链滴

计算机网络常见面试题

作者：fjiang

原文链接：https://ld246.com/article/1624700480650

来源网站：链滴

1. TCP三次握手和四次挥手

1.1 TCP三次握手和四次挥手的过程

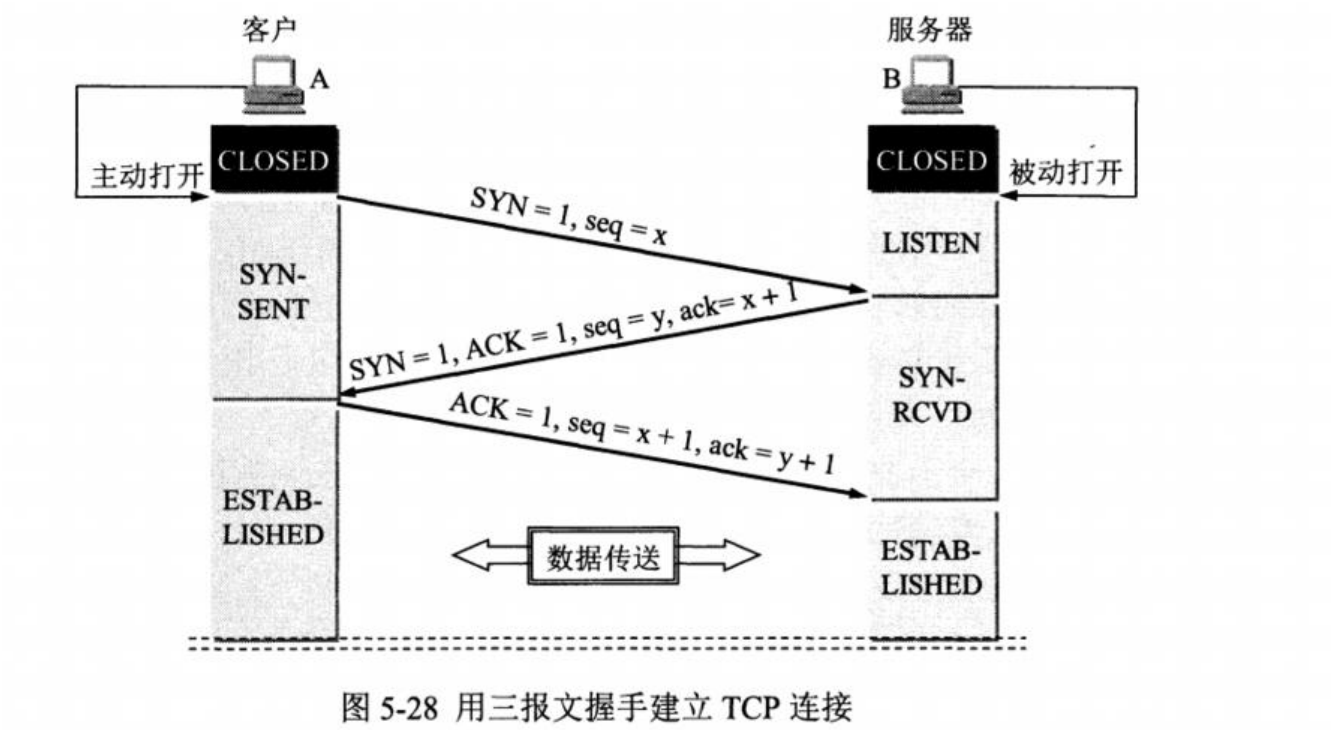


图 5-28 用三报文握手建立 TCP 连接

** **三次握手**

（1）第一次握手：Client将标志位SYN置为1，随机产生一个值seq=J，并将该数据包发送给Serve，Client进入SYN_SENT状态，等待Server确认。

****（2）第二次握手：Server收到数据包后由标志位SYN=1知道Client请求建立连接，Server将标位SYN和ACK都置为1，ack=J+1，随机产生一个值seq=K，并将该数据包发送给Client以确认连接求，Server进入SYN_RCVD状态。****

****（3）第三次握手：Client收到确认后，检查ack是否为J+1，ACK是否为1，如果正确则将标志位CK置为1，ack=K+1，并将该数据包发送给Server，Server检查ack是否为K+1，ACK是否为1，如果确则连接建立成功，Client和Server进入ESTABLISHED状态，完成三次握手，随后Client与Server之可以开始传输数据了。**

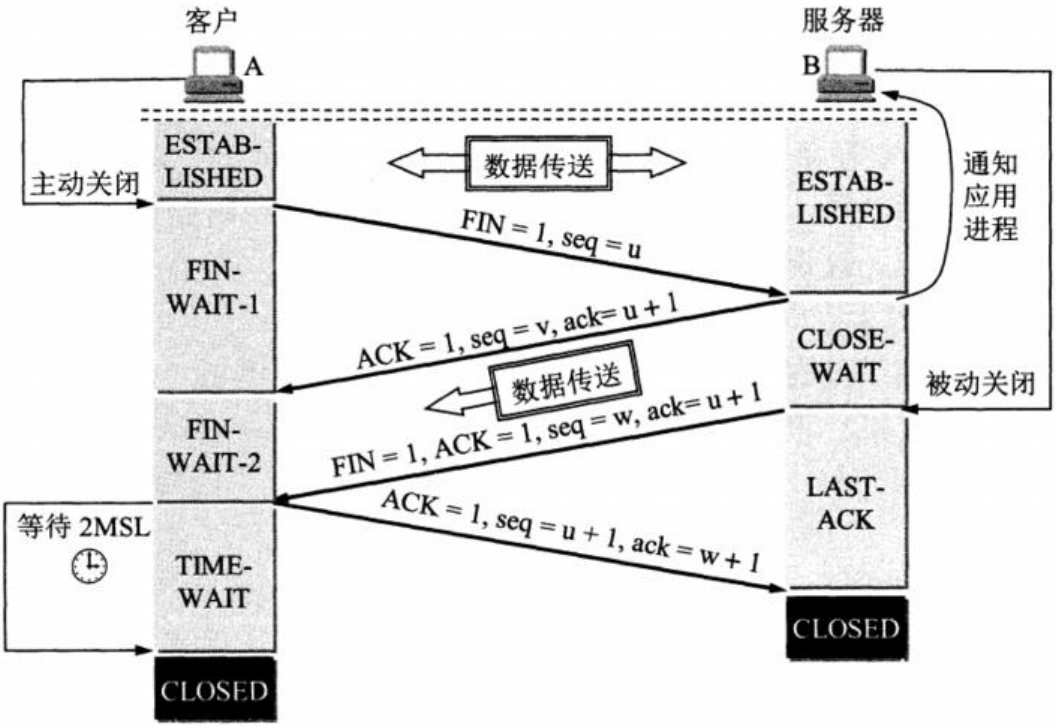


图 5-29　TCP 连接释放的过程

** **四次挥手**

（1）第一次挥手：Client发送一个FIN，用来关闭Client到Server的数据传送，Client进入FIN_WAI_1状态。

（2）第二次挥手：Server收到FIN后，发送一个ACK给Client，确认序号为收到序号+1（与SYN相，一个FIN占用一个序号），Server进入CLOSE_WAIT状态。

（3）第三次挥手：Server发送一个FIN，用来关闭Server到Client的数据传送，Server进入LAST_ACK状态。

（4）第四次挥手：Client收到FIN后，Client进入TIME_WAIT状态，接着发送一个ACK给Server确认序号为收到序号+1，Server进入CLOSED状态，完成四次挥手。

1.2 为什么不是两次握手 四次握手?

** **这是为了避免服务器建立无用连接** ****（客户端服务器建立连接后，却不传输数据）**

** **如果只进行两次握手，如果客户端向服务器第一次发送的建立连接的请求因为某原因，兜兜转绕了一大圈才到达服务器。这期间客户端因为未收到服务器的响应，就会再次发送连接请求，这时服器收到了，向客户端发送连接请求后，连接便建立了。然后数据传输完毕后，释放连接。这时刚刚兜转转一大圈的建立连接的请求到了服务器，服务器收到后再次向客户端发送请求，发送后又建立了连，但是建立连接后客户端没有再理会服务器，客户端与服务器之间没有传输数据，此时服务器的资源会被浪费**

** **为什么不是四次握手**

** **因为通信不可能100%可靠**，而上面的三次握手已经做好了通信的准备工作，**再增加握手，并能显著提高可靠性**，所以只需要三次握手就足够了**

1.3 为什么TCP建立连接需要三次握手,断开连接需要四次挥手

● **建立连接的时候， 服务器在LISTEN状态下，收到建立连接请求的SYN报文后，** **把ACK和SY放在一个报文里发送给客户端**

● **关闭连接时，服务器收到对方的FIN报文时，仅仅表示对方不再发送数据了但是还能接收数据，自己也未必全部数据都发送给对方了，所以己方可以立即关闭，也可以发送一些数据给对方后，再发FIN报文给对方来表示同意现在关闭连接，因此，**己方ACK和FIN一般都会分开发送，从而导致多一次**

1.4 TCP四次挥手为什么有Time-Wait过程

** **虽然按道理，四个报文都发送完毕，我们可以直接进入CLOSE状态了，但是我们必须假象网络不可靠的，有可以最后一个ACK丢失。所以TIME_WAIT状态就是用来重发可能丢失的ACK报文。在Client发送出最后的ACK回复，但该ACK可能丢失。Server如果没有收到ACK，将不断重复发送FIN片。所以Client不能立即关闭，它必须确认Server接收到了该ACK。Client会在发送出ACK之后进入到TME_WAIT状态。Client会设置一个计时器，等待2MSL的时间。如果在该时间内再次收到FIN，那么Cient会重发ACK并再次等待2MSL。所谓的2MSL是两倍的MSL(Maximum Segment Lifetime)。SL指一个片段在网络中最大的存活时间，2MSL就是一个发送和一个回复所需的最大时间。如果直到MSL，Client都没有再次收到FIN，那么Client推断ACK已经被成功接收，则结束TCP连接。**

1.5 TCP第三次握手可以传输数据吗

** ****不能，需要三次握手成功之后才开始发送数据 **

2. TCP和UDP的区别

** 答： **

1. **连接：UDP是无连接的，发送数据之前无需建立连接，发送数据结束后也无需释放连接。TCP是向连接的，在发送数据之前需要通过三次握手建立连接，发送数据结束后需要通过四次挥手释放连接。**

2. **交付：UDP使用尽最大努力交付，即不保证可靠交付，主机不需要维持复杂的连接状态表。TCP供可靠交付，即通过TCP连接传送的数据，无差错、不丢失、不重复，并且按序到达。 **

3. **数据：UDP是面向报文的，UDP对于应用层交下来的报文，添加首部后就向下交给IP层，既不合，也不拆分，一次交付一个完整的报文。TCP是面向字节流的，虽然应用程序和TCP的交互是一次一数据块（大小不等），但TCP把应用程序交下来的数据看出成一连串无结构的字节流。TCP不保证接方应用程序所收到的数据块和发送方应用程序所发出的数据块具有对应大小关系。 **

4. **通信双方：UDP支持一对一、一对多、多对一、多对多的交互通信。TCP连接是点对点（一对一，每一条TCP连接只能有两个端点。 **

5. **拥塞：UDP没有拥塞控制，网络的拥塞不会使源主机的发送速率降低。TCP通过慢开始、拥塞避免快重传、快恢复等算法**进行拥塞控制。 **

6. **首部：UDP首开销小，只有8个字节。TCP首部是20个字节。 **

3. TCP的可靠传输

3.1 TCP如何确保可靠性传输

** **** 对TCP而言在三次握手时的SYN标志会使用上一个ISN值，这个值是使用32位计数器，内由0 4294967295，每一次连接容都会分配到一个ISN值，连接双方对这个值会记录共识，假如这个值不，就说明了这个连接已超时或无效甚至是被人恶意攻击冒充连接。 **

**3.2 TC的流量控制 （防止发送方发的太快，耗尽接收方的资源，从而使接收方来不及处理） **

** **1.滑动窗口是什么?

** **滑动窗口是类似于一个窗口一样的东西，是用来告诉发送端可以发送数据的大小或者说是窗口记了接收端缓冲区的大小，这样就可以实现 **ps：窗口指的是一次批量的发送多少数据

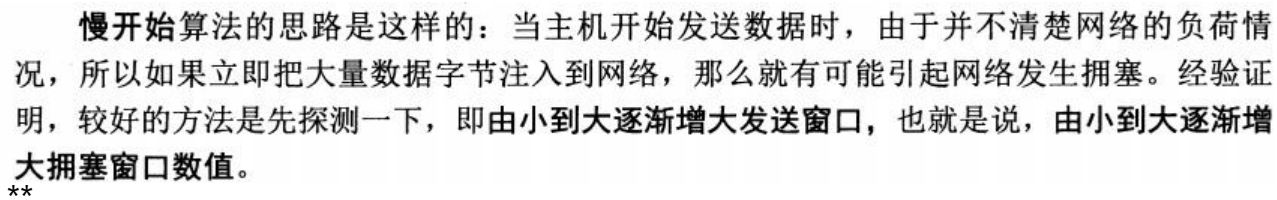
** **2.为什么会出现滑动窗口?

** **在确认应答策略中，对每一个发送的数据段，都要给一个ACK确认应答，收到ACK后再发下一个数据段，这样做有一个比较大的缺点，就是性能比较差，尤其是数据往返的时间长的时候

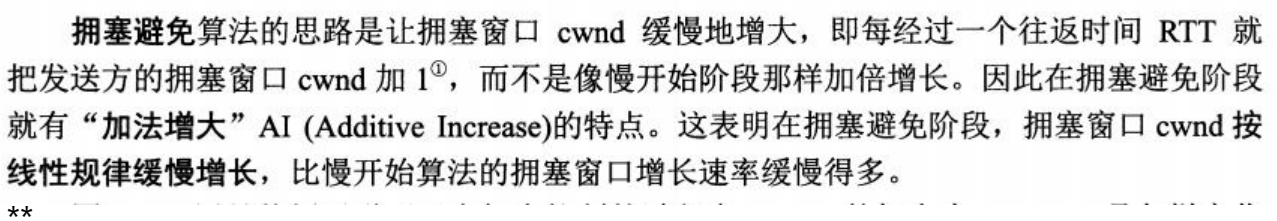
** **使用滑动窗口，就可以一次发送多条数据，从而就提高了性能

3.3 TCP的拥塞控制（防止发送方发的太快，使得网络来不及处理，从而导致网络拥塞）

1. **慢开始 **

**

慢开始算法的思路是这样的：当主机开始发送数据时，由于并不清楚网络的负荷情况，所以如果立即把大量数据字节注入到网络，那么就有可能引起网络发生拥塞。经验证明，较好的方法是先探测一下，即由小到大逐渐增大发送窗口，也就是说，由小到大逐渐增大拥塞窗口数值。
**

** ****2. 拥塞避免 **

**

拥塞避免算法的思路是让拥塞窗口 cwnd 缓慢地增大，即每经过一个往返时间 RTT 就把发送方的拥塞窗口 cwnd 加 1[①]，而不是像慢开始阶段那样加倍增长。因此在拥塞避免阶段就有“加法增大”AI (Additive Increase)的特点。这表明在拥塞避免阶段，拥塞窗口 cwnd 按线性规律缓慢增长，比慢开始算法的拥塞窗口增长速率缓慢得多。
**

3. **快重传 **

当发送方没有按时收到某个应当按时到达的确认报文时，就会要求接收方每收到一个失序的报文段后会立即发出重复确认，连续收到三个重复确认后，发送方立即重传未被确认的报文。

4. **快恢复 ****

****发送方收到三个重复确认后，会把慢开始门限ssthresh设置为出现拥塞时拥塞窗口值的一半，并令塞窗口值与其相等，然后执行拥塞避免算法，使拥塞窗口缓慢线性增大。**

3.4 TCP传输通信时,客户端突然断开连接,服务端如何判断

** **TCP还设有一个保活计时器，显然，客户端如果出现故障，服务器不能一直等下去，白白浪费源。服务器每收到一次客户端的请求后都会重新复位这个计时器，时间通常是设置为2小时，若两小还没有收到客户端的任何数据，服务器就会发送一个探测报文段，以后每隔75秒钟发送一次。若一连送10个探测报文仍然没反应，服务器就认为客户端出了故障，接着就关闭连接。**

4. 常见的HTTP状态码

** 301，302，304，403，404，500，503 **

****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

更改的URL； **

****302 Found：****临时性重定向，表示请求的资源被分配了新的URL，希望本次访问使用新的UR；**

** 301与302的区别：前者是永久移动，后者是临时移动（之后可能还会更改URL）**

304 Not Modified：表示客户端发送附带条件的请求时，服务器端允许访问资源，但是请求未满条件的情况下返回改状态码；**

****403 Forbidden：****服务器拒绝该次访问（访问权限出现问题）**

****404 Not Found：****表示服务器上无法找到请求的资源，除此之外，也可以在服务器拒绝请求不想给拒绝原因时使用；**

****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bg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；**

5. HTTP报文

5.1 HTTP请求报文和响应报文的组成

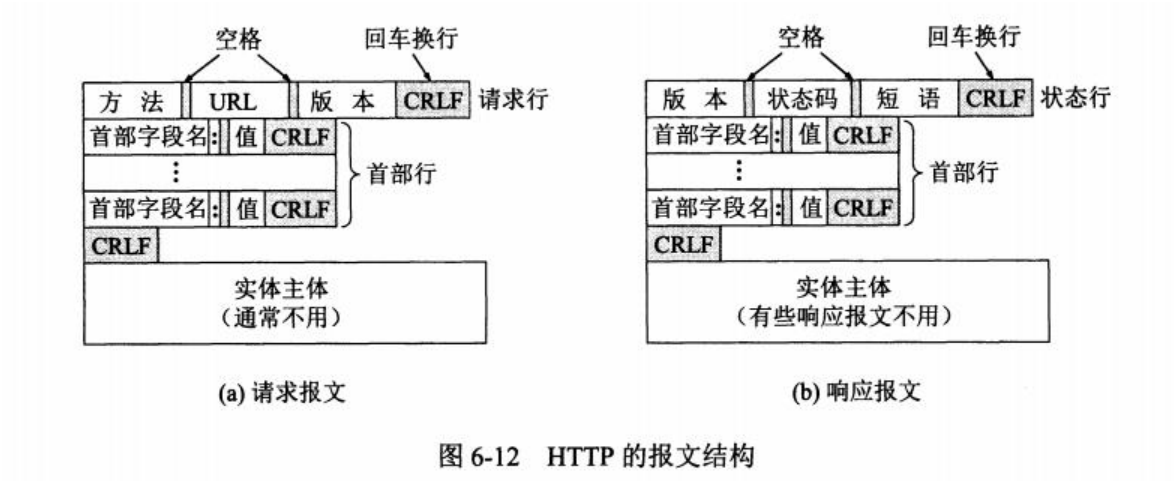


图 6-12 HTTP 的报文结构

5.2 HTTP请求报文包含哪些方法，GET和POST的区别

1. **参数位置：GET方法参数位置包含在URL，POST方法参数包含在请求主体 **
2. **参数长度：GET方法的URL长度有限度，POST长度没有限制 **
3. **参数编码：GET方法参数编码是ASCII码，POST没有限制 **
4. **TCP数据包：GET方法产生一个TCP数据包，把首部和数据一起发送，POST方法产生两个TCP数包，先发首部，服务器响应后再发数据 **

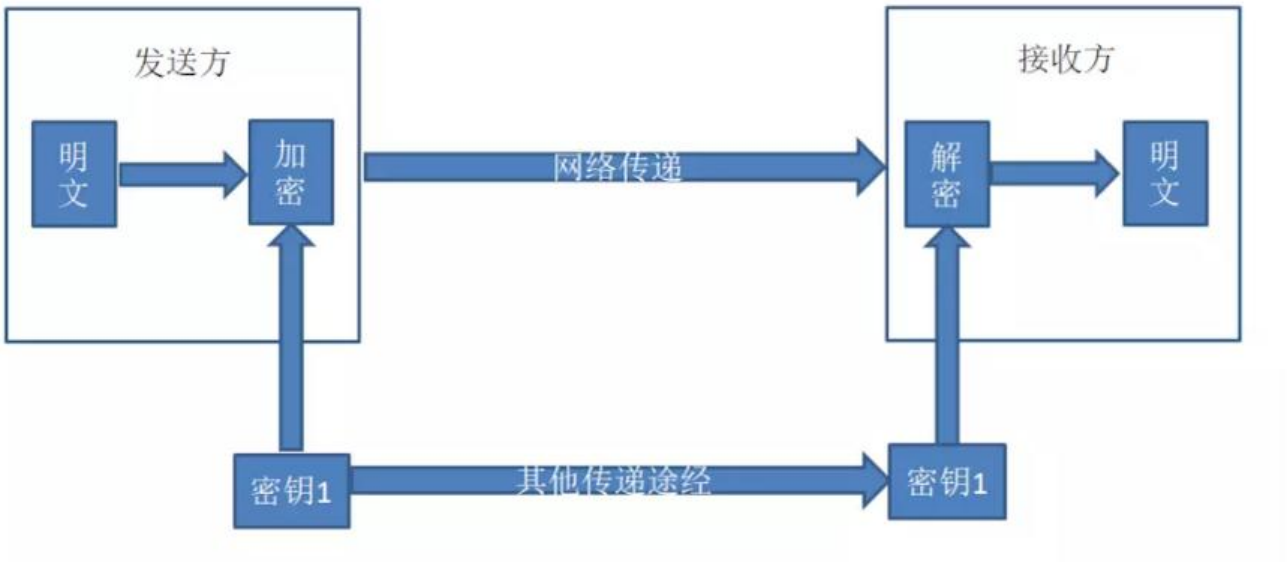
6. HTTP和HTTPS的区别

1. **端口 ： HTTP的URL由“http://”起始且默认使用端口80，而HTTPS的URL由“https://”起始且认使用端口443。 **
2. **安全性和资源消耗： HTTP协议运行在TCP之上，所有传输的内容都是明文，客户端**和服务器端

无法验证对方的身份。HTTPS是运行在SSL/TLS之上的HTTP协议，SSL/TLS 运行在TCP之上。所有传的内容都经过加密，加密采用对称加密，但对称加密的密钥用服务器方的证书进行了非对称加密。所说，HTTP 安全性没有 HTTPS高，但是 HTTPS 比HTTP耗费更多服务器资源。 **

3. **对称加密：密钥只有一个，加密解密为同一个密码，且加解密速度快，典型的对称加密算法**有DE、AES等； **

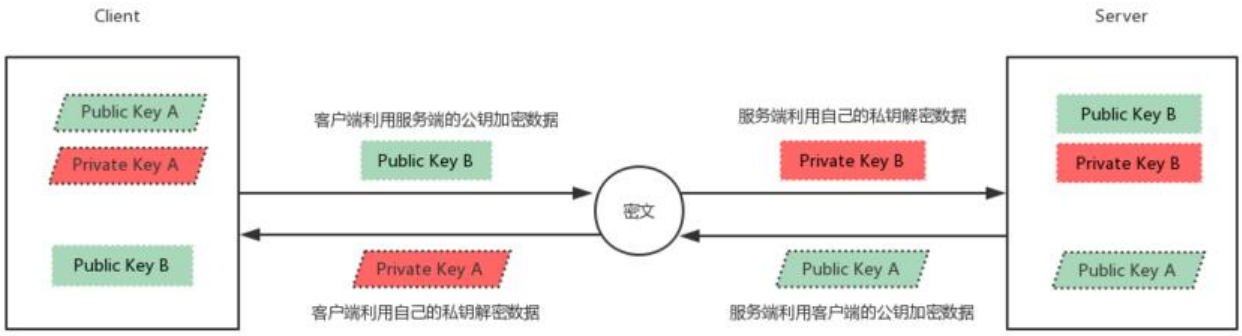
**



**

1. **非对称加密：密钥成对出现（且根据公钥无法推知私钥，根据私钥也无法推知公钥），加密解密使不同密钥（公钥加密需要私钥解密，私钥加密需要公钥解密），相对对称加密速度较慢，典型的非对加密算法**有RSA、DSA等。 **

2.



7. HTTP1.0和1.1和2.0的区别

**HTTP1.0和HTPP1.1的区别 **

1. **缓存处理：在HTTP1.0中主要使用header里的If-Modified-Since,Expires来做为缓存判断的标准HTTP1.1则引入了更多的缓存控制策略例如Entity tag，If-Unmodified-Since, If-Match, If-None-Match等更多可供选择的缓存头来控制缓存策略。 **

2. **带宽优化及网络连接的使用：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则在请求头引入了ange头域，它允许只请求资源的某个部分，即返回码是206（Partial Content），这样就方便了开发自由的选择以便于充分利用带宽和连接。 **

3. **错误通知的管理：在HTTP1.1中新增了24个错误状态响应码，如409（Conflict）表示请求的资与资源的当前状态发生冲突；410（Gone）表示服务器上的某个资源被永久性的删除。 **

4. **Host头处理：在HTTP1.0中认为每台服务器都绑定一个唯一的IP地址，因此，请求消息中的URL没有传递主机名（hostname）。但随着虚拟主机技术的发展，在一台物理服务器上可以存在多个虚主机（Multi-homed Web Servers），并且它们共享一个IP地址。HTTP1.1的请求消息和响应消息应支持Host头域，且请求消息中如果没有Host头域会报告一个错误（400 Bad Request）。 **

5. **长连接：HTTP 1.1支持长连接（PersistentConnection）和请求的流水线（Pipelining）处理，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中认开启Connection： 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。 **

****HTTP2.0与HTTP1.x的区别 ****

1. **新的二进制格式（Binary Format）：HTTP1.x的解析是基于文本。基于文本协议的格式解析存天然缺陷，文本的表现形式有多样性，要做到健壮性考虑的场景必然很多，二进制则不同，只认0和1组合。基于这种考虑HTTP2.0的协议解析决定采用二进制格式，实现方便且健壮 **

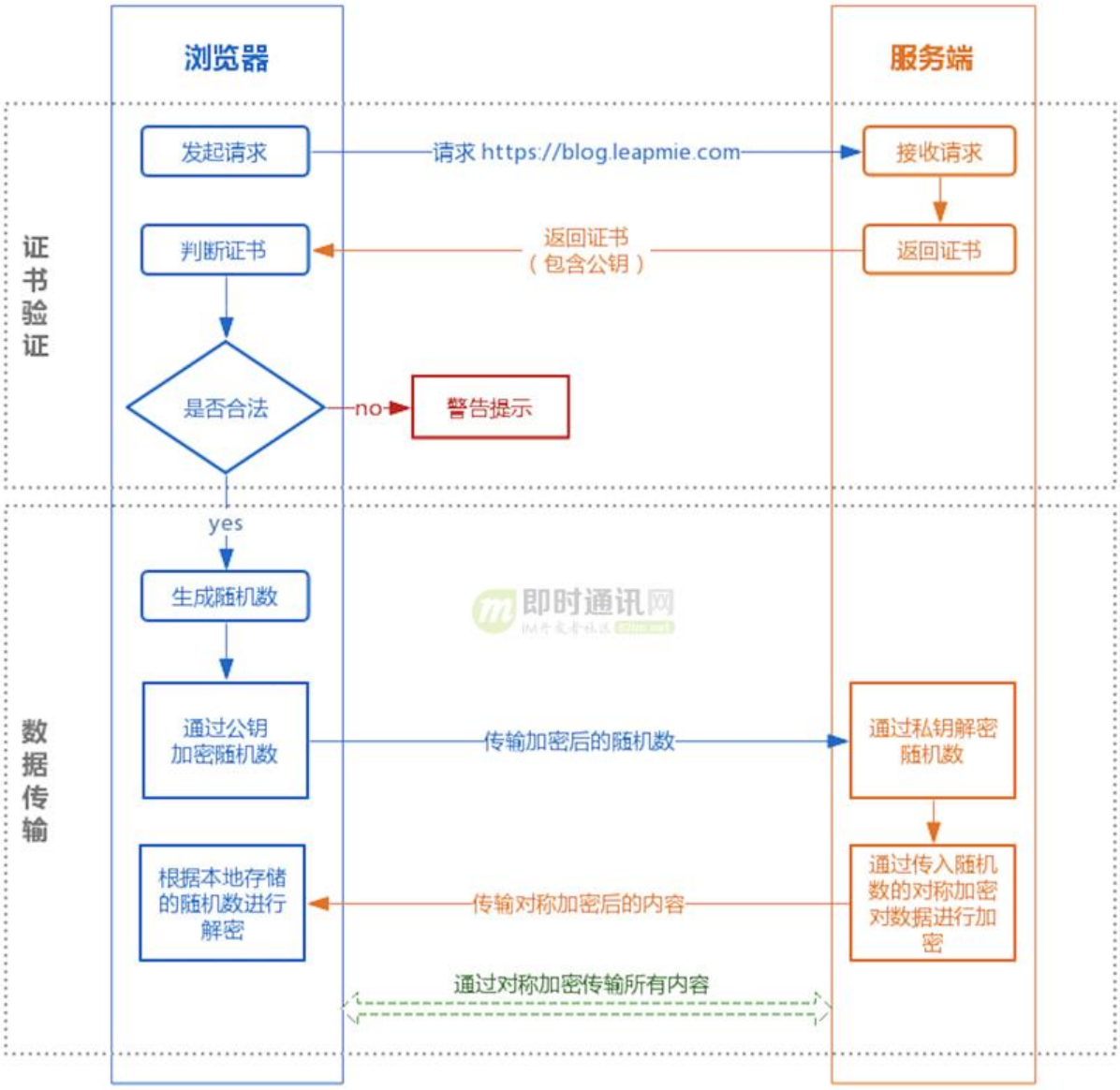
2. **多路复用（MultiPlexing）：即连接共享，即每一个request都是是用作连接共享机制的。一个request对应一个id，这样一个连接上可以有多个request，每个连接的request可以随机的混杂在一起，收方可以根据request的 id将request再归属到各自不同的服务端请求里面。 **

3. **header压缩：HTTP1.x的header带有大量信息，而且每次都要重复发送，HTTP2.0使用encode来减少需要传输的header大小，通讯双方各自cache一份header fields表，既避免了重复header的输，又减小了需要传输的大小。 **

4. **服务端推送（server push）：同SPDY一样，HTTP2.0也具有server push功能。 **

8. HTTPS密钥交换过程

** **



1. **证书验证，客户端发送一个证书请求个服务器端，服务器端返回证书，客户端**对证书进行验证。 **
2. **交换密钥，使用非对称加密，客户端**使用公钥进行加密，服务器端使用密钥解密。 **
3. **交换数据，使用对称加密的方式对数据进行加密，然后进行传输。 **

9. 输入URL跳转网页的过程

1. **DNS域名解析 **
2. **HTTP协议生成请求报文 **
3. **TCP协议将请求报文分割成报文段，进行可靠传输 **
4. **IP协议进行分组转发 **
5. **TCP协议重组请求报文 **
6. **HTTP协议对请求进行处理 **

10. 计算机网络四层协议,五层协议,七层协议

原文链接：计算机网络常见面试题

** **

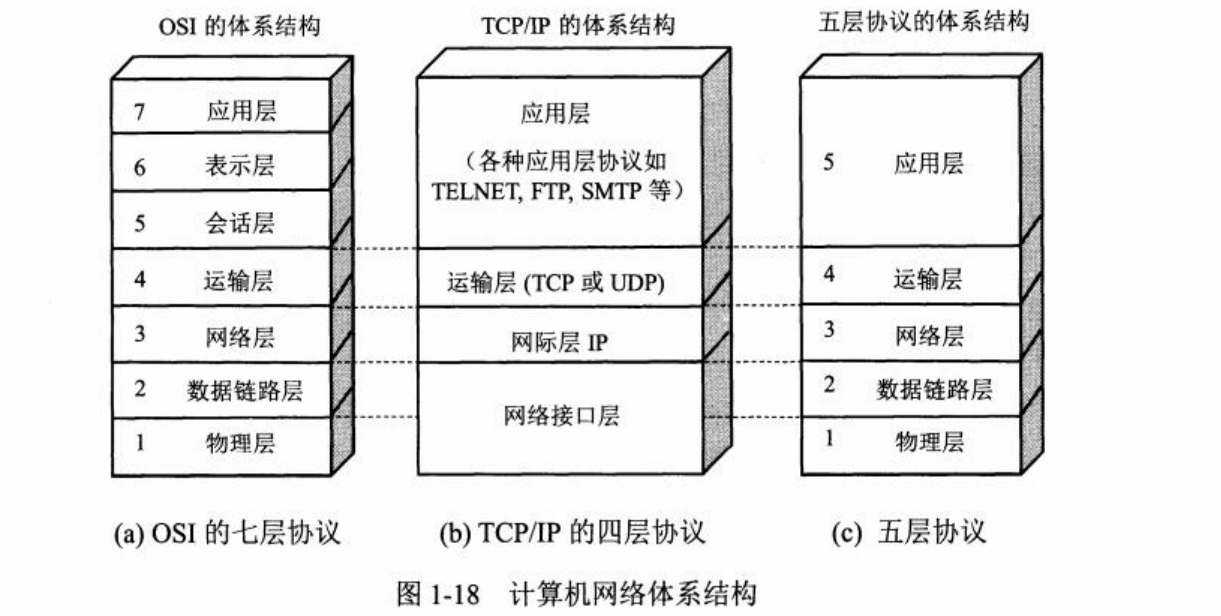


图 1-18　计算机网络体系结构

**TCP/IP协议各层的作用 **

1. **应用层：应用层的任务是通过应用进程之间的交互来完成特定网络应用。应用层协议定义的是应进程间通信和交互的规则。这里的进程是指主机中正在与运行的程序。对于不同的网络应用需要有不的应用层协议。在互联网中的应用层协议有很多，如域名系统DNS，支持万维网应用的HTTP协议，持电子邮件的SMTP协议，等等。 **

2. **传输层：传输层的任务是负责向两台主机中进程之间的通信提供通用的数据传输服务。应用进程用该服务传送应用层报文。主要有TCP协议和UDP协议。 **

3. **网络层：网络层的任务是为分组交换网上的不同主机提供通信服务。在发送数据的时候，网络层运输层产生的报文段或者用户数据报封装成分组或包进行传送。主要使用IP协议。 **

4. **网络接口层：操作系统中的设备驱动和计算机对应的网络接口，它们一起处理与传输媒介的物理接细节。主要有ARP协议和**

11. 什么是cookie和session,区别是什么, 禁用cookie怎么办

** 答： **

1. **储存位置：Cookie是客户端会话技术，数据保存在客户端**，Session是服务器端会话技术，数据存在服务器端。 **

2. **存储容量：Cookie一般<=4KB，Session无限制。 **

3. **跨域支持：Cookie支持跨域，Session不支持。 **

当用户禁止cookie以后，如果要访问某个需要session机制支持的web组件（jsp/servlet），此时，能直接在浏览器地址栏输入该组件的地址，要使用服务器生成的地址，该地址可以使用以下方法来实：

response.encodeURL(String url);

//该方法会在url后面添加sessionId。

**//该方法用在链接地址、表单提交地址。

//如果是重定向，则使用**

response.encodeRedirectURL(String url);

**response.sendRedirect(response.encodeRedirectURL(String url));