1. **HTTP协议简介**
2. **http协议**

HyperText Transfer Protocol，超文本传输协议。

用于浏览器和web服务器直接信息的传输，一般是将超文本标记语言(HTML)文档从Web服务器传送到客户端的浏览器。

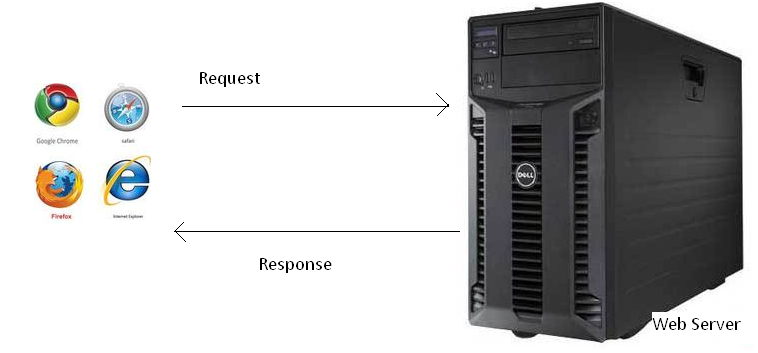
目前主流的http是HTPP 1.1版本。协议的principle可以查阅rfc 2616定义：

<https://www.w3.org/Protocols/rfc2616/rfc2616.html>

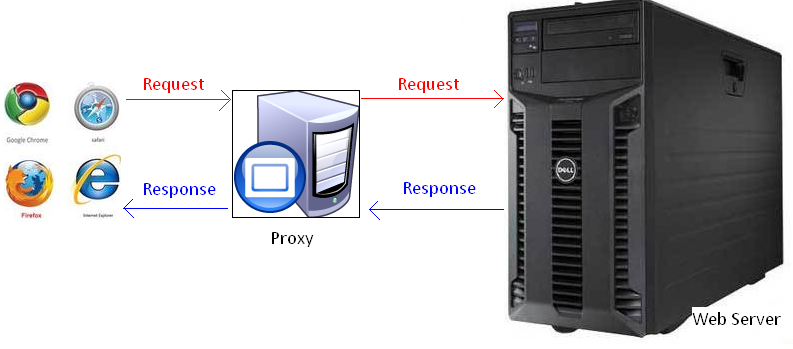
HTML：HyperText Mark-up Language，超文本标记语言，一般浏览器里面显示的内容就是html文件。

1. **Web服务器，浏览器，代理服务器**

实际在浏览器里我们输入URL后，浏览器会给Web服务器发送一个Request，Web服务器接到Request后进行处理，生成相应的Response，然后发送给浏览器， 浏览器解析Response中的HTML,这样我们就看到了网页，过程如下图所示



我们的Request 有可能是经过了代理服务器，最后才到达Web服务器的，如：



1. **URL介绍**

Uniform Resource Locator，地址用于描述一个网络上的资源，

基本格式如下：

scheme:[//[user:password@]host[:port]][/]path[?query][#fragment]

一般只用到下面几个字段。

scheme: 指定低层使用的协议(例如：http, https, ftp)

host : HTTP服务器的IP地址或者域名

port: HTTP服务器的默认端口是80，这种情况下端口号可以省略。如果使用了别的端口，必须指明。

path: 访问资源的路径

query: 是发送给http服务器的数据，可以理解为传给资源的参数。在”?”符号后面，一

般是key1=value1&key2=value2的形式。中间用&隔开。

如：

<http://172.21.29.243/ota_upgrade.php?version=TVOS.04.15.010.03.17&device=miraclefruit>

http协议，host ip地址为172.21.29.243， 资源文件为ota\_upgrade.php，version和device为传递的参数。

1. **http消息**

http消息分为2种，一种是client端给server端发送的Request 消息；另外一种是server给client发送的Response 消息。

1. **Request 消息**

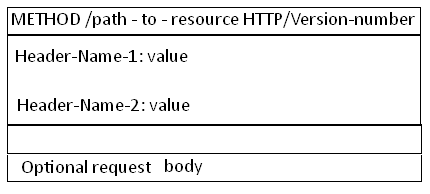
Request 消息分为3部分，每个字段直接用CRLF(\r+\n)隔开。

第一部分叫请求行，Request line，

第二部分叫http header。

第三部分是body。header和body之间有个空行，body是可选的，可以拿掉。

结构如下图:



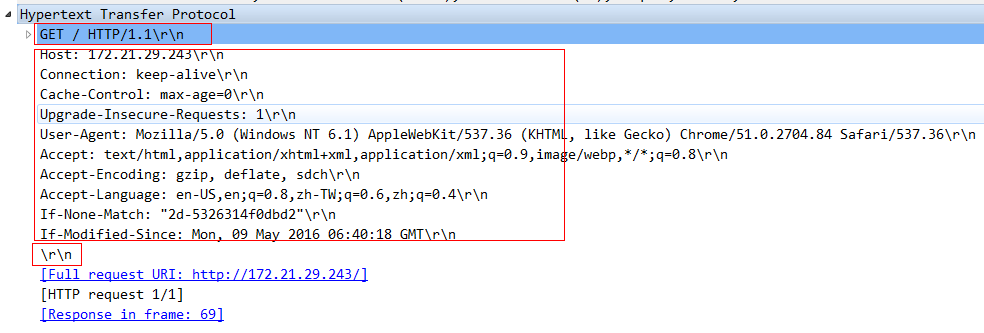
Method表示请求方法，比如"POST"，"GET"等。

Path-to-resoure表示请求的资源，

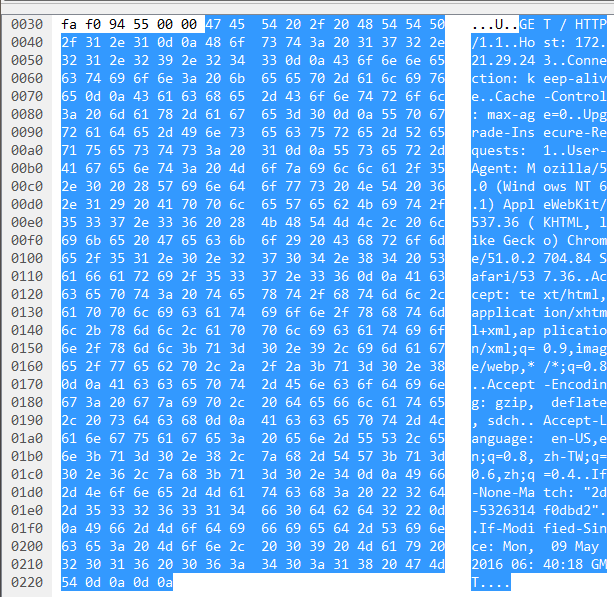
Http/version-number 表示HTTP协议的版本号，如HTTP/1.1

当使用的是"GET" 方法的时候，body是为空的。

一个用wireshark抓到的http包如下：



对应的二进制数据：



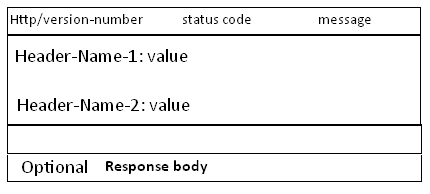
1. **Response消息**

和Request消息的结构基本一样。 同样也分为三部分，

第一部分叫Response line,

第二部分叫Response header，

第三部分是body， header和body之间也有个空行，结构如下图：



HTTP/version-number：表示HTTP协议的版本号，如HTTP/1.1

status code：表示状态码。

　HTTP/1.1中定义了5类状态码， 状态码由三位数字组成，第一个数字定义了响应的类别

　1XX 提示信息 - 表示请求已被成功接收，继续处理

　2XX 成功 - 表示请求已被成功接收，理解，接受

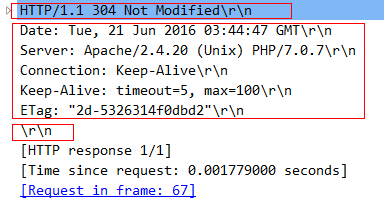
　3XX 重定向 - 要完成请求必须进行更进一步的处理

　4XX 客户端错误 - 请求有语法错误或请求无法实现

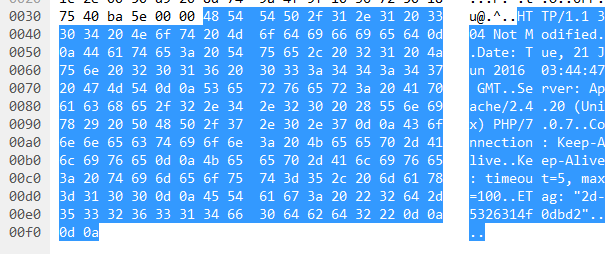
　5XX 服务器端错误 - 服务器未能实现合法的请求

message表示提示信息。

一个用wireshark抓到的HTTP Response包：



对应的二进制数据：



1. **Method请求方法**

client端发出的request method主要有下面几种：

1. GET

从指定的url上获取内容

1. HEAD

从指定的url上获取header内容(类似Get方式)

The HEAD method asks for a response identical to that of a GET request, but without the response body. This is useful for retrieving meta-information written in response headers, without having to transport the entire content.

1. OPTIONS

获取指定url中能接收的请求方法

1. TRACE

Allows a programmer to see how the client's message is modified as it passes through a series of proxy servers. The recipient of a TRACE method echoes the HTTP request headers back to the client

1. CONNECT

连接指定频段。当客户端需要通过代理服务器连接HTTPS服务器时用到

1. POST

提交body中的内容给服务器中指定的url中

1. PUT

将body上传至服务器指定url处

1. DELETE

在指定url处删除资源

1. PATCH

The PATCH method applies partial modifications to a resource

HTTP 1.0里面定义了GET/HEAD/POST。

HTTP 1.1里面增加了OPTIONS/TRACE/CONNECT/PUT/DELETE/PATCH

Get和Post方法的区别：

1. GET提交的数据会放在URL之后，以?分割URL和传输数据，参数之间以&相连，如EditPosts.aspx?name=test1&id=123456.  POST方法是把提交的数据放在HTTP包的Body中.
2. GET提交的数据大小有限制（因为浏览器对URL的长度有限制），而POST方法提交的数据没有限制.
3. GET方式提交数据，会带来安全问题，比如一个登录页面，通过GET方式提交数据时，用户名和密码将出现在URL上，如果页面可以被缓存或者其他人可以访问这台机器，就可以从历史记录获得该用户的账号和密码.
4. **Header Field Definition**

Header Field里面可以有多个key-value值，可以理解为用来设定一些property。

主要的value有下面几个。

1. **If-Modified-Since**

把浏览器端缓存页面的最后修改时间发送到服务器去，服务器会把这个时间与服务器上实际文件的最后修改时间进行对比。如果时间一致，那么返回304，客户端就直接使用本地缓存文件。如果时间不一致，就会返回200和新的文件内容。客户端接到之后，会丢弃旧文件，把新文件缓存起来，并显示在浏览器中。例如：If-Modified-Since: Thu, 09 Feb 2012 09:07:57 GMT

1. **If-None-Match**

If-None-Match和ETag一起工作，工作原理是在HTTP Response中添加ETag信息。 当用户再次请求该资源时，将在HTTP Request 中加入If-None-Match信息(ETag的值)。如果服务器验证资源的ETag没有改变（该资源没有更新），将返回一个304状态告诉客户端使用本地缓存文件。否则将返回200状态和新的资源和Etag.  使用这样的机制将提高网站的性能

　　例如: If-None-Match: "03f2b33c0bfcc1:0"

1. **Pragma**

防止页面被缓存， 在HTTP/1.1版本中，它和Cache-Control:no-cache作用一模一样

　　Pargma只有一个用法， 例如： Pragma: no-cache

　　注意: 在HTTP/1.0版本中，只实现了Pragema:no-cache, 没有实现Cache-Control

1. **Cache-Control**

这个是非常重要的规则。 这个用来指定Response-Request遵循的缓存机制。各个指令含义如下

　　Cache-Control:Public   可以被任何缓存所缓存（）

　　Cache-Control:Private     内容只缓存到私有缓存中

　　Cache-Control:no-cache  所有内容都不会被缓存

　　还有其他的一些用法， 我没搞懂其中的意思， 请大家参考其他的资料

1. **Accept**

 浏览器端可以接受的媒体类型,

　　例如：  Accept: text/html  代表浏览器可以接受服务器回发的类型为 text/html  也就是我们常说的html文档,

　　如果服务器无法返回text/html类型的数据，服务器应该返回一个406错误(non acceptable)

　　通配符 \* 代表任意类型

　　例如  Accept: \*/\*  代表浏览器可以处理所有类型，(一般浏览器发给服务器都是发这个)

1. **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（gzip，deflate），（注意：这不是只字符编码）;

　　例如： Accept-Encoding: gzip, deflate

1. **Accept-Language**

浏览器申明自己接收的语言。

　　语言跟字符集的区别：中文是语言，中文有多种字符集，比如big5，gb2312，gbk等等；

　　例如： Accept-Language: en-us

1. **User-Agent**

告诉HTTP服务器， 客户端使用的操作系统和浏览器的名称和版本.

我们上网登陆论坛的时候，往往会看到一些欢迎信息，其中列出了你的操作系统的名称和版本，你所使用的浏览器的名称和版本，这往往让很多人感到很神奇，实际上，服务器应用程序就是从User-Agent这个请求报头域中获取到这些信息User-Agent请求报头域允许客户端将它的操作系统、浏览器和其它属性告诉服务器。

　　例如： User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; CIBA; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729; .NET4.0C; InfoPath.2; .NET4.0E)

1. **Accept-Charset**

浏览器申明自己接收的字符集，这就是本文前面介绍的各种字符集和字符编码，如gb2312，utf-8（通常我们说Charset包括了相应的字符编码方案）；

1. **Content-Length**

发送给HTTP服务器数据的长度。

例如： Content-Length: 38

1. **Content-Type**

指定entity-body media type。

The Content-Typeentity-header field indicates the media type of the entity-body sent to the recipient or, in the case of the HEADmethod, the media type that would have been sent had the request been a GET.

Content-Type = "Content-Type" ":" media-type

An example of the field is:

Content-Type: text/html; charset=ISO-8859-4

1. **Referer:**

　　作用： 提供了Request的上下文信息的服务器，告诉服务器我是从哪个链接过来的，比如从我主页上链接到一个朋友那里，他的服务器就能够从HTTP Referer中统计出每天有多少用户点击我主页上的链接访问他的网站。

例如: Referer:http://translate.google.cn/?hl=zh-cn&tab=wT

1. **Connection**

　　例如：　Connection: keep-alive   当一个网页打开完成后，客户端和服务器之间用于传输HTTP数据的TCP连接不会关闭，如果客户端再次访问这个服务器上的网页，会继续使用这一条已经建立的连接

例如：  Connection: close  代表一个Request完成后，客户端和服务器之间用于传输HTTP数据的TCP连接会关闭， 当客户端再次发送Request，需要重新建立TCP连接。

1. **Host**

发送请求时，该报头域是必需的

　　作用: 请求报头域主要用于指定被请求资源的Internet主机和端口号，它通常从HTTP URL中提取出来的

　　例如: 我们在浏览器中输入：http://www.guet.edu.cn/index.html

　　浏览器发送的请求消息中，就会包含Host请求报头域，如下：

　　Host：http://www.guet.edu.cn

　　此处使用缺省端口号80，若指定了端口号，则变成：Host：指定端口号