**HTTP协议+实例**

# HTTP协议

## 一、概念

HTTP协议，即超文本传输协议(Hypertext transfer protocol)。是一种详细规定了浏览器和万维网(WWW = World Wide Web)服务器之间互相通信的规则，通过因特网传送万维网文档的数据传送协议。

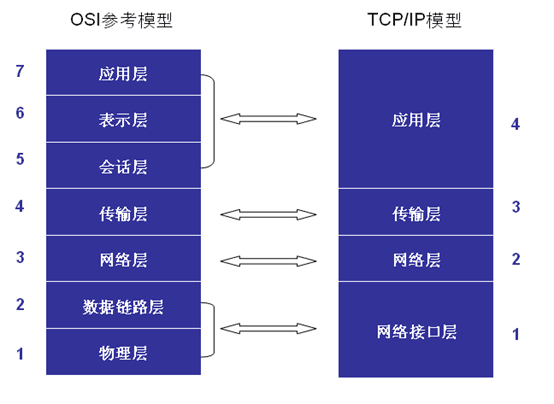
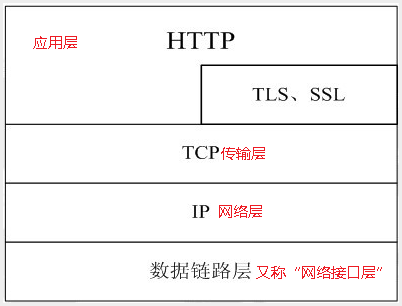
### 1、HTTP协议的特点

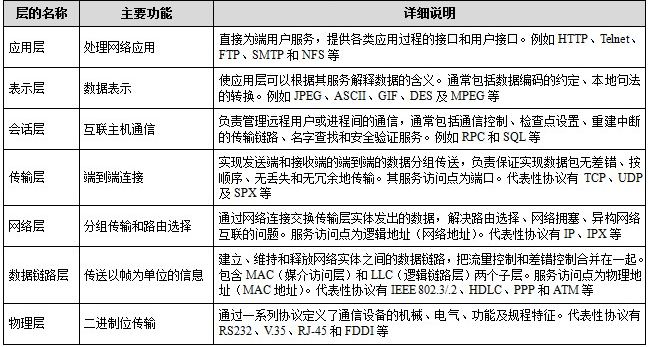
1. 支持客户/服务器模式。
2. 简单快速：客户向服务器请求服务时，只需传送请求方法和路径。请求方法常用的有GET、HEAD、POST。每种方法规定了客户与服务器联系的类型不同。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快。
3. 灵活：HTTP允许传输任意类型的数据对象。正在传输的类型由Content-Type加以标记。
4. 无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。
5. 无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。

### 2、TCP/IP 模型和HTTP连接

#### 2.1、TCP/IP 模型

在Internet中所有的传输都是通过TCP/IP进行的。HTTP协议作为TCP/IP模型中应用层的协议也不例外。HTTP协议通常承载于TCP协议之上，有时也承载于TLS或SSL协议层之上，这个时候，就成了我们常说的HTTPS。如下图所示：





浏览网页是HTTP的主要应用，但是这并不代表HTTP就只能应用于网页的浏览。HTTP是一种协议，只要通信的双方都遵守这个协议，HTTP就能有用武之地。比如咱们常用的QQ，迅雷这些软件，都会使用HTTP协议(还包括其他的协议)。

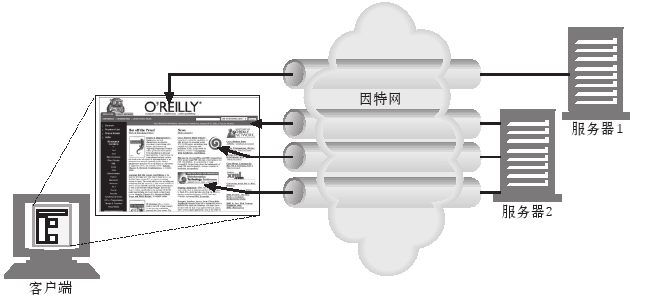
#### 2.2、HTTP连接

HTTP通信是由TCP/IP承载的，HTTP紧挨着TCP，位于其上层，所以HTTP事务的性能很大程度上取决于底层TCP通道的性能。

如果对事务进行串行加载，即发送一个请求收到一个响应，再发送一个请求接收一个响应。比如加载一副图片时页面上其他地方都没有动静会让人觉得速度很慢。用户更希望能够同时加载多副图片。有四种技术可以提高HTTP的连接性能：并行连接，持久连接，管道化连接，复用的连接。

##### 2.2.1并行连接

HTTP允许客户端打开多条连接，并行地执行多个HTTP事务。并行连接可能会提高页面的加载速度，但是并行连接不一定更快。比如带宽不足时。下面例子中，并行加载四副嵌入式图片，每个事务都有自己的TCP连接，浏览器将并行连接的总数限制为一个较小的数值。服务器可随意关闭来自特定客户端的超量连接。。



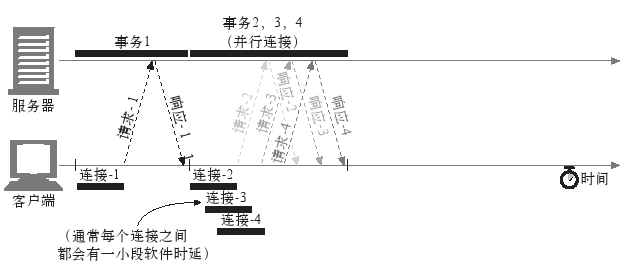
##### 2.2.2持久连接

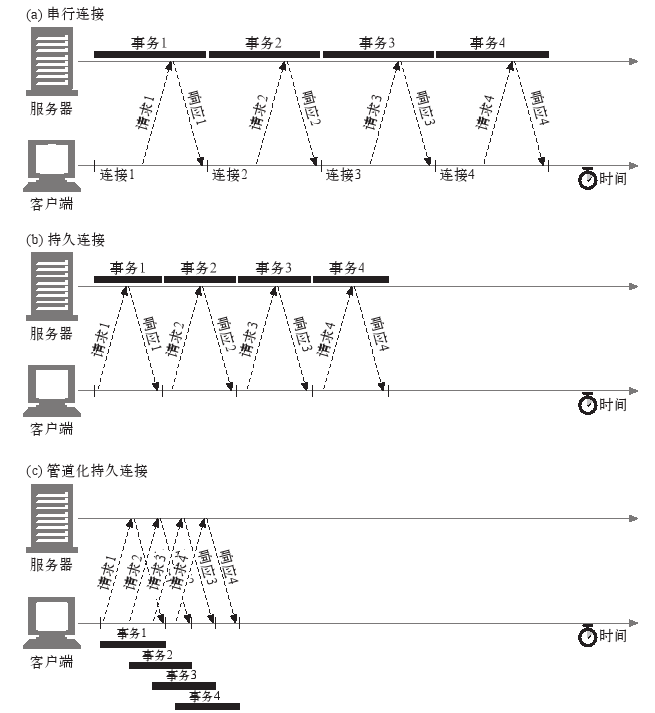
在事务处理结束后仍然保持打开状态的TCP连接被称为持久连接。重用已对服务器打开的空闲持久连接，可以避开缓慢的连接建立阶段，而且，还可以避免慢启动的拥塞适应阶段，以便更快速地传输数据。持久连接有两种类型：HTTP/1.0+ “keep-alive”以及HTTP/1.1 ”persistent"。持久连接和并行连接配合使用可能是最高效的方式。

##### 2.2.3管道化连接

HTTP/1.1允许在持久连接上可选地使用请求管道。在响应到达之前，可以将多条请求放入队列。当第一条请求通过网络流向服务器时，第二条和第三条请求也可以开始发送。但是HTTP客户端必须做好连接会在任意时刻关闭的准备，还要准备好重发所有未完成的管道化请求。同时HTTP客户端不应该用管道化的方式发送会产生副作用的请求（比如POST）。

**连接比较**：





##### 2.2.4复用的连接

交替传送请求和响应的报文（实验阶段）。

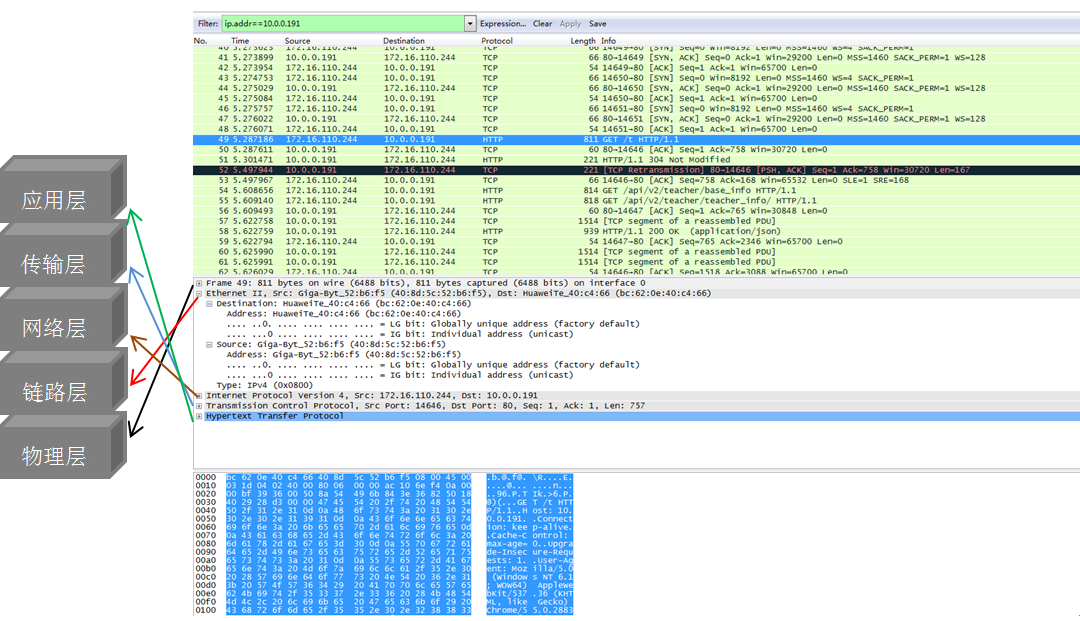
#### 2.3、TCP/IP报文封装

http://my.csdn.net/uploads/201205/25/1337910967_8133.jpg

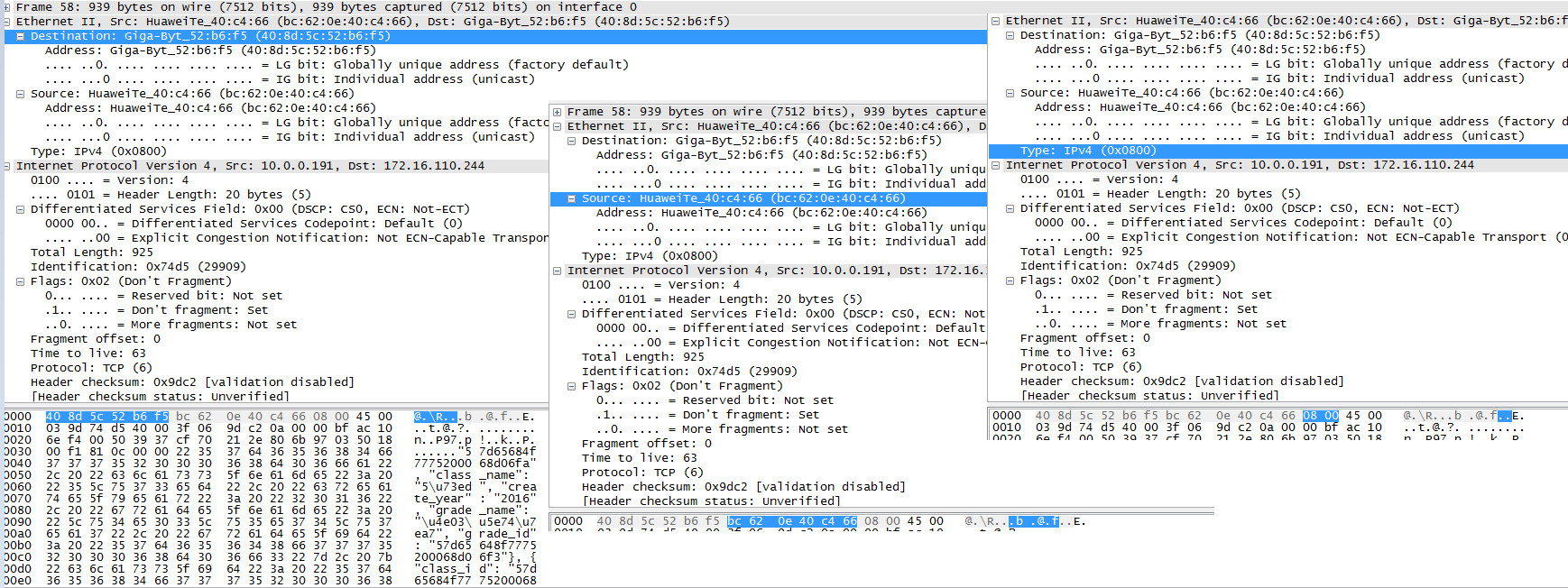
帧：是指封装上MAC地址和[FCS](https://www.baidu.com/s?wd=FCS&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9Pyn4PvcvnHbkrj6smvf30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHbzn1TYrHfYrH63rHD4njfzPs" \t "_blank)的数据结构，最大为1522bytes。

数据报文：一般指[IP](https://www.baidu.com/s?wd=IP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9Pyn4PvcvnHbkrj6smvf30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHbzn1TYrHfYrH63rHD4njfzPs)datagram,就是封装了[IP](https://www.baidu.com/s?wd=IP&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9Pyn4PvcvnHbkrj6smvf30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnHbzn1TYrHfYrH63rHD4njfzPs" \t "_blank)报文头的数据结构。

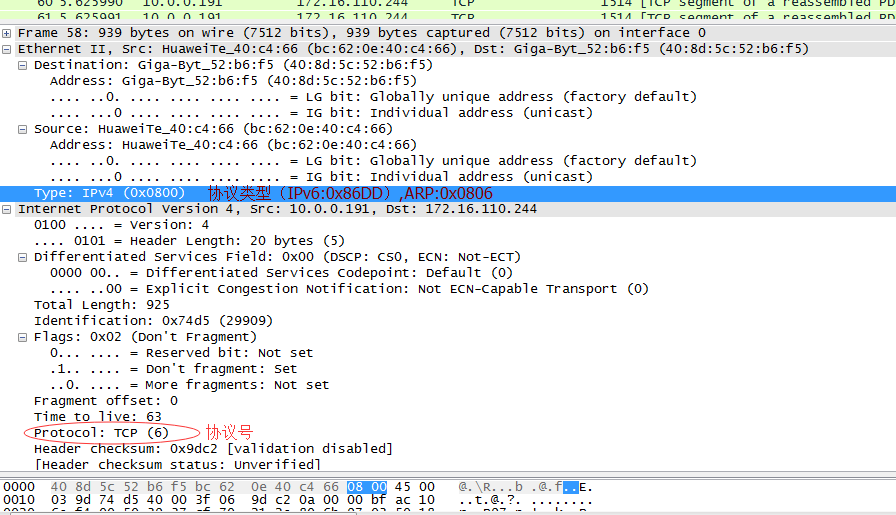
例如：应用层向传输层提交数据，通过wirshark抓包效果如下:



和邮局寄信是一个概念。应用层就是写信，并且放入信封。传输层就是你把信件投到邮局，网络层就是邮局把你的邮件投递到对方所属的中心邮局，链路层就是对方中心邮局分发到所属的小邮政点，物理层就是绿衣使者骑着自行车把信送到您的手上。



层层封装；



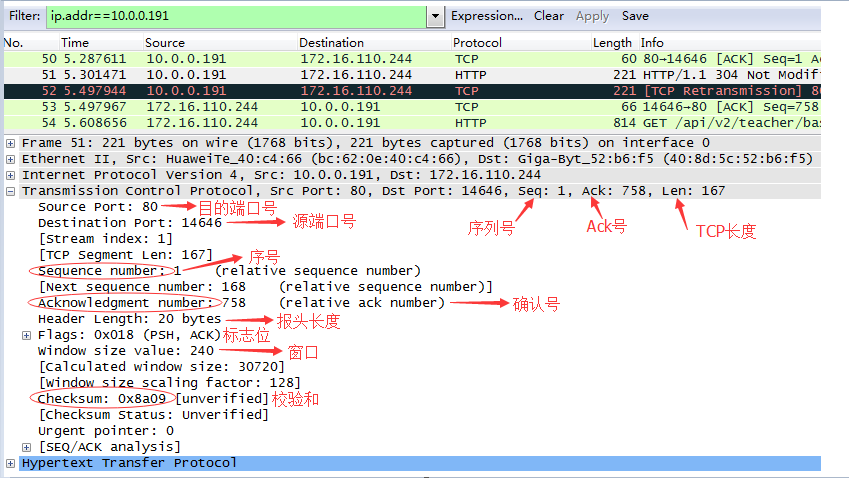
IPv4，是[互联网协议](http://baike.baidu.com/view/1087323.htm" \t "_blank)（Internet Protocol，IP）的第四版，也是第一个被广泛使用；IPv6正处在不断发展和完善的过程中，它在不久的将来将取代目前被广泛使用的IPv4。

常用IP协议号：

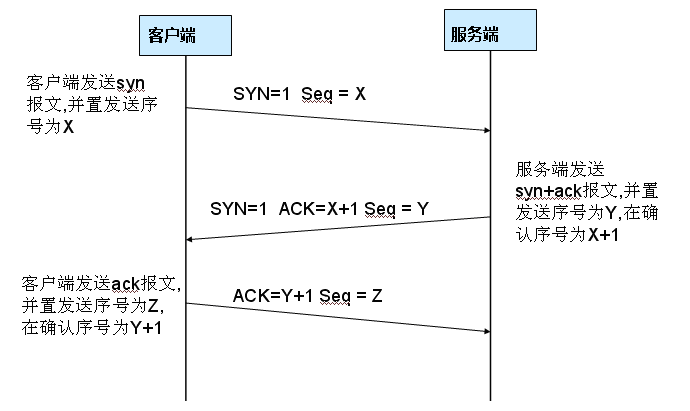


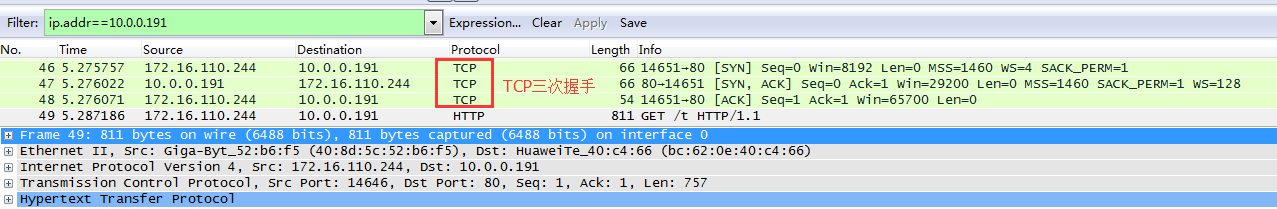
#### 2.4、TCP报文格式





#### 2.5、TCP三次握手过程

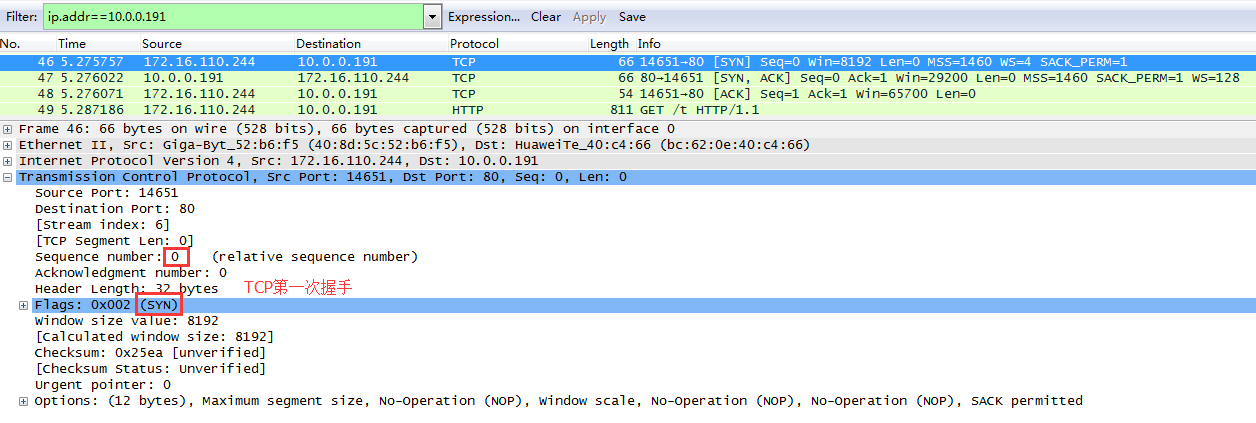




图中可以看到wireshark截获到了三次握手的三个数据包。第四个包才是HTTP的，这说明HTTP的确是使用TCP建立连接的。

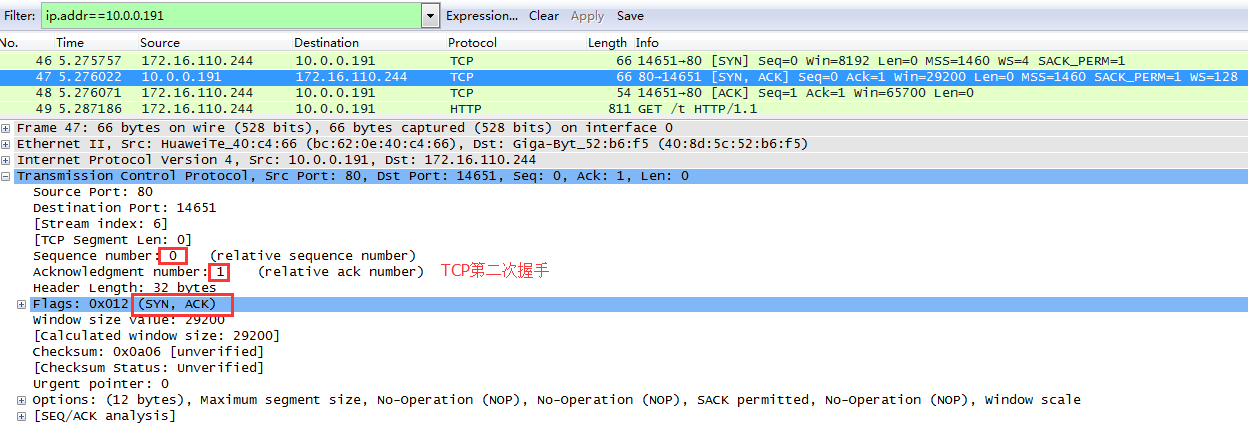
第一次握手数据包：

客户端发送一个TCP，标志位为SYN，序列号为0， 代表客户端请求建立连接。如下图



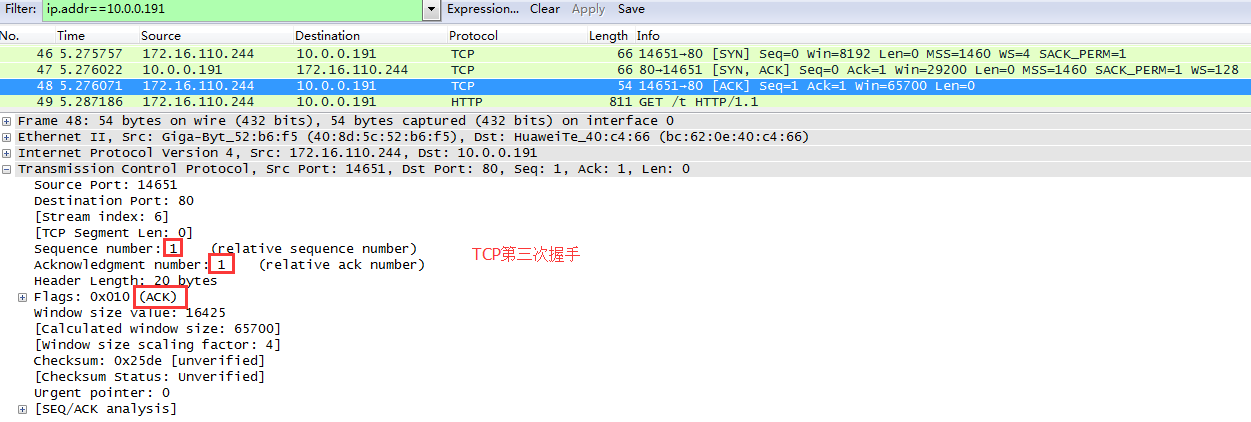
第二次握手的数据包：

服务器发回确认包,标志位为SYN,ACK.将确认序号(Acknowledgement Number)设置为客户的ISN加1以.即0+1=1,如下图



第三次握手的数据包：

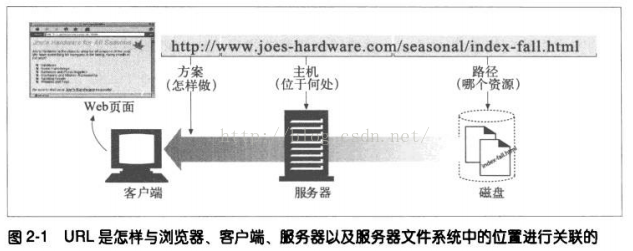
客户端再次发送确认包(ACK)SYN标志位为0,ACK标志位为1.并且把服务器发来ACK的序号字段+1,放在确定字段中发送给对方，并且在数据段放写ISN的+1,如下图



### 3、HTTP URL与资源

HTTP URL (URL是一种特殊类型的URI，包含了用于查找某个资源的足够的信息)的格式：[http://host[":"port][abs\_path](http://host[/)]。

http表示要通过HTTP协议来定位网络资源；host表示合法的Internet主机域名或者IP地址；port指定一个端口号，为空则使用缺省端口80；abs\_path指定请求资源的URI；如果URL中没有给出abs\_path，那么当它作为请求URI时，必须以“/”的形式给出，通常这个工作浏览器自动帮我们完成。  
eg:

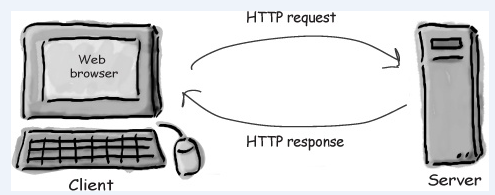
1. 输入：yixun.idealsee.com  
   浏览器自动转换成：http://yixun.idealsee.com  
   <http://10.0.0.190:55672>
2. URL分三部分：  
   例子：http://www.ifeng.com/news/index.html  
   第一部分（http）：URL方案。告知web客户端怎样访问资源。  
   第二部分（www.ifeng.com）：服务器的位置。告知web客户端资源位于何处。  
   第三部分（/news/index.html）：资源路径。路径说明了请求的是服务器上哪个特定的本地资源。  
   
3. 例子<http://www.joes-hardware.com/specials/saw-blade.gif>



## 二、HTTP请求响应

### 1、HTTP请求响应流程

HTTP协议永远都是客户端发起请求，服务器回送响应。无法实现在客户端没有发起请求的时候，服务器将消息推送给客户端。HTTP协议是一个无状态的协议，同一个客户端的这次请求和上次请求是没有对应关系。见下图：



### 2、HTTP通信机制

HTTP通信机制是在一次完整的HTTP通信过程中，Web浏览器与Web服务器之间将完成下列9个步骤：

（1）首先，在浏览器里输入网址：edu.idealsee.com



（2）浏览器根据域名解析IP地址：



**浏览器根据访问的域名找到其IP地址。DNS查找过程如下：**

1）浏览器缓存：浏览器会缓存DNS记录一段时间。 但操作系统没有告诉浏览器储存DNS记录的时间，这样不同浏览器会储存个自固定的一个时间（2分钟到30分钟不等）。

2）系统缓存：如果在浏览器缓存里没有找到需要的域名，浏览器会做一个系统调用（windows里是gethostbyname），这样便可获得系统缓存中的记录。

3）路由器缓存：如果系统缓存也没找到需要的域名，则会向路由器发送查询请求，它一般会有自己的DNS缓存。

4）ISP DNS缓存：如果依然没找到需要的域名，则最后要查的就是ISP缓存DNS的服务器。在这里一般都能找到相应的缓存记录。

（3）建立TCP连接

在HTTP工作开始之前，Web浏览器首先要通过网络与Web服务器建立连接，该连接是通过TCP来完成的，该协议与IP协议共同构建Internet，即著名的TCP/IP协议族，因此Internet又被称作是TCP/IP网络。HTTP是比TCP更高层次的应用层协议，根据规则，只有低层协议建立之后才能，才能进行更层协议的连接，因此，首先要建立TCP连接，一般承载http请求的TCP连接的端口号是80，http服务器也可配置其他端口。

（4）Web浏览器向Web服务器发送请求命令

一旦建立了TCP连接，Web浏览器就会向Web服务器发送请求命令

（5）Web浏览器发送请求头信息

浏览器发送其请求命令之后，还要以头信息的形式向Web服务器发送一些别的信息，之后浏览器发送了一空白行来通知服务器，它已经结束了该头信息的发送。

（6）Web服务器应答

客户机向服务器发出请求后，服务器会客户机回送应答，HTTP/1.1 200 OK

应答的第一部分是协议的版本号和应答状态码

（7）Web服务器发送应答头信息

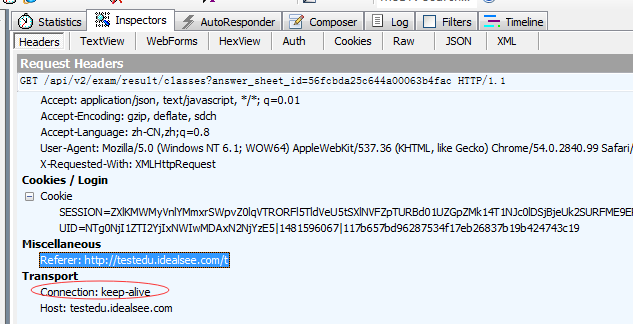
正如客户端会随同请求发送关于自身的信息一样，服务器也会随同应答向用户发送关于它自己的数据及被请求的文档。

（8）Web服务器向浏览器发送数据

Web服务器向浏览器发送头信息后，它会发送一个空白行来表示头信息的发送到此为结束，接着，它就以Content-Type应答头信息所描述的格式发送用户所请求的实际数据

（9）Web服务器关闭TCP连接

一般情况下，一旦Web服务器向浏览器发送了请求数据，它就要关闭TCP连接，然后如果浏览器或者服务器在其头信息加入了这行代码：Connection:keep-alive；TCP连接在发送后将仍然保持打开状态，于是，浏览器可以继续通过相同的连接发送请求。保持连接节省了为每个请求建立新连接所需的时间，还节约了网络带宽。

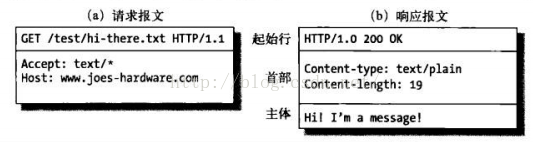


### 3、HTTP请求报文

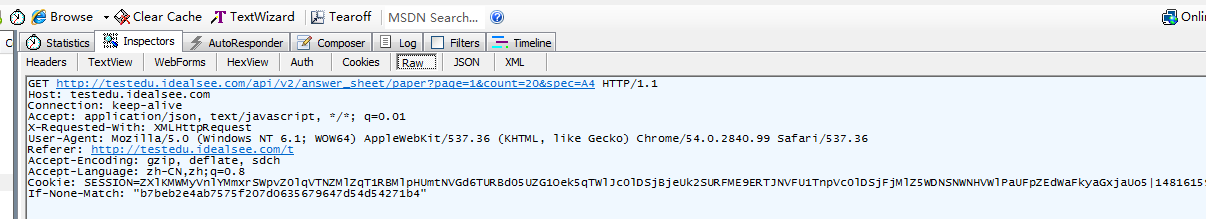
http由两部分构成，即请求和响应，请求由三部分构成：分别是：请求行、消息报头、请求正文；HTTP响应也是由三个部分组成，分别是：状态行、消息报头、响应正文。

当浏览器向Web服务器发出请求时，它向服务器传递了一个数据块，也就是请求信息，HTTP请求信息由3部分组成：请求方法URI协议/版本；请求头(Request Header)；请求正文；HTTP应答与请求相似，也是由3部分组成：协议状态版本代码描述；响应头(Response Header)；响应正文。

HTTP报文的类型：请求报文（request message）、响应报文（response message）;请求报文会向Web服务器请求一个动作，响应报文会将请求的结果返回给客户端。



**请求例子**：



（1）请求方法URI协议/版本:

请求的第一行是“方法URL议/版本”：

GET <http://testedu.idealsee.com/api/v2/answer_sheet/paper?page=1&count=20&spec=A4> HTTP/1.1

以上代码中“GET”代表请求方法，“<http://testedu.idealsee.com/api/v2/answer_sheet/paper?page=1&count=20&spec=A4>”表示URI，“HTTP/1.1”代表协议和协议的版本。

HTTP请求可以使用多种请求方法。例如：HTTP1.1支持7种请求方法：GET、POST、HEAD、OPTIONS、PUT、DELETE和TARCE。在Internet应用中，最常用的方法是GET和POST。

（2）请求头(Request Header):

请求头包含许多有关的客户端环境和请求正文的有用信息。例如，请求头可以声明浏览器所用的语言，请求正文的长度等。

Host: testedu.idealsee.com

Connection: keep-alive（http1.1中默认的持续连接）

Accept: application/json, text/javascript, \*/\*; q=0.01

X-Requested-With: XMLHttpRequest

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/54.0.2840.99 Safari/537.36

Referer: <http://testedu.idealsee.com/t>

Accept-Encoding: gzip, deflate, sdch

Accept-Language: zh-CN,zh;q=0.8

Cookie:

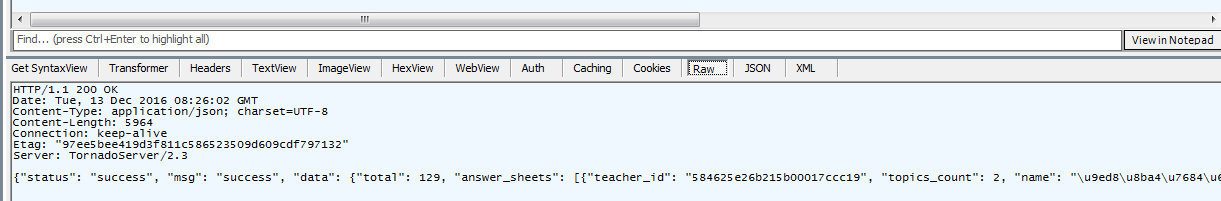
（3）请求正文：

请求头和请求正文之间是一个空行，这个行非常重要，它表示请求头已经结束，接下来的是请求正文。请求正文中可以包含客户提交的查询字符串信息：

page=1&count=20&spec=A4

### 4、HTTP响应报文

**响应例子**：



（1）协议状态版本代码描述:

协议状态代码描述HTTP响应的第一行类似于HTTP请求的第一行，它表示通信所用的协议是HTTP1.1服务器已经成功的处理了客户端发出的请求（200表示成功）:

HTTP/1.1 200 OK

（2）响应头(Response Header)：

响应头(Response Header)响应头也和请求头一样包含许多有用的信息，例如服务器类型、日期时间、内容类型和长度等：

Date: Tue, 13 Dec 2016 08:26:02 GMT

Content-Type: application/json; charset=UTF-8

Content-Length: 5964

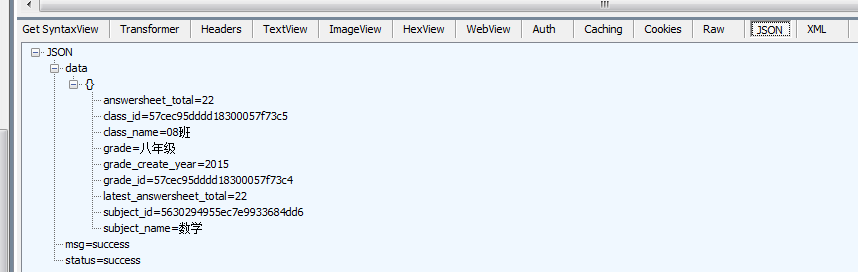
Connection: keep-alive

Etag: "97ee5bee419d3f811c586523509d609cdf797132"

Server: TornadoServer/2.3

（3）响应正文：

响应正文响应正文就是服务器返回的HTML页面。





### 5、HTTP请求方法

请求方法是请求一定的Web页面的程序或用于特定的URL，常用的HTTP方法：

GET：请求指定的页面信息，并返回实体主体。

POST：请求服务器接受所指定的文档作为对所标识的URI的新的从属实体。。

HEAD：只请求页面的首部。

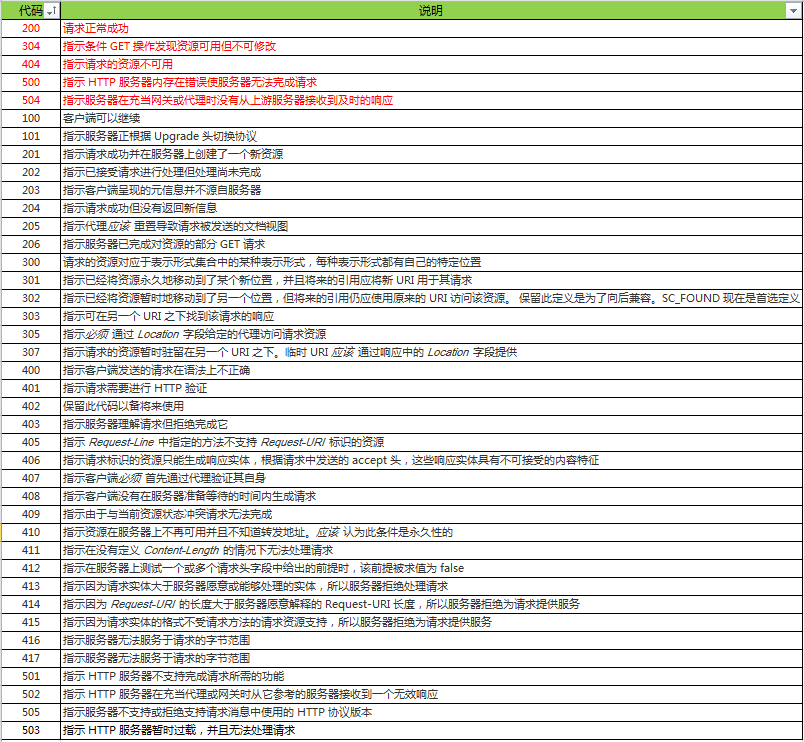
PUT：从客户端向服务器传送的数据取代指定的文档的内容。

TRACE：请求服务器在响应中的实体主体部分返回所得到的内容。

OPTIONS：允许客户端查看服务器的性能。

DELETE：请求服务器删除指定的页面。

### 6、HTTP状态码



## 三、HTTP头部信息



从图可以看出，头部信息包含三部分，通用头部，请求头部，响应头部和查询字符串参数。

### 1、通用头部

通用头域包含请求和响应消息都支持的头域。

Request URL:请求的URL地址

Request Method: 请求方法，get/post……

Status Code：状态码，200 为请求成功

Remote Address：路由地址

### 2、请求头部

1. Accept：告诉WEB服务器自己接受什么介质类型，\*/\*表示任何类型，type/\*表示该类型下的所有子类型；
2. Accept-Encoding：浏览器申明自己接收的编码方法，通常指定压缩方法，是否支持压缩，支持什么压缩方法；
3. Accept-Language：浏览器申明自己接收的语言，如果网站支持多语种的话，可以使用这个信息来决定返回什么语言的网页，zh-CN是表示中文；
4. Cookie：这是最重要的请求头信息之一；
5. Host：发送请求页面所在域；
6. If-None-Match:它和ETags常用来判断当前请求资源是否改变；
7. Connection：表示是否需要持久连接。close（告诉WEB服务器或者代理服务器，在完成本次请求的响应后，断开连接，不要等待本次连接的后续请求了）。keep-alive（告诉WEB服务器或者代理服务器，在完成本次请求的响应后，保持连接，等待本次连接的后续请求）。
8. Referer:发送请求页面URL。浏览器向WEB服务器表明自己是从哪个网页/URL获得/点击，当前请求中的网址/URL；
9. User-Agent:浏览器表明自己的身份（是哪种浏览器）。
10. X-Requested-With:在服务器端判断request来自Ajax请求(异步)还是传统请求(同步)；
11. Form：一种请求头标，给定控制用户代理的人工用户的电子邮件地址；
12. Cache-Control：浏览器应遵循的缓存机制；

no-cache（不要缓存的实体，要求现在从WEB服务器去取）；

max-age：（只接受Age值小于max-age值，并且没有过期的对象）；

max-stale：（可以接受过去的对象，但是过期时间必须小于max-stale值）；

min-fresh：（接受其新鲜生命期大于其当前Age跟min-fresh值之和的缓存对象）；

### 3、响应头部

1. Connection：是否需要持久连接；

close（连接已经关闭）；

keepalive（连接保持着，在等待本次连接的后续请求）；

Keep-Alive：如果浏览器请求保持连接，则该头部表明希望WEB服务器保持连接多长时间（秒）。例如：Keep-Alive：300；

1. Age：当代理服务器用自己缓存的实体去响应请求时，用该头部表明该实体从产生到现在经过多长时间了；
2. Accept-Ranges：WEB服务器表明自己是否接受获取其某个实体的一部分（比如文件的一部分）的请求。bytes：表示接受，none：表示不接受；
3. Cache-Control：服务器应遵循的缓存机制；

public(可以用Cached内容回应任何用户)；

private（只能用缓存内容回应先前请求该内容的那个用户）；

no-cache（可以缓存，但是只有在跟WEB服务器验证了其有效后，才能返回给客户端）；

max-age：（本响应包含的对象的过期时间）；

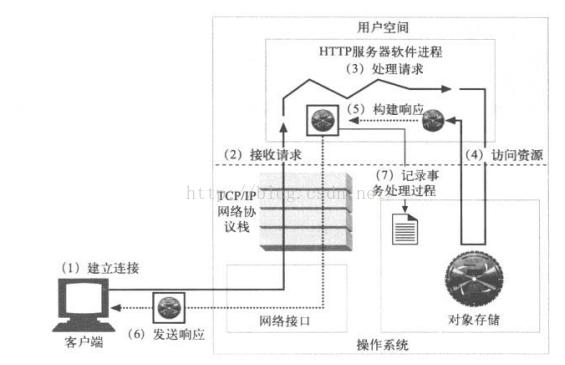
ALL:no-store（不允许缓存）；

1. Content-Encoding：WEB服务器表明自己使用了什么压缩方法（gzip，deflate）压缩响应中的对象。例如：Content-Encoding：gzip；
2. Content-Language：WEB 服务器告诉浏览器自己响应的对象的语言；
3. Content-Length：WEB 服务器告诉浏览器自己响应的对象的长度。例如：Content-Length: 26012；
4. Content-Range：WEB 服务器表明该响应包含的部分对象为整个对象的哪个部分。例如：Content-Range: bytes 21010-47021/47022；
5. Content-Type：WEB 服务器告诉浏览器自己响应的对象的类型。例如：Content-Type：application/xml；
6. Expired：WEB服务器表明该实体将在什么时候过期，对于过期了的对象，只有在跟WEB服务器验证了其有效性后，才能用来响应客户请求；
7. Last-Modified：WEB服务器认为对象的最后修改时间，比如文件的最后修改时间，动态页面的最后产生时间等等；
8. Proxy-Authenticate： 代理服务器响应浏览器，要求其提供代理身份验证信息；
9. Server:WEB服务器表明自己是什么软件及版本等信息；

## 四、服务器、代理、缓存

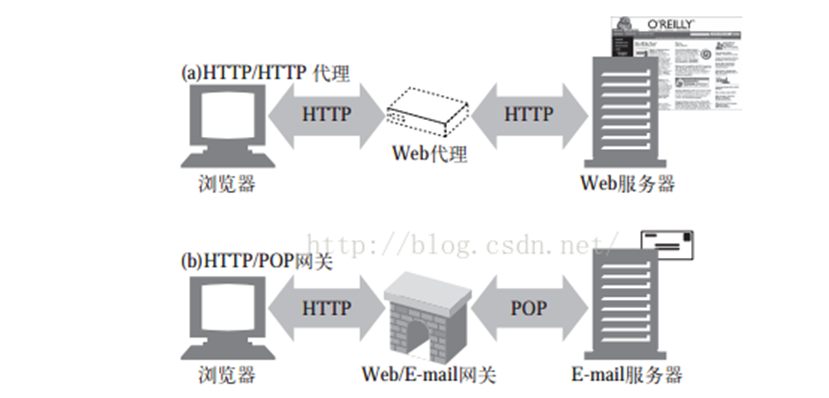
### 1、WEB服务器

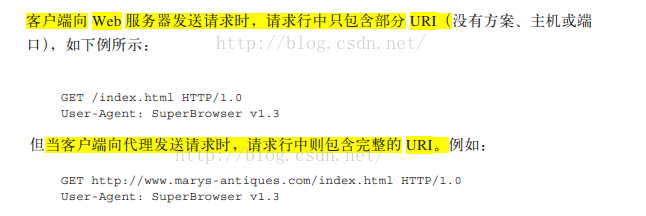
Web服务器工作的过程：  
第一步：接受客户端连接==>接受一个客户端连接，或者如果不希望与这个客户端建立连接，就将其关闭。  
第二步：接收请求报文==>从网络中读取一条HTTP请求报文。  
第三步：处理请求==>对请求报文进行解释，并采取行动。  
第四步：对资源的映射及访问==>访问报文中指定的资源。  
第五步：构建响应==>创建带有正确首部的HTTP响应报文  
第六步：发送响应==>将响应回送给客户端。  
第七步：记录日志==>将与已完成事务有关的内容记录在一个日志文件中。



### 2、代理

HTTP的代理服务器（分为公共代理和私人代理）是代表客户端完成事务的中间人，既是web服务器又是web客户端。HTTP客户端会向代理发送请求报文，代理服务器必须向web服务器一样，正确的处理请求和连接，然后返回响应。同时，代理自身要向服务器发送请求，这样，其行为就必须向正确的HTTP客户端一样，要发送请求并接受响应。

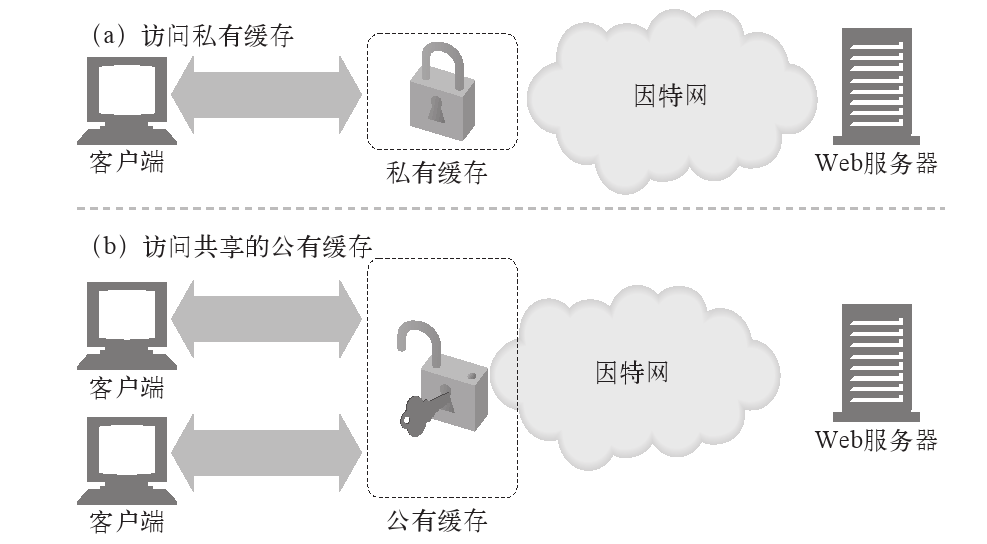




### 3、缓存

Web缓存是可以自动保存常见文档副本的HTTP设备。当Web请求抵达缓存时，若本地有“已缓存的”副本，就可以从本地存储，设备而不是原始服务器中提取这个文档。

缓存可以是单个用户专用的，也可以是数千名用户共享的。专用缓存被称为私有缓存（private cache），私有缓存是个人缓存，包含了单个用户最常用的页面；共享缓存被称为公有缓存（public cache）,公有缓存中包含某个用户团体的常用页面。



#### 3.1缓存的优缺点

优点：

1)、缓存减少冗余的数据传输；  
2)、缓存缓解了网络瓶颈的问题；  
3)、缓存降低了对原始服务器的要求；  
4)、缓存降低了距离时延。

缺点：

1)、放在缓存中的数据如果出现异常容易丢失数据；  
2)、缓存的有效周期和硬盘的空间难于估计；  
3)、如果资源发生变化，则需要取得新的资源，并在缓存中替换旧资源，有时间差；

#### 3.1GET报文缓存步骤

1）接收--缓存从网络中读取抵达的请求报文；  
2）解析--缓存对报文进行解析，提取出URL和各种首部；  
3）查询--缓存查看是否有本地副本可用，若无，就获取一份副本并将其保存在本地；  
4）新鲜度检测--缓存查看已缓存副本是否足够新鲜，若否，则询问服务器是否有任何更新；  
5）创建响应--缓存会用新的首部和已缓存的主体来构建一条响应报文；  
6）发送--缓存通过网络将响应发回给客户端；  
7）日志--缓存可选地创建一个日志文件条目来描述该事务；

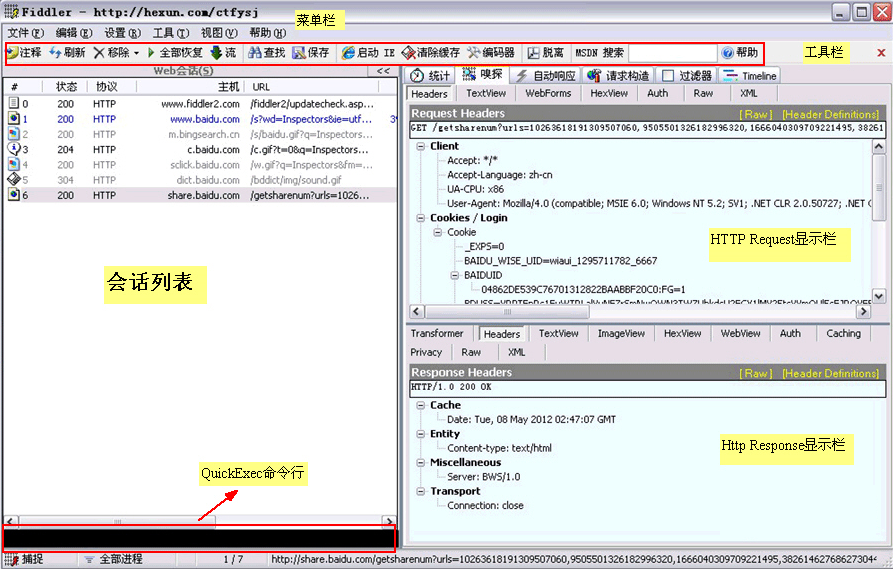
## 五、抓包工具

### 1、fiddler抓包

一款免费且功能强大的数据包抓取软件。它通过代理的方式获取程序http通讯的数据，可以用其检测网页和服务器的交互情况，能够记录所有客户端和服务器间的http请求，支持监视、设置断点、甚至修改输入输出数据等功能。

1. fiddler的下载与安装（本次内容介绍中使用fiddler4）；
2. fiddler主界面介绍：

软件窗口大体可以分为六大部分：菜单栏、工具栏、会话列表、命令行工具栏、HTTP Request显示栏、Http Response显示栏。让你在捕获数据的时候一目了然。

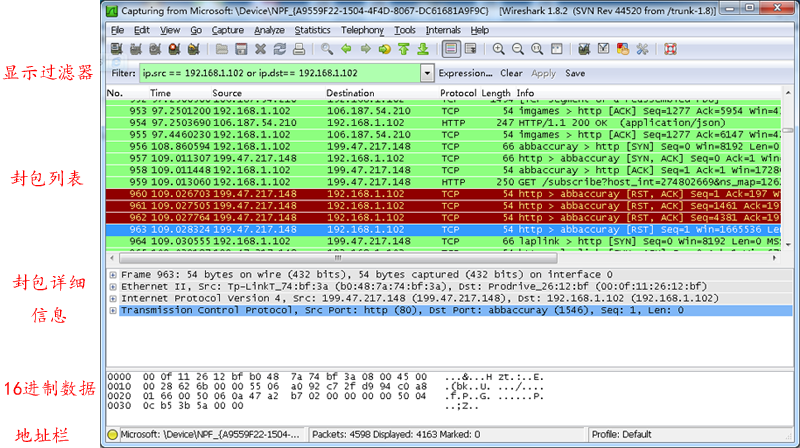


需要注意的是#号列中的图标，每种图标代表不同的相应类型，具体的类型包括：



### 2、wireshark抓包

Wireshark是世界上最流行的网络分析工具。这个强大的工具可以捕捉网络中的数据，并为用户提供关于网络和上层协议的各种信息。



**WireShark主要分为以下几个界面：**

1.Display Filter(显示过滤器)，用于过滤；

2.Packet List Pane(封包列表)，显示捕获到的封包， 有源地址和目标地址，端口号。代表颜色不同；

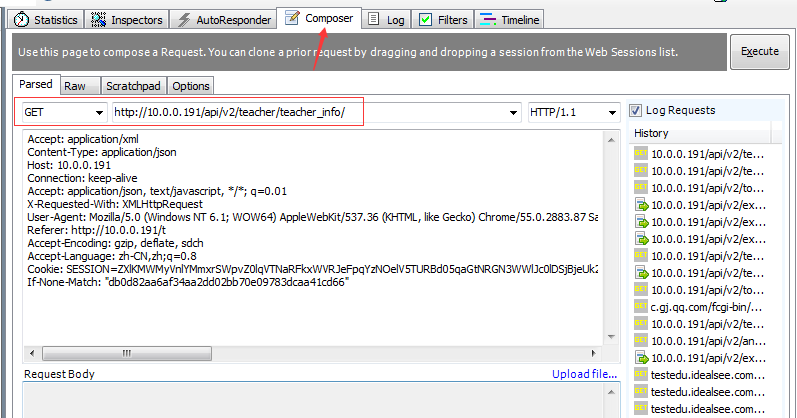
3.Packet Details Pane(封包详细信息), 显示封包中的字段；

4.Dissector Pane(16进制数据)；

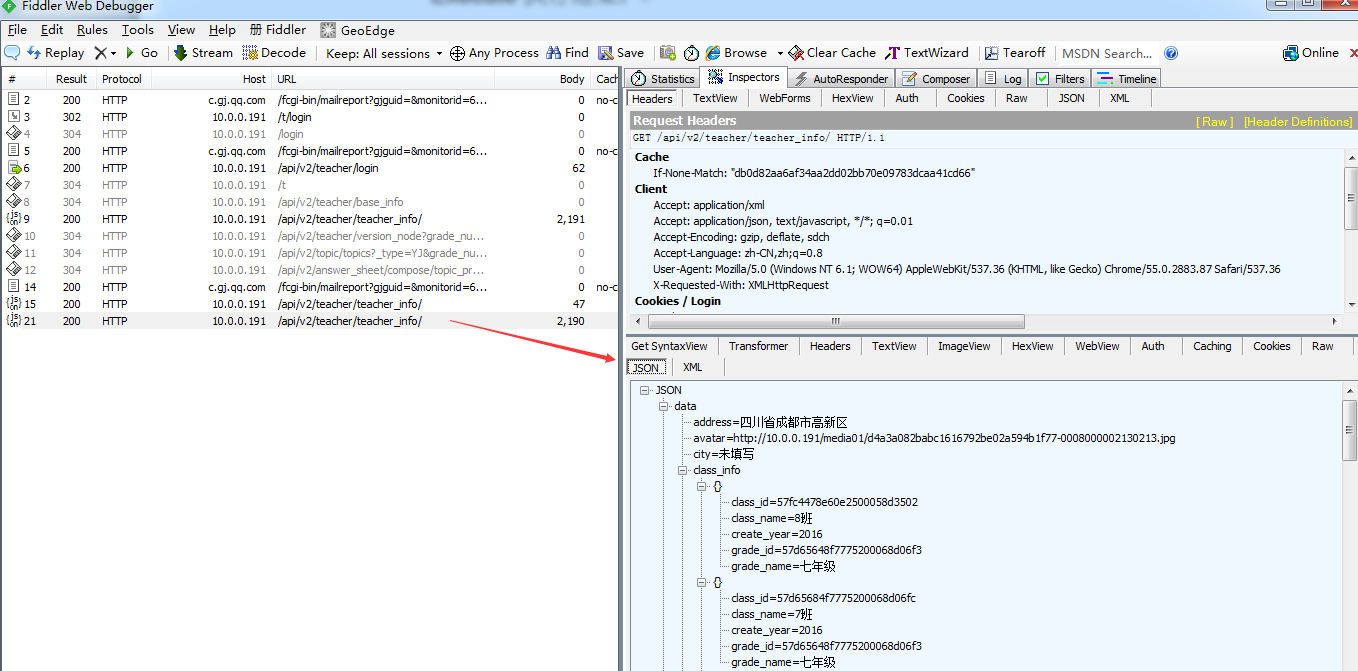
5.Miscellanous(地址栏，杂项)；

## 六、[使用Fiddler测试WebApi接口](http://www.cnblogs.com/weixing/p/5254836.html)

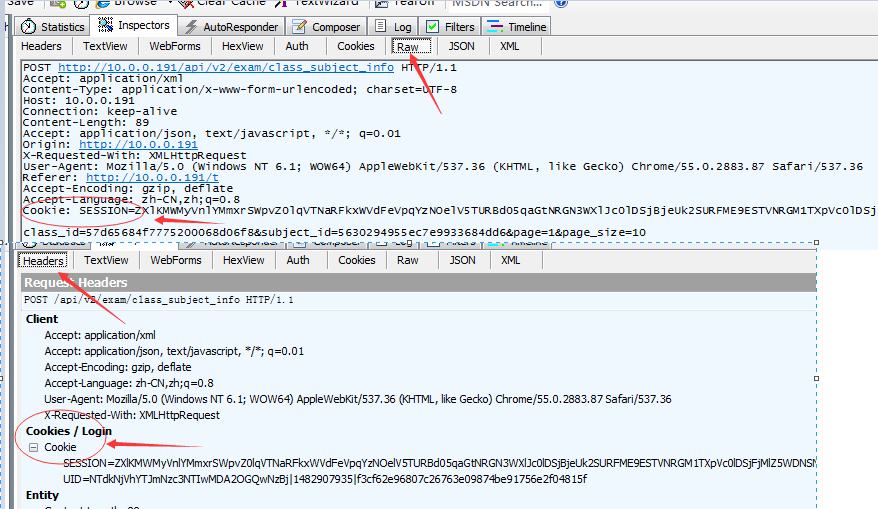
1. **WebAPI get方法**



Execute执行结果：



注意：get请求url拼装以?结束紧跟参数字段和值，多个参数以&分隔。请求需要带cookie信息，也因此登录信息才会被带入，获取方法如下



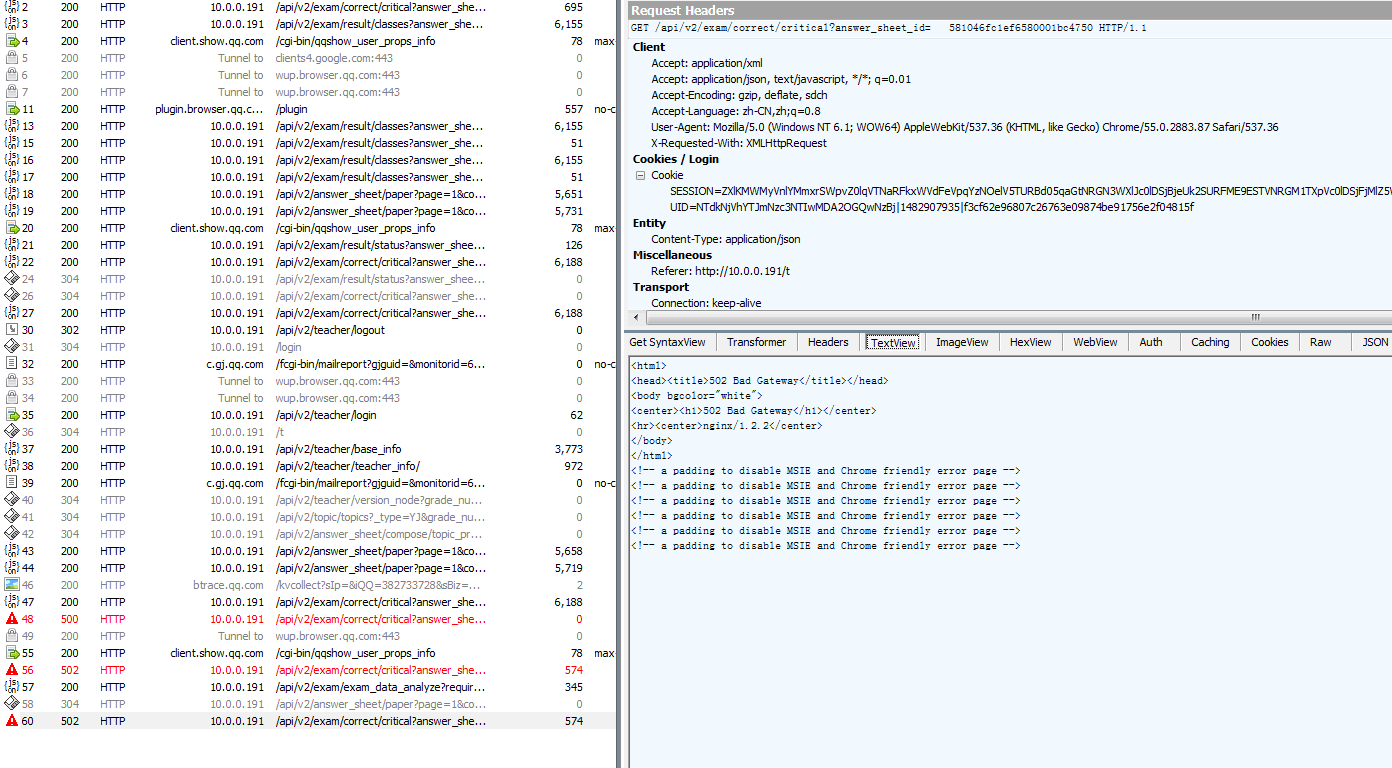
如果是带了参数的，以教育项目为例，通过修改参数来验证用户B访问用户A的试卷是否做了校验：如下

用户A：[liaochangjian@idealsee.cn](mailto:liaochangjian@idealsee.cn)

用户B：1010547853@qq.com

用户A中存在A3试卷ID：581046fc1ef6580001bc4750

带用户B的登录信息，修改A3试卷ID为581046fc1ef6580001bc4750，此时服务器返回502，代表做了校验



可以在地址后面跟“' and '1'='1';”检测sql注入

1. **WebAPI post方法**

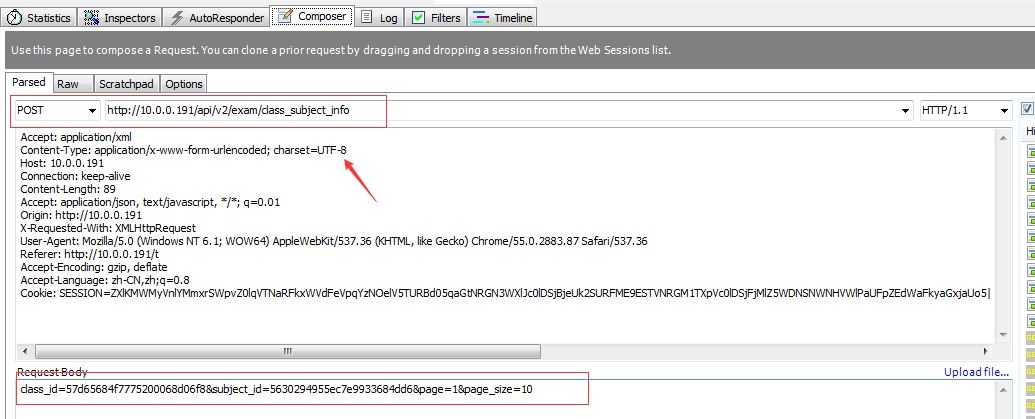
在请求头加上：

Accept:application/xml

Content-Type:application/json;charset=UTF-8

Content-Type: application/x-www-form-urlencoded;charset=UTF-8

Content-Type取值可以通过抓包来查看，教育统计分析为例；



## 参考资料

1.《HTTP权威指南》

2.MAC、IP头、TCP头、UDP头帧格式、详解“http://blog.sina.com.cn/s/blog\_692533700101bffw.html”

3.[wireshark抓包分析TCPIP协议的握手](http://blog.csdn.net/xluren/article/details/38535565)“http://blog.csdn.net/xluren/article/details/38535565”

4.浅析HTTPS中间人攻击与证书校验“http://www.2cto.com/article/201607/523509.html”

5.http://www.doc88.com/p-3744316603048.html