**通讯协议**

1. **通讯协议**

**谈到任何联网的协议，我们就必须要谈到OSI（网络七层协议模型），必须遵循这个协议模型，我们的手机和电脑才可以联网通信，首先来看一下OSI**

**OSI是一个开放性的通信系统互连参考模型，他是一个定义得非常好的协议规范。OSI模型有7层结构，每层都可以有几个子层。**

1. **应用层：**应用层决定了向用户提供应用服务时通信的活动，如文件传输，文件服务，电子邮件等

**协议**：TELNET，HTTP，FTP，NFS，SMTP等。

1. **表示层：**没有协议，数据格式化，代码转换，数据加密

**协议**：加密，ASCII等。

1. **会话层：**没有协议，接触或简历与别的节点的联系

**协议**：RPC，SQL等。

1. **传输层：**传输层对上层应用层，提供处于网络连接中的两台计算机之间的数据传输。提供端对端的接口，当我发大量数据时候，可能需要好长时间，例如一个视频格式的，网络会中断好多次（事实上，即使有了物理层和数据链路层，网络还是经常中断，只是中断的时间是毫秒级别的）。那么，我还须要保证传输大量文件时的准确性。于是，我要对发出去的数据进行封装。就像发快递一样，一个个地发。例如TCP，是用于发大量数据的，我发了1万个包出去，另一台电脑就要告诉我是否接受到了1万个包，如果缺了3个包，就告诉我是第1001，234，8888个包丢了，那我再发一次。这样，就能保证对方把这个视频完整接收了

**协议**：TCP，UDP，SPX。

1. **网络层：**网络层用来处理在网络上流动的数据包。数据包是网络传输的最小数据单位。该层规定了通过怎样的路径（所谓的传输路线）到达对方计算机，并把数据包传送给对方。与对方计算机之间通过多台计算机或网络设备进行传输时，网络层所起的作用就是在众多的选项内选择一条传输路线。

**协议**：IP，IPX等。

1. **数据链路层：** 定义了如何让格式化数据以进行传输，以及如何让控制对物理介质的访问。这一层通常还提供错误检测和纠正，以确保数据的可靠传输，数据以不同的物理介质进行传送

**协议**：ATM，FDDI等。

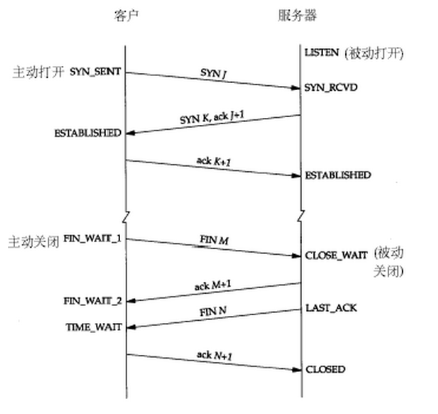
1. **物理层：**以二进制数据形式在物理媒体上传输数据（比特流），主要定义物理设备标准，如网线的接口类型、光纤的接口类型、各种传输介质的传输速率等。它的主要作用是传输比特流(就是由1、0转化为电流强弱来进行传输，到达目的地后在转化为1、0，也就是我们常说的数模转换与模数转换)。这一层的数据叫做比特。

**协议**：Rj45，802.3等。

**注:很多应用场景将其分为四层，分别为最上面的就是应用层了，这里面有http，ftp,等等我们熟悉的协议。而第二层则是传输层，著名的TCP和UDP协议就在这个层次。第三层是网络层，IP协议就在这里，它负责对数据加上IP地址和其他的数据（后面会讲到）以确定传输的目标。第四层是叫数据链路层**

1. **TCP/IP协议**

**建立起一个TCP连接需要经过“三次握手”：**



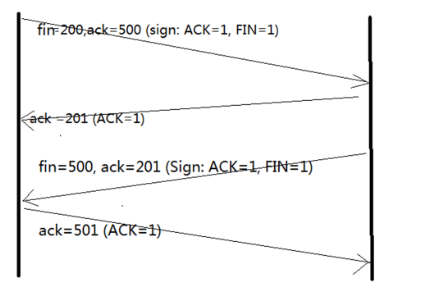
**第一次握手**：客户端发送syn包(syn=j)到服务器，并进入SYN\_SEND状态，等待服务器确认；

**第二次握手**：服务器收到syn包，必须确认客户的SYN（ack=j+1），同时自己也发送一个SYN包（syn=k），即SYN+ACK包，此时服务器进入SYN\_RECV状态；

**第三次握手：**客户端收到服务器的SYN＋ACK包，向服务器发送确认包ACK(ack=k+1)，此包发送完毕，客户端和服务器进入established状态，完成三次握手。

握 手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。理想状态下，TCP连接一旦建立，在通信双方中的任何一方主动关闭连 接之前，TCP 连接都将被一直保持下去。断开连接时服务器和客户端均可以主动发起断开TCP连接的请求，断开过程需要经过“四次挥手”（过程就不细写 了，就是服务器和客户端交互，最终确定断开）

**断开一个tcp连接的四次挥手**



（1）客户端A发送一个FIN，用来关闭客户A到服务器B的数据传送。

（2）服务器B收到这个FIN，它发回一个ACK，确认序号为收到的序号加1。和SYN一样，一个FIN将占用一个序号。

（3）服务器B关闭与客户端A的连接，发送一个FIN给客户端A。

（4）客户端A发回ACK报文确认，并将确认序号设置为收到序号加1。

**注：**

**1.TCP是底层通讯协议，定义的是数据传输和连接方式的规范**

HTTP是应用层协议，定义的是传输数据的内容的规范

HTTP协议中的数据是利用TCP协议传输的，所以支持HTTP也就一定支持TCP

**2.HTTP支持的是www服务**

而TCP/IP是协议

它是Internet国际互联网络的基础。TCP/IP是网络中使用的基本的通信协议。

TCP/IP实际上是一组协议，它包括上百个各种功能的协议，如：远程登录、文件传输和电子邮件等，而TCP协议和IP协议是保证数据完整传输的两个基本的重要协议。通常说TCP/IP是Internet协议族，而不单单是TCP和IP。

1. **http协议：**

**HTTP--Hyper Text Transfer Protocol，超文本传输协议，是一种建立在TCP上的无状态连接，整个基本的工作流程是客户端发送一个HTTP请求，说明客户端想要访问的资源和请求的动作，服务端收到请求之后，服务端开始处理请求，并根据请求做出相应的动作访问服务器资源，最后通过发送HTTP响应把结果返回给客户端。其中一个请求的开始到一个响应的结束称为事务，当一个事物结束后还会在服务端添加一条日志条目。**

* 1. **HTTP请求**

**HTTP请求由状态行、请求头、请求正文三部分组成：**

状态行：包括请求方式Method、资源路径URL、协议版本Version；

请求头：包括一些访问的域名、用户代理、Cookie等信息；

请求正文：就是HTTP请求的数据。

**现在大多数协议版本为http/1.1**

* 1. **HTTP响应**

1. **响应数据格式**

**服务器收到了客户端发来的HTTP请求后，根据HTTP请求中的动作要求，服务端做出具体的动作，将结果回应给客户端，称为HTTP响应。**

        HTTP响应由三部分组成：状态行、响应头、响应正文；

状态行：包括协议版本Version、状态码Status Code、回应短语；

响应头：包括搭建服务器的软件，发送响应的时间，回应数据的格式等信息；

响应正文：就是响应的具体数据。

**备注：我们主要关心并且能够在客户端浏览器看得到的是三位数的状态码，不同的状态码代表不同的含义**

|  |  |
| --- | --- |
| 1xx | 表示HTTP请求已经接受，继续处理请求 |
| 2xx | 表示HTTP请求已经处理完成 |
| 3xx | 表示把请求访问的URL重定向到其他目录 |
| 4xx | 表示客户端出现错误 |
| 5xx | 表示服务端出现错误 |

1. **常见状态码的含义**

        200---OK/请求已经正常处理完毕

        301---/请求永久重定向

        302---/请求临时重定向

        304---/请求被重定向到客户端本地缓存

        400---/客户端请求存在语法错误

        401---/客户端请求没有经过授权

        403---/客户端的请求被服务器拒绝，一般为客户端没有访问权限

        404---/客户端请求的URL在服务端不存在

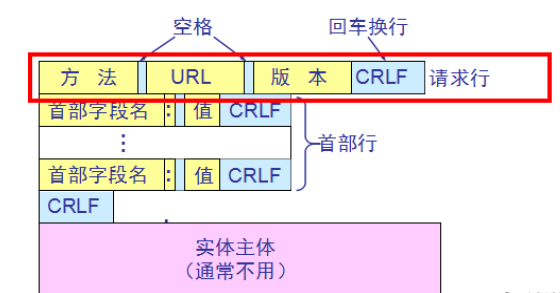
        500---/服务端永久错误

        503---/服务端发生临时错误

**三、HTTP报文格式**

        HTTP报文是HTTP应用程序之间传输的数据块，HTTP报文分为HTTP请求报文和HTTP响应报文，但是无论哪种报文，他的整体格式是类似的，大致都是由起始、首部、主体三部分组成，起始说明报文的动作，首部说明报文的属性，主体则是报文的数据。接下来具体说明。

1. **HTTP请求报文**



        请求报文的起始由请求行构成（有些资料称为状态行，名字不一样而已，都是指的一个东西），用来说明该请求想要做什么，由<Method>、<URL>、<Version> 三个字段组成，注意每个字段之间都有一个空格。

        其中<Method>字段有不同的值：

                GET   --- 访问服务器的资源

                POST  --- 向服务器发送要修改的数据

                HEAD  --- 获取服务器文档的首部

                PUT   --- 向服务器上传资源

                DELETE--- 删除服务器的资源

        <URL>字段表示服务器的资源目录定位

        <Version>字段表示使用的http协议版本

**首部部分由多个请求头（也叫首部行）构成，那些首部字段名有如下**：

                Accept     指定客户端能够接收的内容格式类型

               Accept-Language 指定客户端能够接受的语言类型

                Accept-Ecoding  指定客户端能够接受的编码类型

                User-Agent      用户代理，向服务器说明自己的操作系统、浏览器等信息

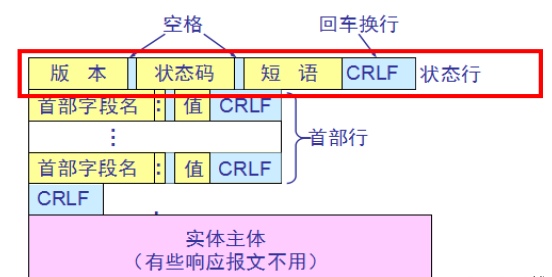
                Connection      是否开启持久连接（keepalive）

                Host            服务器域名

        主体部分就是报文的具体数据。

**3.2** **HTTP响应报文**

        响应报文的起始由状态行构成，用来说明服务器做了什么，由<Version>、<Status-Code>、<Phrase>三个字段组成，同样的每个字段之间留有空格；



        首部由多个响应头(也叫首部行)组成， 首部字段名如下，不全：

                Server    服务器软件名，Apache/Nginx

                Date      服务器发出响应报文的时间

                Last-Modified   请求资源的最后的修改时间

        主体部分是响应报文的具体数据。

**四、HTTP协议版本更替**

HTTP/0.9

        HTTP协议的最初版本，功能简陋，仅支持请求方式GET，并且仅能请求访问HTML格式的资源。

HTTP/1.0

        在0.9版本上做了进步，增加了请求方式POST和HEAD；不再局限于0.9版本的HTML格式，根据Content-Type可以支持多种数据格式，即MIME多用途互联网邮件扩展，例如text/html、image/jpeg等；同时也开始支持cache，就是当客户端在规定时间内访问统一网站，直接访问cache即可。

        但是1.0版本的工作方式是每次TCP连接只能发送一个请求，当服务器响应后就会关闭这次连接，下一个请求需要再次建立TCP连接，就是不支持keepalive。

HTTP/1.1

        解决了1.0版本的keepalive问题，1.1版本加入了持久连接，一个TCP连接可以允许多个HTTP请求； 加入了管道机制，一个TCP连接同时允许多个请求同时发送，增加了并发性；新增了请求方式PUT、PATCH、DELETE等。

        但是还存在一些问题，服务端是按队列顺序处理请求的，假如一个请求处理时间很长，则会导致后边的请求无法处理，这样就造成了队头阻塞的问题；同时HTTP是无状态的连接，因此每次请求都需要添加重复的字段，降低了带宽的利用率。

HTTP/2.0

        为了解决1.1版本利用率不高的问题，提出了HTTP/2.0版本。增加双工模式，即不仅客户端能够同时发送多个请求，服务端也能同时处理多个请求，解决了队头堵塞的问题；HTTP请求和响应中，状态行和请求/响应头都是些信息字段，并没有真正的数据，因此在2.0版本中将所有的信息字段建立一张表，为表中的每个字段建立索引，客户端和服务端共同使用这个表，他们之间就以索引号来表示信息字段，这样就避免了1.0旧版本的重复繁琐的字段，并以压缩的方式传输，提高利用率。

        另外也增加服务器推送的功能，即不经请求服务端主动向客户端发送数据。

当前主流的协议版本还是HTTP/1.1版本

**五、网站访问量**

        IP IP访问量

相同的公网IP计算一次，就是同一个局域网内的所有用户访问一个网站，但是他们都是借助一个公网IP去访问那个网站的（NAT），因此这也只能算作一个IP访问量。换一次公网IP则会加1。

        PV 网页访问量

用户访问的页面数就是PV访问量，同一个局域网的不同用户，而且就算是同一个用户，只要刷新一次网站页面，PV访问量就加1，三个访问量的值往往数PV的值最大。

        UV 访客访问量

这里的访客不是用户，而是电脑，一台电脑算一个访客，即使是同一台电脑的不同用户，访问同一个网站UV也只能加1，只有更换电脑才会使UV加1，因为服务端会记录客户端电脑的信息。