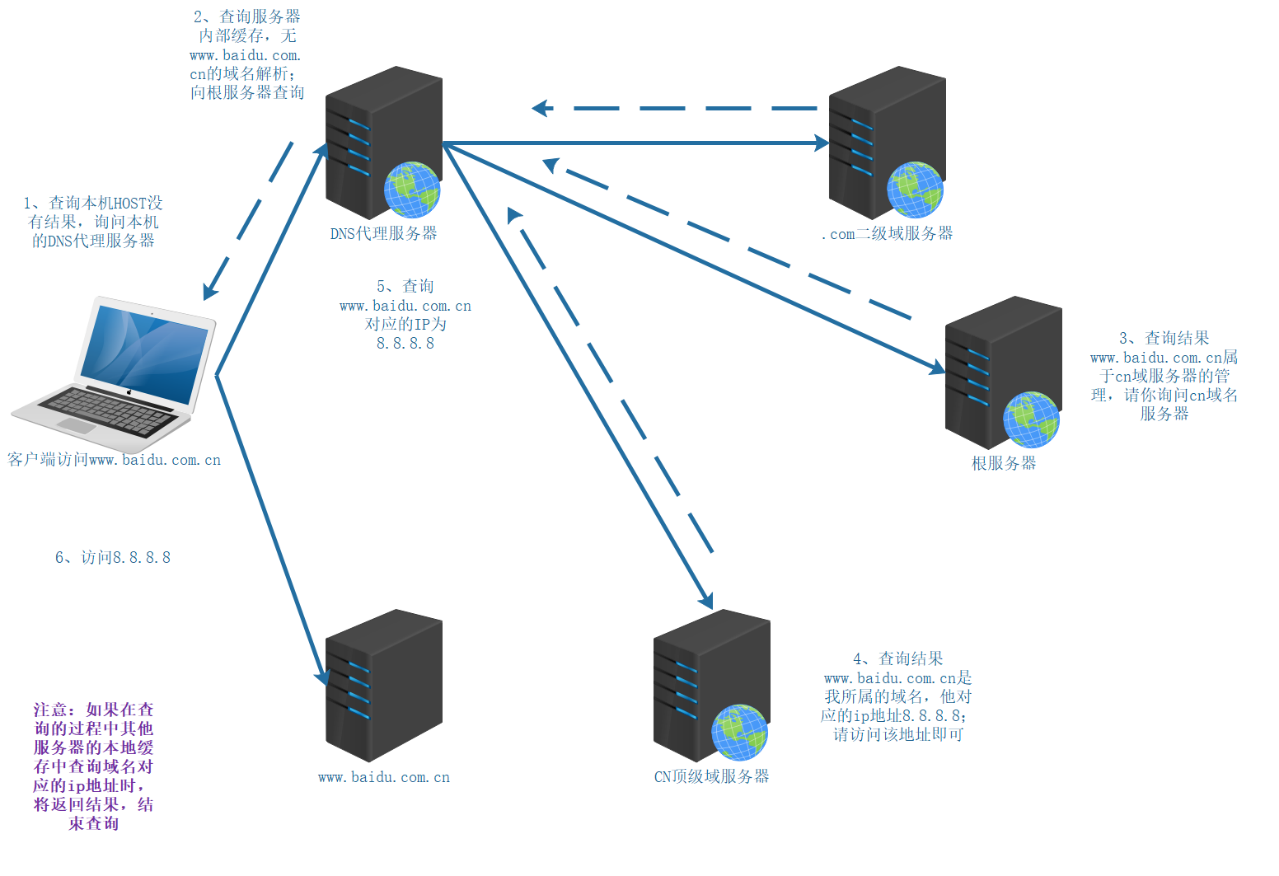
**1、DNS工作原理**

**分级查询（递归/迭代）**



请求主机→本地域名服务器→根域名服务器→顶级域名服务器→权威域名服务器

**2、OSI与TCP/IP各层的结构与功能，都有哪些协议**



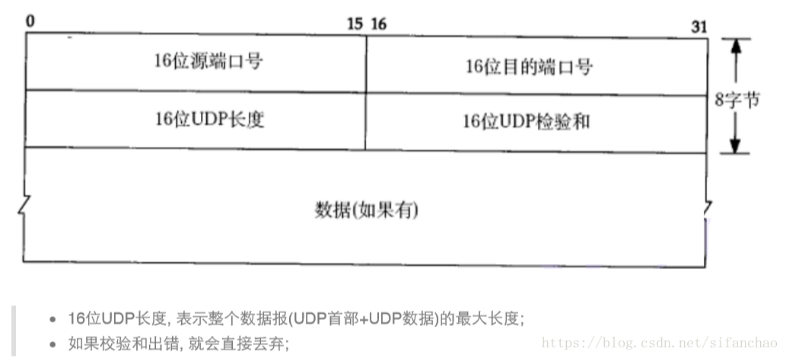
**3、TCP与UDP的区别**

UDP协议和TCP协议都是传输层协议。

1、TCP提供的是**面向连接**，**可靠**的字节流服务。即客户和服务器交换数据前，必须现在双方之间建立一个TCP连接，之后才能传输数据。并且提供超时重发，丢弃重复数据，检验数据，流量控制等功能，保证数据能从一端传到另一端。

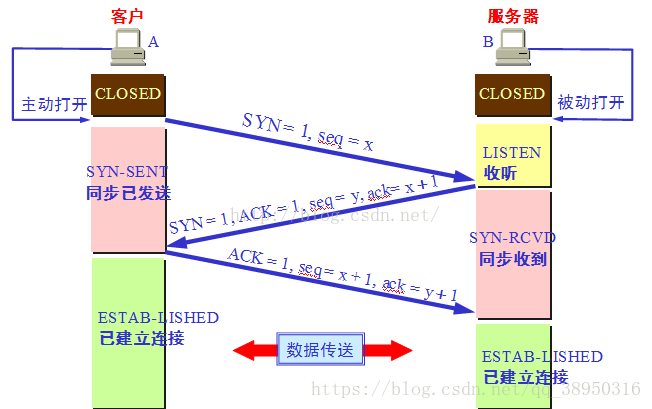
2、UDP是一个简单的面向数据报的运输层协议。它不提供可靠性，只是把应用程序传给IP层的数据报发送出去，但是不能保证它们能到达目的地。由于UDP在传输数据报前不用再客户和服务器之间建立一个连接，且没有超时重发等机制，所以传输速度很快。

**4、TCP、UDP报文结构**



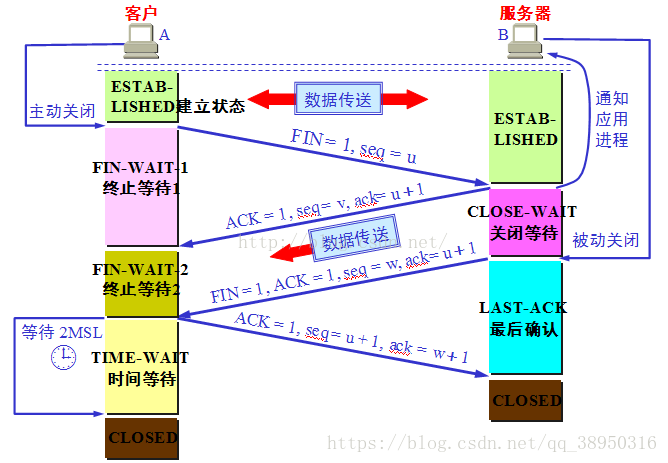


**5、TCP的三次握手与四次挥手，TIMEWAIT的作用**

**

为什么需要三次握手呢？为了防止已失效的连接请求报文段突然又传送到了服务端，因而产生错误。

比如：client发出的第一个连接请求报文段并没有丢失，而是在某个网络结点长时间的滞留了，以致延误到连接释放以后的某个时间才到达server。本来这是一个早已失效的报文段，但是server收到此失效的连接请求报文段后，就误认为是client再次发出的一个新的连接请求，于是就向client发出确认报文段，同意建立连接。假设不采用“三次握手”，那么只要server发出确认，新的连接就建立了，由于client并没有发出建立连接的请求，因此不会理睬server的确认，也不会向server发送数据，但server却以为新的运输连接已经建立，并一直等待client发来数据。所以没有采用“三次握手”，这种情况下server的很多资源就白白浪费掉了。



为什么需要四次挥手呢？TCP是全双工模式，当client发出FIN报文段时，只是表示client已经没有数据要发送了，client告诉server，它的数据已经全部发送完毕了；但是，这个时候client还是可以接受来server的数据；当server返回ACK报文段时，表示它已经知道client没有数据发送了，但是server还是可以发送数据到client的；当server也发送了FIN报文段时，这个时候就表示server也没有数据要发送了，就会告诉client，我也没有数据要发送了，如果收到client确认报文段，之后彼此就会愉快的中断这次TCP连接。

TCP协议规定,主动关闭连接的一方要处于**TIME\_ WAIT**状态,等待两个**MSL(最大报文生存周期)**的时间后才能回到CLOSED状态。TIME\_WAIT持续存在2MSL的话，就能保证在两个传输方向上的尚未被接收或迟到的报文段都已经消失，同时如果最后一个ACK丢失，TIME\_ WAIT状态可以重传最后的确认报文。

**6、TCP如何实现可靠传输**

1、应用数据被分割成 TCP 认为最适合发送的数据块。

2、TCP 给发送的每一个包进行编号，接收方对数据包进行排序，把有序数据传送给应用层。

3、校验和： TCP将保持它首部和数据的检验和。这是一个端到端的检验和，目的是检测数据在传输过程中的任何变化。如果收到段的检验和有差错，TCP 将丢弃这个报文段和不确认收到此报文段。

4、TCP 的接收端会丢弃重复的数据。

5、流量控制： TCP连接的每一方都有固定大小的缓冲空间，TCP的接收端只允许发送端发送接收端缓冲区能接纳的数据。当接收方来不及处理发送方的数据，能提示发送方降低发送的速率，防止包丢失。TCP 使用的流量控制协议是可变大小的**滑动窗口协议**。

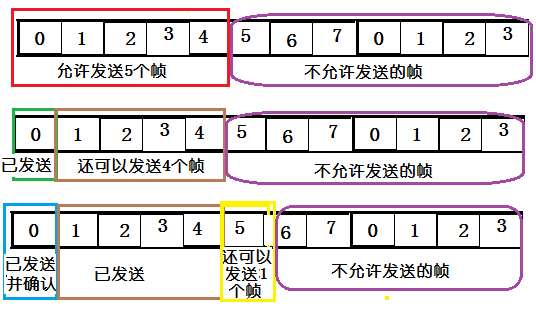
6、拥塞控制：当网络拥塞时，减少数据的发送。

7、停止等待协议：也是为了实现可靠传输的，它的基本原理就是每发完一个分组就停止发送，等待对方确认。在收到确认后再发下一个分组。

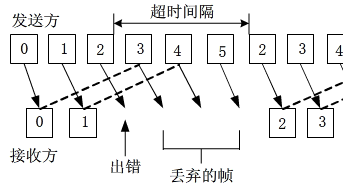
检验和、序号、定时器、肯定和否定确认 回退N步、选择重传

**7、TCP实现流量控制-滑动窗口/回退N针协议**

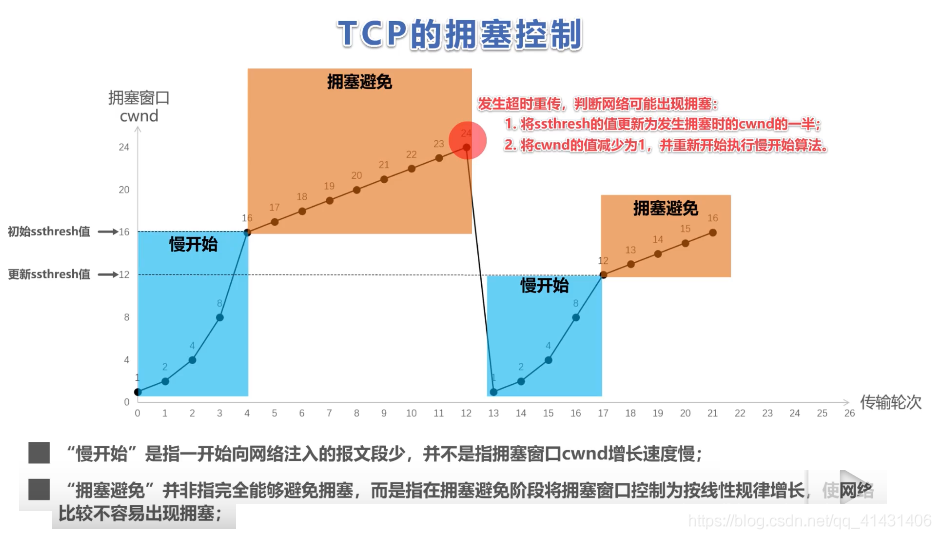
1、发送端和接收端分别设定发送窗口和接收窗口。   
2、三次握手的时候，客户端把自己的缓冲区大小也就是窗口大小发送给服务器，服务器回应是也将窗口大小发送给客户端，服务器客户端都知道了彼此的窗口大小。   
3、比如主机A的发送窗口大小为5，主机A可以向主机B发送5个单元，如果B缓冲区满了，A就要等待B确认才能继续发送数据。   
4、如果缓冲区中有1个报文被进程读取，主机B就会回复ACK给主机A，接收窗口向前滑动，报文中窗口大小为1，就说明A还可以发送1个单元的数据，发送窗口向前滑动，之后等待主机B的确认报文。只有接收窗口向前滑动并发送了确认时，发送窗口才能向前滑动。



在发送完一个帧后，不用停下来等待确认，而是可以连续发送多个数据帧。这样就减少了等待时间，整个通信的通吞吐量提高。 如果前一个帧在超时时间内未得到确认，就认为丢失或被破坏，需要重发出错帧及其后面的所有数据帧。这样有可能有把正确的数据帧重传一遍，降低了传送效率。



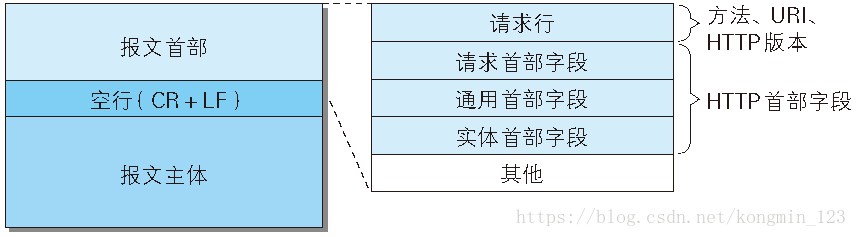
**8、TCP实现拥塞控制**



（收到三个冗余ACK时，进入拥塞避免状态）

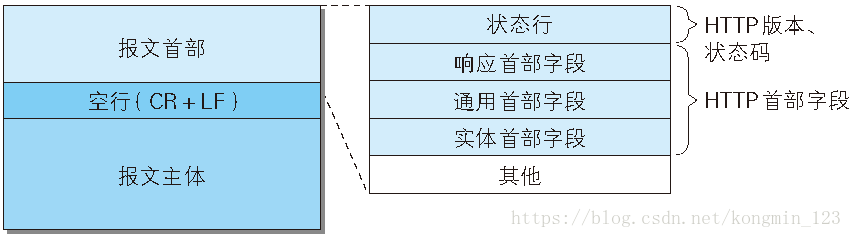
**9、Http的报文结构**

**请求报文：**





**响应报文：**





**10、Http的状态码含义**



**301 Moved Permanently：**永久性重定向，表示请求的资源被分配了新的URL，之后应使用更改的URL

**400 Bad Request：**表示请求报文中存在语法错误

**404 Not Found：**表示服务器上无法找到请求的资源

**500 Inter Server Error：**表示服务器在执行请求时发生了错误，也有可能是web应用存在的bug或某些临时的错误时

**503 Server Unavailable：**表示服务器暂时处于超负载或正在进行停机维护，无法处理请求

**11、Http request的几种类型**

1、OPTIONS：允许客户端查看服务器的性能。

2、HEAD：类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头。

3、GET：向特定的资源发出请求。

4、POST：向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的创建和/或已有资源的修改。

5、PUT：向指定资源位置上传其最新内容。

6、DELETE：请求服务器删除Request-URI所标识的资源。

7、TRACE：回显服务器收到的请求，主要用于测试或诊断。

8、CONNECT：HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

**12、Http1.0、1.1、2.0的区别**

**1、HTTP1.0和HTTP1.1的区别**

**1.1 长连接(keep-alive)**

       HTTP1.1支持长连接和请求的流水线处理，在一个TCP连接上可以传送多个HTTP请求和响应，减少了建立和关闭连接的消耗和延迟，在HTTP1.1中默认开启长连接keep-alive，一定程度上弥补了HTTP1.0每次请求都要创建连接的缺点。HTTP1.0需要使用keep-alive参数来告知服务器端要建立一个长连接。

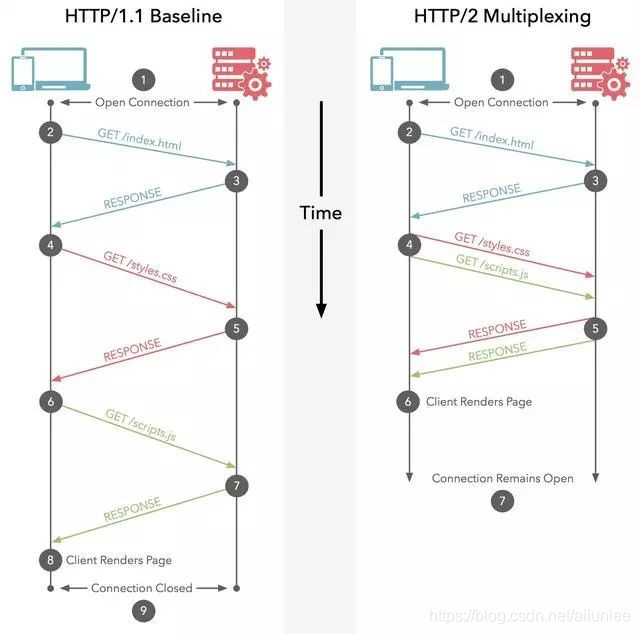
**1.2 节约带宽**

       HTTP1.0中存在一些浪费带宽的现象，例如客户端只是需要某个对象的一部分，而服务器却将整个对象送过来了，并且不支持断点续传功能。HTTP1.1支持只发送header信息（不带任何body信息），如果服务器认为客户端有权限请求服务器，则返回100，客户端接收到100才开始把请求body发送到服务器；如果返回401，客户端就可以不用发送请求body了节约了带宽。

**2、HTTP1.1和HTTP2.0的区别**

**2.1 多路复用**

       HTTP2.0使用了多路复用的技术，做到同一个连接并发处理多个请求，而且并发请求的数量比HTTP1.1大了好几个数量级。HTTP1.1也可以多建立几个TCP连接，来支持处理更多并发的请求，但是创建TCP连接本身也是有开销的。



**2.2 头部数据压缩**

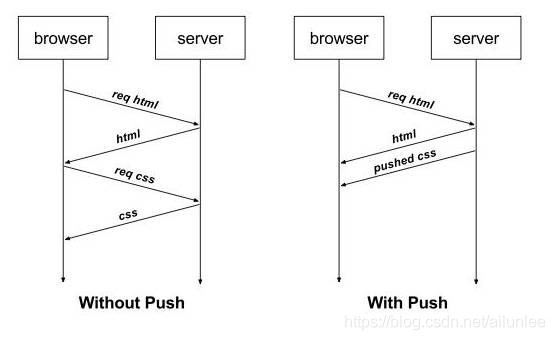
       在HTTP1.1中，HTTP请求和响应都是由状态行、请求/响应头部、消息主体三部分组成。一般而言，消息主体都会经过gzip压缩，或者本身传输的就是压缩过后的二进制文件，但状态行和头部却没有经过任何压缩，直接以纯文本传输。随着Web功能越来越复杂，每个页面产生的请求数也越来越多，导致消耗在头部的流量越来越多，尤其是每次都要传输UserAgent、Cookie这类不会频繁变动的内容，完全是一种浪费。

       HTTP1.1不支持header数据的压缩，HTTP2.0使用HPACK算法对header的数据进行压缩，这样数据体积小了，在网络上传输就会更快。

**2.3 服务器推送**

       服务端推送是一种在客户端请求之前发送数据的机制。网页使用了许多资源：HTML、样式表、脚本、图片等等。在HTTP1.1中这些资源每一个都必须明确地请求。这是一个很慢的过程。浏览器从获取HTML开始，然后在它解析和评估页面的时候，增量地获取更多的资源。因为服务器必须等待浏览器做每一个请求，网络经常是空闲的和未充分使用的。

       为了改善延迟，HTTP2.0引入了server push，它允许服务端推送资源给浏览器，在浏览器明确地请求之前，免得客户端再次创建连接发送请求到服务器端获取。这样客户端可以直接从本地加载这些资源，不用再通过网络。



**13、电脑上访问一个网页，整个过程是怎么样的**

1、浏览器查询 DNS，获取域名对应的IP地址

2、浏览器获得域名对应的IP地址以后，浏览器向服务器请求建立链接，发起三次握手

3、TCP/IP链接建立起来后，浏览器向服务器发送HTTP请求；

4、服务器接收到这个请求，并根据路径参数映射到特定的请求处理器进行处理，并将处理结果及相应的视图返回给浏览器；

5、浏览器解析并渲染视图，若遇到对js文件、css文件及图片等静态资源的引用，则重复上述步骤并向服务器请求这些资源；

6、浏览器根据其请求到的资源、数据渲染页面，最终向用户呈现一个完整的页面。

**14、路由器与交换机区别**

1、路由器实现了不同网络之间的数据转发，交换机实现了特定网络内的数据交换

2、交换机转发所依据的对象时：MAC地址。（物理地址）  
3、路由转发所依据的对象是：IP地址。（网络地址）

**\*\*15、HTTP、HTTPS**

HTTPS是一种通过计算机网络进行安全通信的传输协议，经由HTTP进行通信，利用**SSL/TLS**建立全信道，加密数据包。HTTPS使用的主要目的是提供对网站服务器的身份认证，同时保护交换数据的隐私与完整性。

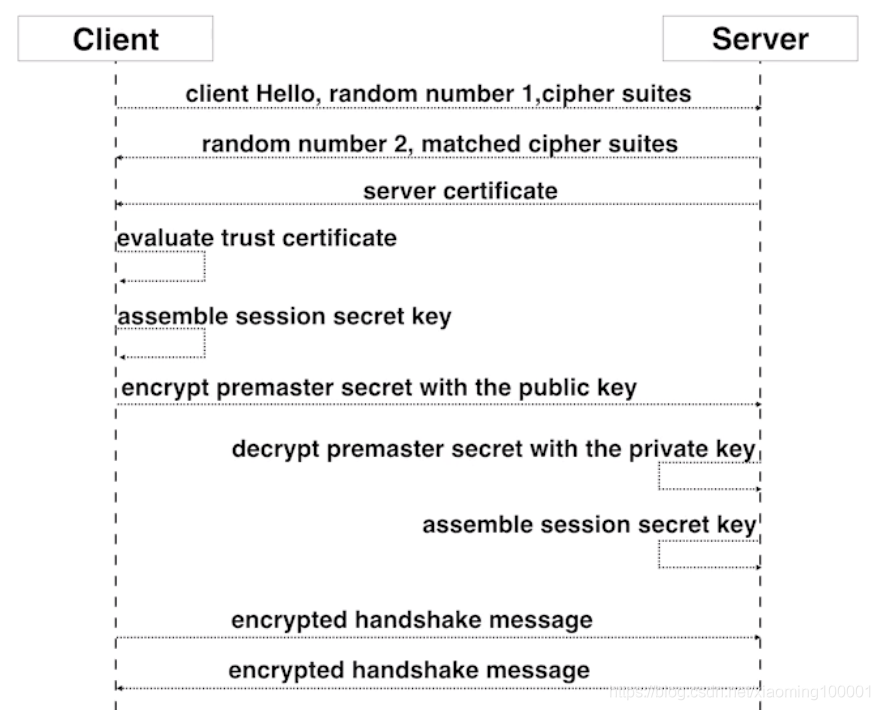
特点：

**内容加密：**采用混合加密技术，中间者无法直接查看明文内容

**验证身份：**通过证书认证客户端访问的是自己的服务器

**保护数据完整性：**防止传输的内容被中间人冒充或者篡改

SSL建立连接过程：



1、client向server发送请求https://baidu.com，然后连接到server的443端口，发送的信息主要是**随机值1**和客户端支持的**加密算法**。

2、server接收到信息之后给予client响应握手信息，包括**随机值2**和匹配好的**协商加密算法**，这个加密算法一定是client发送给server加密算法的子集。

3、随即server给client发送第二个响应报文是**数字证书**。这套证书其实就是一对公钥和私钥。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、服务端的公钥，第三方证书认证机构(CA)的签名，服务端的域名信息等内容。

4、客户端**解析证书**，这部分工作是由客户端的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（**预主秘钥**）。

5、客户端认证证书通过之后，接下来是通过**随机值1**、**随机值2**和**预主秘钥**组装**会话秘钥**。然后通过证书的公钥加密会话秘钥。

6、传送加密信息，这部分传送的是用证书加密后的**会话秘钥**，目的就是让服务端使用秘钥解密得到随机值1、随机值2和预主秘钥。

7、服务端解密得到**随机值1**、**随机值2**和**预主秘钥**，然后组装会话秘钥，跟客户端会话秘钥相同。

8、客户端通过会话秘钥**加密**一条消息发送给服务端，主要验证服务端是否正常接受客户端加密的消息。

9、同样服务端也会通过会话秘钥**加密**一条消息回传给客户端，如果客户端能够正常接受的话表明SSL层连接建立完成了。

**16、转发和重定向**

**forward：（转发）**指服务器内部的转发，服务器直接访问目标地址的URL，把那个URL的响应内容读取过来，而客户端并不知道，因此在客户端的地址栏中不会显示转向后的地址，还是原来的地址。由于整个转发的过程用的是同一个Request，因此forward会将Request的信息带到被定向的JSP或Servlet中使用。

**redirect：（重定向）**客户端的重定向，是完全的跳转，即客户端浏览器会获取到跳转后的地址，然后重新发送请求，因此浏览器中显示跳转后的地址。同时，这种方式比forward方式多了一次网络请求，因此效率要低于forward方式。