一、HTTP 协议简述

HTTP 是一个客户端和服务器端请求和应答的标准（TCP）。客户端是终端用户，服务器端是网站。通过使用 Web 浏览器、网络爬虫或者其它的工具，客户端发起一个到服务器上指定端口（默认端口为 80）的 HTTP 请求。（我们称这个客户端）叫用户代理（user agent）。应答的服务器上存储着（一些）资源，比如 HTML 文件和图像。（我们称）这个应答服务器为源服务器（origin server）。在用户代理和源服务器中间可能存在多个中间层，比如代理，网关，或者隧道（tunnels）。尽管 TCP/IP 协议是互联网上最流行的应用，HTTP 协议并没有规定必须使用它和（基于）它支持的层。 事实上，HTTP 可以在任何其他互联网协议上，或者在其他网络上实现。HTTP 只假定（其下层协议提供）可靠的传输，任何能够提供这种保证的协议都可以被其使用。

二、HTTP 协议通信过程

当我们在浏览器的地址栏输入“www.baidu.com”然后按回车，这之后发生了什么事，我们直接看到的是打开了对应的网页，那么内部客户端和服务端是如何通信的呢？

1、URL 自动解析

HTTP URL 包含了用于查找某个资源的足够信息，基本格式如下：HTTP://host[“:”port] [abs_path]，其中 HTTP 表示桶盖 HTTP 协议来定位网络资源；host 表示合法的主机域名或 IP 地址，port 指定一个端口号，缺省 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。

例如：输入 www.163.com;浏览器会自动转换成：HTTP://www.163.com/

2、获取 IP，建立 TCP 连接

浏览器地址栏中输入"HTTP://www.xxx.com/"并提交之后，首先它会在 DNS 本地缓存表中查找，如果有则直接告诉 IP 地址。如果没有则要求网关 DNS 进行查找，如此下去，找到对应的 IP 后，则返回会给浏览器。

当获取 IP 之后，就开始与所请求的 Tcp 建立三次握手连接，连接建立后，就向服务器发出 HTTP 请求。

3、客户端浏览器向服务器发出 HTTP 请求

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，接着以头信息的形式向 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。

4、Web 服务器应答，并向浏览器发送数据

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，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。

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

5、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。

三、实例分析 HTTP 通信

先 介绍一个工具，HTTP Analyzer ，为一款实时分析 HTTP/HTTPS 数据流的工具。它可以实时捕捉 HTTP/HTTPS 协议数据，可以显示许多信息（包括：文件头、内容、Cookie、查询字符窜、提交的数据、

重定向的 URL 地址），可以提供缓冲区信息、清理对话内容、 HTTP 状态信息和其他过滤选项。同时还是一个非常有用的分析、调试和诊断的开发工具。

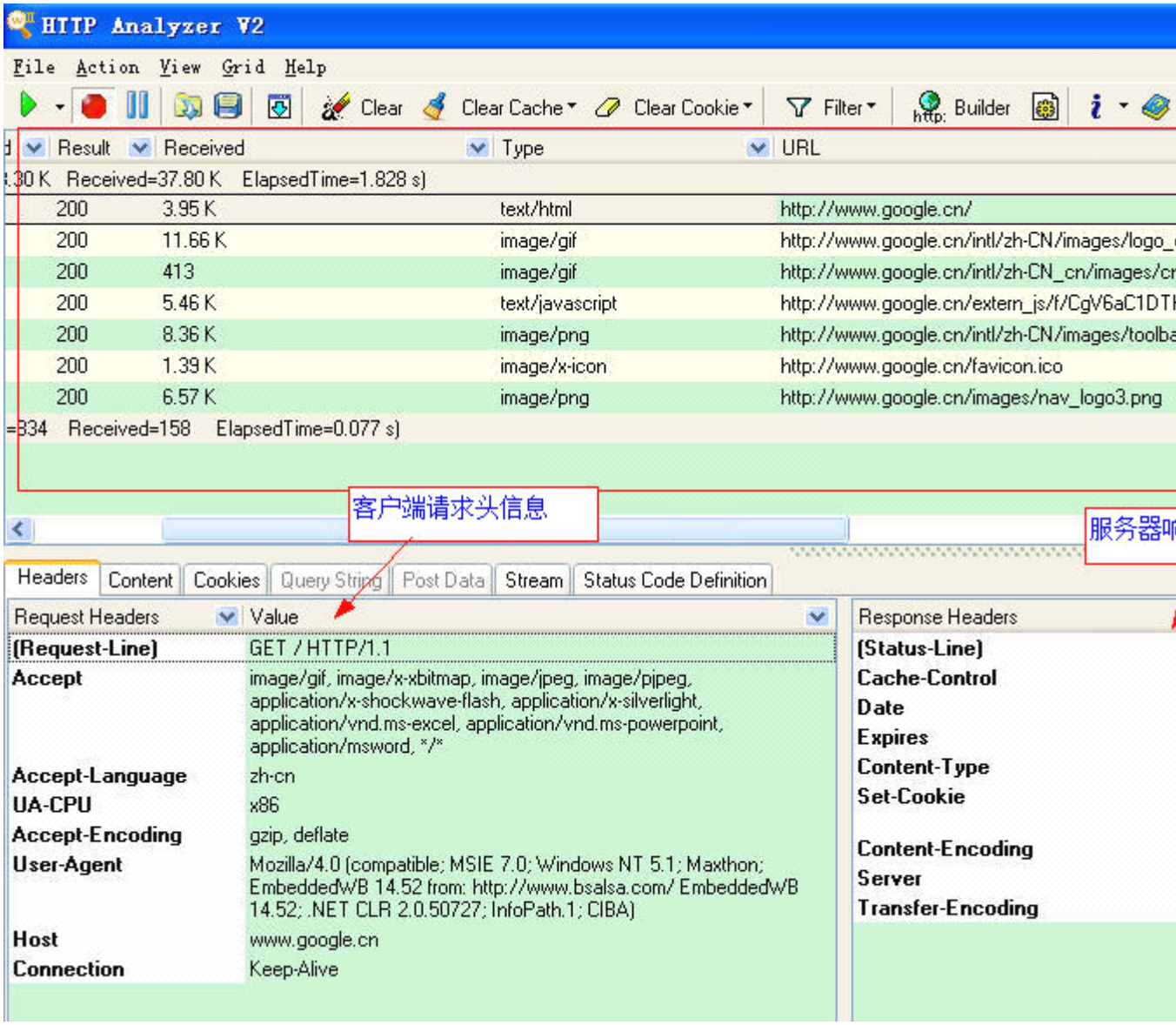
下面我们访问 http://www.google.cn/ ，HTTP analyzer 将抓包来分析访问浏览器和服务器通信的过程。

1、 运行 HTTP Analyzer，选择菜单 Action—start 开始抓包；

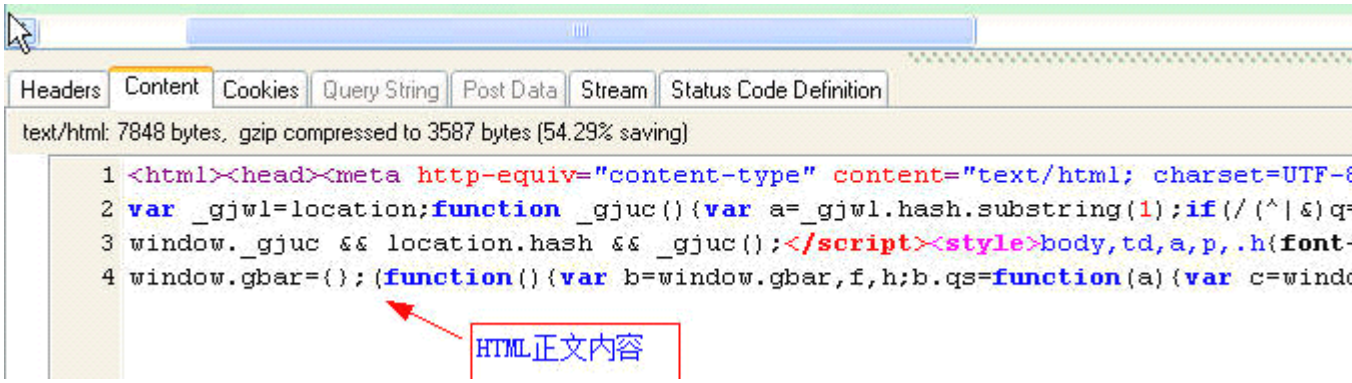
2、 浏览器中输入 http://www.google.cn/，网页打开后，在 HTTP Analyzer 中选择 Action—stop 停止抓包;

工具已经详细列出了访问的数据包信息。通过截图见到了解下抓包信息

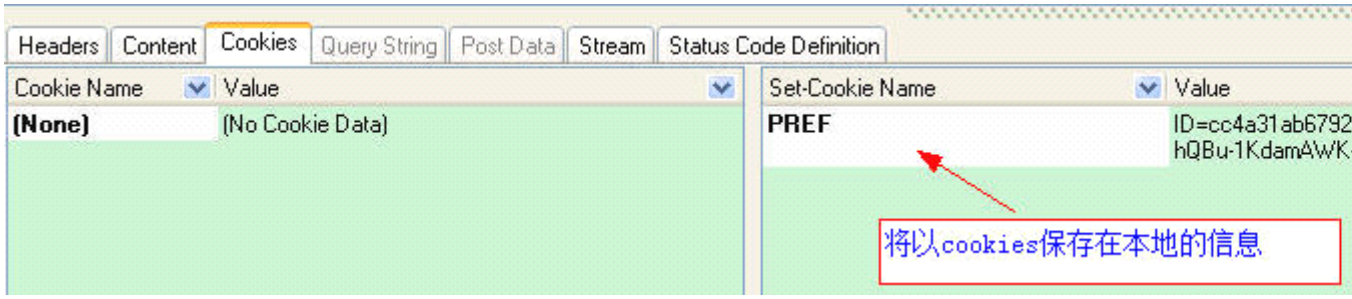
- 抓包结果和文件头信息（下图）



- 一次请求的 html 正文内容（下图）



- 本次请求是否存在 cookies 信息（下图）



- 一次请求的整个数据包信息，包括头信息和正文(下图)。

你 会发现浏览器中只点击了一个超级链接，却发送了多个数据包。那是因为,我们请求的网页文件中有很多图片、音乐、电影等信息时，服务器返回的信息中并不直接 包含图片数据，而只是保存该图片的链接，当浏览器进行解释的时候，遇到图片的 url 时,才向服务器发出对图片的请求信息。

下面我们来详细分析 HTTP 的请求和响应信息：

1）HTTP 请求消息，当客户端和服务端建立 TCP 连接后，客户端就会向服务器发送一个请求信息， 如：

[1] GET / HTTP/1.1

[2] Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/x-silverlight, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword, */* 客户端可识别的内容类型列表。

[3] Accept-Language: zh-cn 客户端所能解释的语言：简体中文

[4] UA-CPU: x86

[5] Accept-Encoding: gzip, deflate 客户端可以解释的类型

[6] User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Maxthon; EmbeddedWB 14.52 from: http://www.bsalsa.com/ EmbeddedWB 14.52; .NET CLR 2.0.50727; InfoPath.1; CIBA) 客户端浏览器型号

[7] Host: http://www.google.cn/ 提交请求页面

[8] Connection: Keep-Alive TCP 连接保持打开

[9]

该请求信息主要由4部分组成：

- 请求方法 URI 协议/版本：以上代码第[1]行“GET”表示请求方法，，“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，最常用的为 GET 和 POST 方法
- 请求头：[2]-[8]行，包含许多有关客户端环境和请求正文的有用信息。
- 空行 ：[9] 请求头和请求正文之间是一个空行，这个行非常重要，表示请求头已经结束，接下来是正文，这个行非常重要，它表示请求头已经结束，接下来是请求正文。
- 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，如用户名和密码等。这里没有。

这里有一点值得说明的是:请求方法中的 GET 和 POST 方法；

GET 方法是默认的 HTTP 请求方法，我们日常用 GET 方法来提交表单数据，然而用 GET 方法提交的表单数据只经过了简单的编码，同时它将作为 URL 的一部分向 Web 服务器发送，因此，如果使用 GET 方法来提交表单数据就存在着安全隐患上，同时这个 URL 长度还有限制，不允许超过1k。

POST 方法是 GET 方法的一个替代方法，它主要是向 Web 服务器提交表单数据，尤其是大批量的数据。POST 方法克服了 GET 方法的一些缺点。通过 POST 方法提交表单数据时，数据不是作为 URL 请求的一部分而是作为标准数据传送给 Web 服务器，这就克服了 GET 方法中的信息无法保密和数据量太小的缺点。因此，出于安全的考虑以及对用户隐私的尊重，通常表单提交时采用 POST 方法。

2）HTTP 响应消息，响应跟请求类似，如：

[1]HTTP/1.1 200 OK
[2]Cache-Control: private, max-age=0
[3]Date: Fri, 27 Feb 2009 07:53:36 GMT
[4]Expires: -1
[5]Content-Type: text/html; charset=UTF-8
[6]Set-Cookie:
PREF=ID=cc4a31ab6792ef2c:NW=1:TM=1235721216:LM=1235721216:S=q1hQBu-1KdamAWK-; expires=Sun, 27-Feb-2011 07:53:36 GMT; path=/; domain=.google.cn
[7]Content-Encoding: gzip
[8]Server: gws
[9]Transfer-Encoding: chunked
[10]

[11]ddc

该响应信息也以对应的4部分组成：

- 协议状态描述，HTTP/1.1表示协议版本，200 OK 表示服务器已经成功处理了客户端发出的请求。200表示 HTTP 的应答码成功。HTTP 应答码由3位数字构成，其中首位数字定义了应答码的类型：

  1XX－信息类(Information),表示收到 Web 浏览器请求，正在进一步的处理中

  2XX－成功类（Successful）,表示用户请求被正确接收，理解和处理例如：200 OK

  3XX-重定向类(Redirection),表示请求没有成功，客户必须采取进一步的动作。

  4XX-客户端错误(Client Error)，表示客户端提交的请求有错误 例如：404 NOT Found，意味着请求中所引用的文档不存在。

  5XX-服务器错误(Server Error)表示服务器不能完成对请求的处理：如 500

- 响应头：跟请求头一样，它指出服务器的功能，标识出响应数据的细节。
- 空行：也是属于响应头和响应正文之间必须存在的一个空行，表示响应头结束，接下来是响应正文
- 响应正文：也就是服务器返回的网页内容。

根据上文的描述，再结合工具实际验证一回，相信应该能对 HTTP 协议和其通信流程有个大致的了解。