网络编程懒人入门(二)：快速理解网络通信协议（下篇）-网络编程/专项技术区 - 即时通讯开发者社区!

原作者：阮一峰(ruanyifeng.com），本文由即时通讯网重新整理发布，感谢原作者的无私分享。

1、前言

本文上篇《网络编程懒人入门(一)：快速理解网络通信协议（上篇）》分析了互联网的总体构思，从下至上，每一层协议的设计思想。基于知识连贯性的考虑，建议您先看完上篇后再来阅读本文。

本文从设计者的角度看问题，今天我想切换到用户的角度，看看用户是如何从上至下，与这些协议互动的。

2、系列文章

本文是系列文章中的第1篇，本系列文章的大纲如下：

- 《网络编程懒人入门(一)：快速理解网络通信协议（上篇）》
- 《网络编程懒人入门(二)：快速理解网络通信协议（下篇）》（本文）
- 《网络编程懒人入门(三)：快速理解TCP协议一篇就够》
- 《网络编程懒人入门(四)：快速理解TCP和UDP的差异》
- 《网络编程懒人入门(五)：快速理解为什么说UDP有时比TCP更有优势》
- 《网络编程懒人入门(六)：史上最通俗的集线器、交换机、路由器功能原理入门》
- 《网络编程懒人入门(七)：深入浅出，全面理解HTTP协议》
- 《网络编程懒人入门(八)：手把手教你写基于TCP的Socket长连接》
- 《网络编程懒人入门(九)：通俗讲解，有了IP地址，为何还要用MAC地址？》

本站的《脑残式网络编程入门》也适合入门学习，本系列大纲如下：

- 《脑残式网络编程入门(一)：跟着动画来学TCP三次握手和四次挥手》

- 《脑残式网络编程入门(二)：我们在读写Socket时，究竟在读写什么？》
- 《脑残式网络编程入门(三)：HTTP协议必知必会的一些知识》
- 《脑残式网络编程入门(四)：快速理解HTTP/2的服务器推送(Server Push)》

如果您觉得本系列文章过于基础，您可直接阅读《不为人知的网络编程》系列文章，该系列目录如下：

- 《不为人知的网络编程(一)：浅析TCP协议中的疑难杂症(上篇)》
- 《不为人知的网络编程(二)：浅析TCP协议中的疑难杂症(下篇)》
- 《不为人知的网络编程(三)：关闭TCP连接时为什么会TIME_WAIT、CLOSE_WAIT》
- 《不为人知的网络编程(四)：深入研究分析TCP的异常关闭》
- 《不为人知的网络编程(五)：UDP的连接性和负载均衡》
- 《不为人知的网络编程(六)：深入地理解UDP协议并用好它》
- 《不为人知的网络编程(七)：如何让不可靠的UDP变的可靠？》
- 《不为人知的网络编程(八)：从数据传输层深度解密HTTP》
- 《不为人知的网络编程(九)：理论联系实际，全方位深入理解DNS》

关于移动端网络特性及优化手段的总结性文章请见：

- 《现代移动端网络短连接的优化手段总结：请求速度、弱网适应、安全保障》
- 《移动端IM开发者必读(一)：通俗易懂，理解移动网络的“弱”和“慢”》
- 《移动端IM开发者必读(二)：史上最全移动弱网络优化方法总结》

3、参考资料

《TCP/IP详解 - 第11章·UDP：用户数据报协议》
《TCP/IP详解 - 第17章·TCP：传输控制协议》
《TCP/IP详解 - 第18章·TCP连接的建立与终止》
《TCP/IP详解 - 第21章·TCP的超时与重传》
《通俗易懂-深入理解TCP协议（上）：理论基础》
《通俗易懂-深入理解TCP协议（下）：RTT、滑动窗口、拥塞处理》
《理论经典：TCP协议的3次握手与4次挥手过程详解》
《理论联系实际：Wireshark抓包分析TCP 3次握手、4次挥手过程》
《计算机网络通讯协议关系图（中文珍藏版）》
《高性能网络编程(一)：单台服务器并发TCP连接数到底可以有多少》
《高性能网络编程(二)：上一个10年，著名的C10K并发连

接问题》

《高性能网络编程(三)：下一个10年，是时候考虑C10M并发问题了》

《高性能网络编程(四)：从C10K到C10M高性能网络应用的理论探索》

《简述传输层协议TCP和UDP的区别》

《为什么QQ用的是UDP协议而不是TCP协议？》

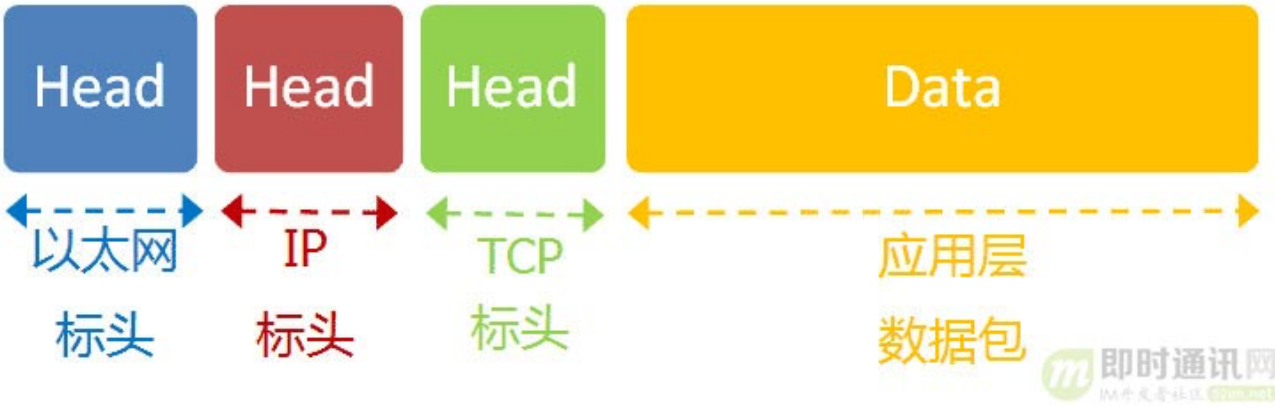
《移动端即时通讯协议选择：UDP还是TCP？》

4、一个小结

先对前面的内容，做一个小结（详见本文上篇《网络编程懒人入门(一)：快速理解网络通信协议（上篇）》）。

我们已经知道，网络通信就是交换数据包。电脑A向电脑B发送一个数据包，后者收到了，回复一个数据包，从而实现两台电脑之间的通信。

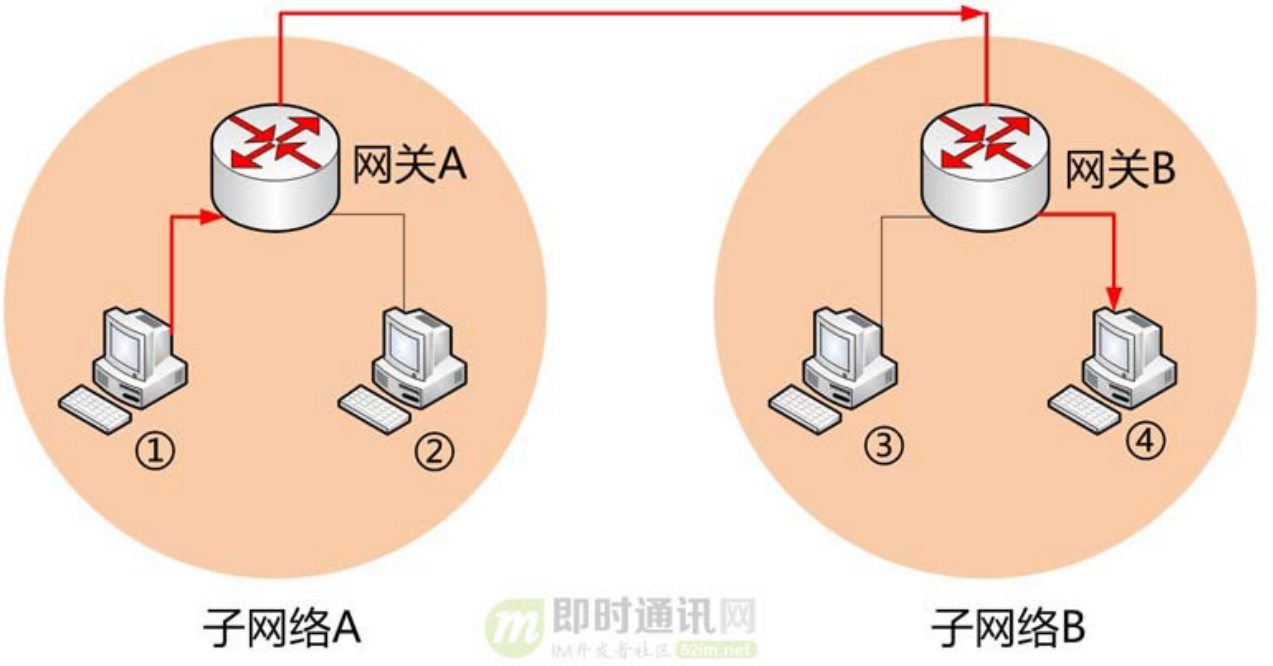
数据包的结构，基本上是下面这样：



发送这个包，需要知道两个地址：

- * 对方的MAC地址；
- * 对方的IP地址。

有了这两个地址，数据包才能准确送到接收者手中。但是，前面说过，MAC地址有局限性，如果两台电脑不在同一个子网络，就无法知道对方的MAC地址，必须通过网关（gateway）转发。



上图中，1号电脑要向4号电脑发送一个数据包。它先判断4号电脑是否在同一个子网络，结果发现不是（后文介绍判断方法），于是就把这个数据包发到网关A。网关A通过路由协议，发现4号电脑位于子网络B，又把数据包发给网关B，网关B再转发到4号电脑。

1号电脑把数据包发到网关A，必须知道网关A的MAC地址。所以，数据包的目标地址，实际上分成两种情况：

场景	数据包地址
同一个子网络	对方的MAC地址，对方的IP地址
非同一个子网络	网关的MAC地址，对方的IP地址

发送数据包之前，电脑必须判断对方是否在同一个子网络，然后选择相应的MAC地址。接下来，我们就来看，实际使用中，这个过程是怎么完成的。

5、用户的上网设置

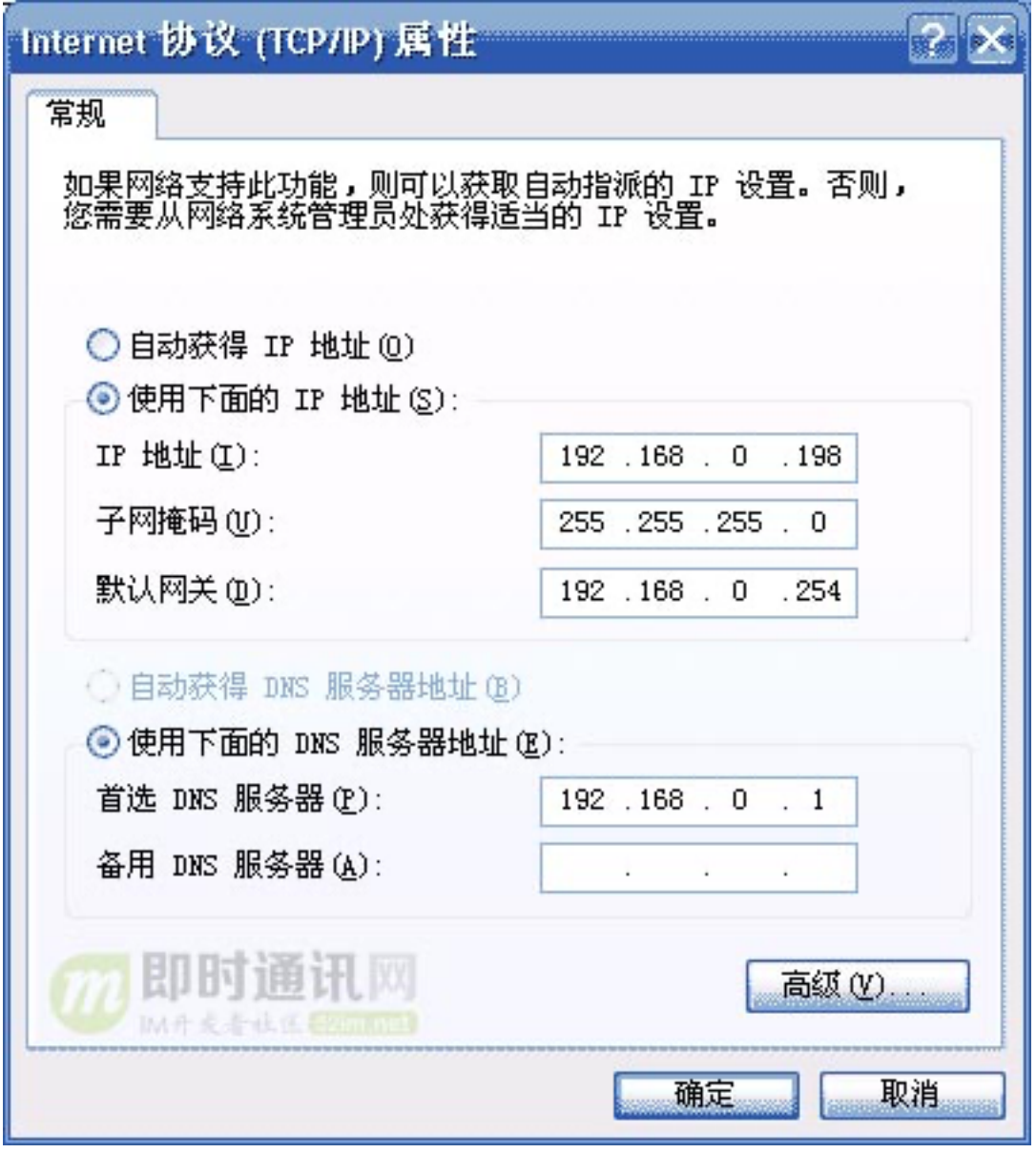
5.1静态IP地址

你买了一台新电脑，插上网线，开机，这时电脑能够上网吗？

通常你必须做一些设置。有时，管理员（或者ISP）会告诉你下面四个参数，你把它们填入操作系统，计算机就能连上网了：

- * 本机的IP地址；
- * 子网掩码；
- * 网关的IP地址；
- * DNS的IP地址。

下图是Windows系统的设置窗口：



这四个参数缺一不可，后文会解释为什么需要知道它们才能上网。由于它们是给定的，计算机每次开机，都会分到同样的IP地址，所以这种情况被称作"静态IP地址上网"。但是，这样的设置很专业，普通用户望而生畏，而且如果一台电脑的IP地址保持不变，其他电脑就不能使用这个地址，不够灵活。出于这两个原因，大多数用户使用"动态IP地址上网"。

5.2动态IP地址

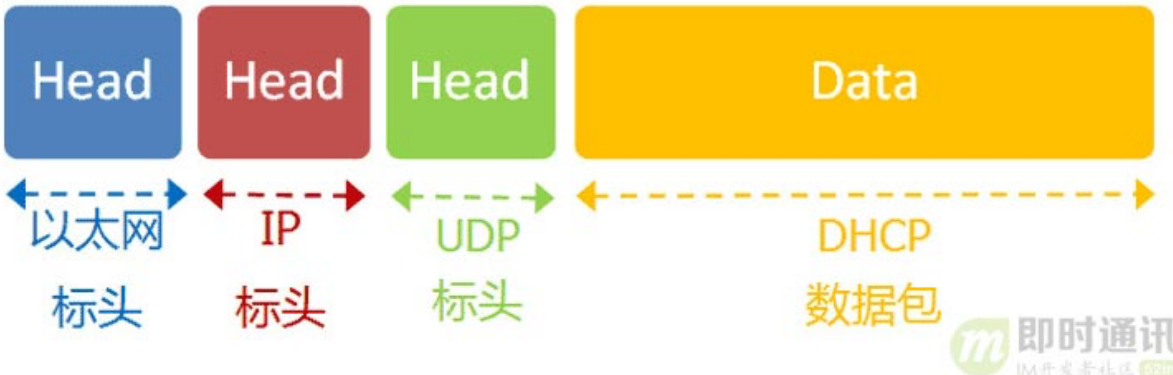
所谓"动态IP地址"，指计算机开机后，会自动分配到一个IP地址，不用人为设定。它使用的协议叫做DHCP协议。

这个协议规定，每一个子网络中，有一台计算机负责管理本网络的所有IP地址，它叫做"DHCP服务器"。新的计算机加入网络，必须向"DHCP服务器"发送一个"DHCP请求"数据包，申请IP地址和相关的网络参数。

前面说过，如果两台计算机在同一个子网络，必须知道对方的MAC地址和IP地址，才能发送数据包。但是，新加入的计算机不知道这两个地址，怎么发送数据包呢？DHCP协议做了一些巧妙的规定。

5.3DHCP协议

首先，它是一种应用层协议，建立在UDP协议之上，所以整个数据包是这样的：



- **1）最前面的"以太网标头"：**设置发出方（本机）的MAC地址和接收方（DHCP服务器）的MAC地址。前者就是本机网卡的MAC地址，后者这时不知道，就填入一个广播地址：FF-FF-FF-FF-FF-FF。
- **2）后面的"IP标头"：**设置发出方的IP地址和接收方的IP地址。这时，对于这两者，本机都不知道。于是，发出方的IP地址就设为0.0.0.0，接收方的IP地址设为255.255.255.255。
- **3）最后的"UDP标头"：**设置发出方的端口和接收方的端口。这一部分是DHCP协议规定好的，发出方是68端口，接收方是67端口。

这个数据包构造完成后，就可以发出了。以太网是广播发送，同一个子网络的每台计算机都收到了这个包。因为接收方的MAC地址是FF-FF-FF-FF-FF-FF，看不出是发给谁的，所以每台收到这个包的计算机，还必须分析这个包的IP地址，才能确定是不是发给自己的。当看到发出方IP地址是0.0.0.0，接收方是255.255.255.255，于是DHCP服务器知道"这个包是发给我的"，而其他计算机就可以丢弃这个包。

接下来，DHCP服务器读出这个包的数据内容，分配好IP地址，发送回去一个"DHCP响应"数据包。这个响应包的结构也是类似的，以太网标头的MAC地址是双方的网卡地址，IP标头的IP地址是DHCP服务器的IP地址（发出方）和255.255.255.255（接收方），UDP标头的端口是67（发出方）和68（接收方），分配给请求端的IP地址和本网络的具体参数则包含在Data部分。

新加入的计算机收到这个响应包，于是就知道了自己的IP地址、子网掩码、网关地址、DNS服务器等等参数。

5.4上网设置：小结

这个部分，需要记住的就是一点：不管是"静态IP地址"还是"动态IP地址"，电脑上网的首要步骤，是确定四个参数。

这四个值很重要，值得重复一遍：

- * 本机的IP地址；
- * 子网掩码；
- * 网关的IP地址；
- * DNS的IP地址。

有了这几个数值，电脑就可以上网"冲浪"了。接下来，我们来看一个实例，当用户访问网页的时候，互联网协议是怎么运作的。

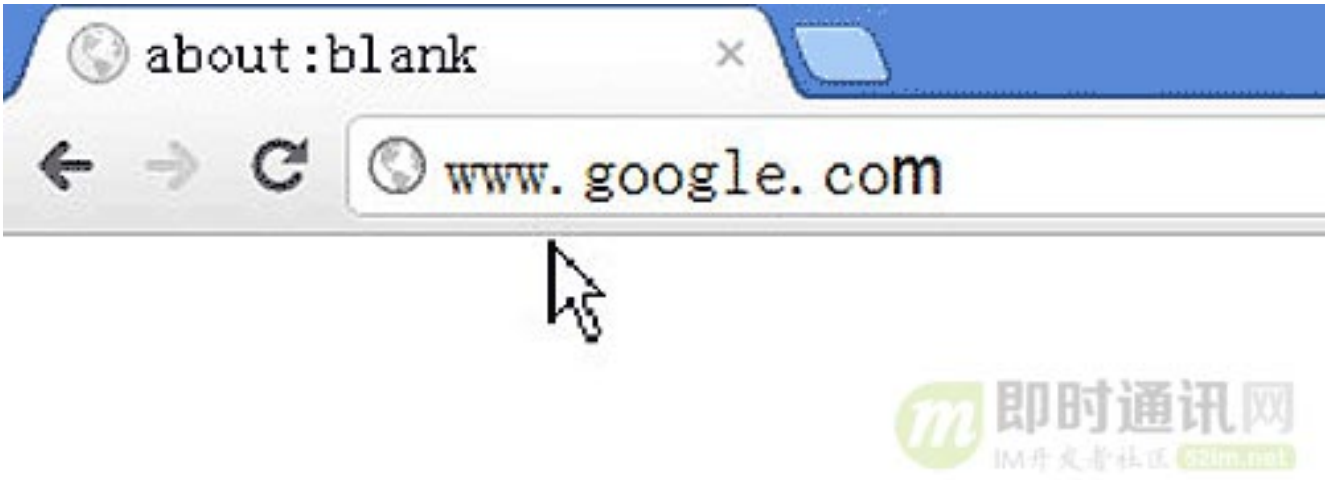
6、一个实例：访问网页

6.1本机参数

我们假定，经过上一节的步骤，用户设置好了自己的网络参数：

- * 本机的IP地址：192.168.1.100；
- * 子网掩码：255.255.255.0；
- * 网关的IP地址：192.168.1.1；
- * DNS的IP地址：8.8.8.8。

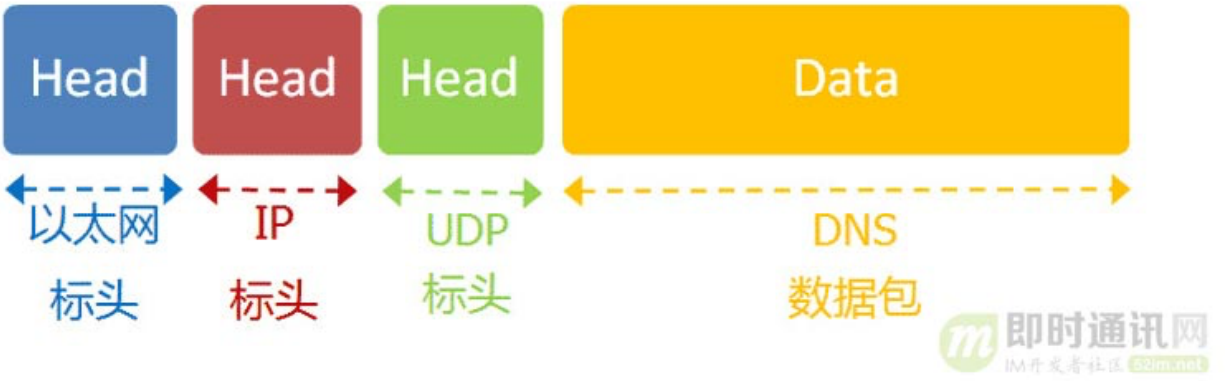
然后他打开浏览器，想要访问Google，在地址栏输入了网址：www.google.com。



这意味着，浏览器要向Google发送一个网页请求的数据包。

6.2DNS协议

我们知道，发送数据包，必须要知道对方的IP地址。但是，现在，我们只知道网址www.google.com，不知道它的IP地址。DNS协议可以帮助我们，将这个网址转换成IP地址。已知DNS服务器为8.8.8.8，于是我们向这个地址发送一个DNS数据包（53端口）。



然后，DNS服务器做出响应，告诉我们Google的IP地址是172.194.72.105。于是，我们知道了对方的IP地址。

6.3子网掩码

接下来，我们要判断，这个IP地址是不是在同一个子网络，这就要用到子网掩码。

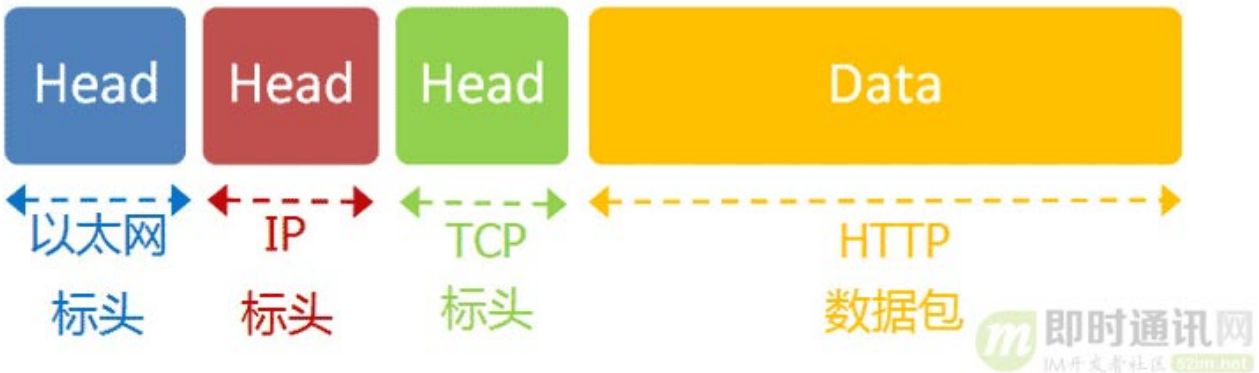
已知子网掩码是255.255.255.0，本机用它对自己的IP地址192.168.1.100，做一个二进制的AND运算（两个数位都为1，结果为1，否则为0），计算结果为192.168.1.0；然后对Google的IP地址172.194.72.105也做一个AND运算，计

算结果为172.194.72.0。这两个结果不相等，所以结论是，Google与本机不在同一个子网络。

因此，我们要向Google发送数据包，必须通过网关192.168.1.1转发，也就是说，接收方的MAC地址将是网关的MAC地址。

6.4应用层协议

浏览网页用的是HTTP协议，它的整个数据包构造是这样的：



HTTP部分的内容，类似于下面这样：

```
GET / HTTP/1.1
Host: [url=http://www.google.com]www.google.com[/url]
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 6.1) ......
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,
```

```
Accept-Encoding: gzip,deflate,sdch

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

Accept-Charset: GBK,utf-8;q=0.7,*;q=0.3

Cookie: ... ...

```

我们假定这个部分的长度为4960字节，它会被嵌在TCP数据包之中。

6.5TCP协议

TCP数据包需要设置端口，接收方（Google）的HTTP端口默认是80，发送方（本机）的端口是一个随机生成的1024-65535之间的整数，假定为51775。TCP数据包的标头长度为20字节，加上嵌入HTTP的数据包，总长度变为4980字节。

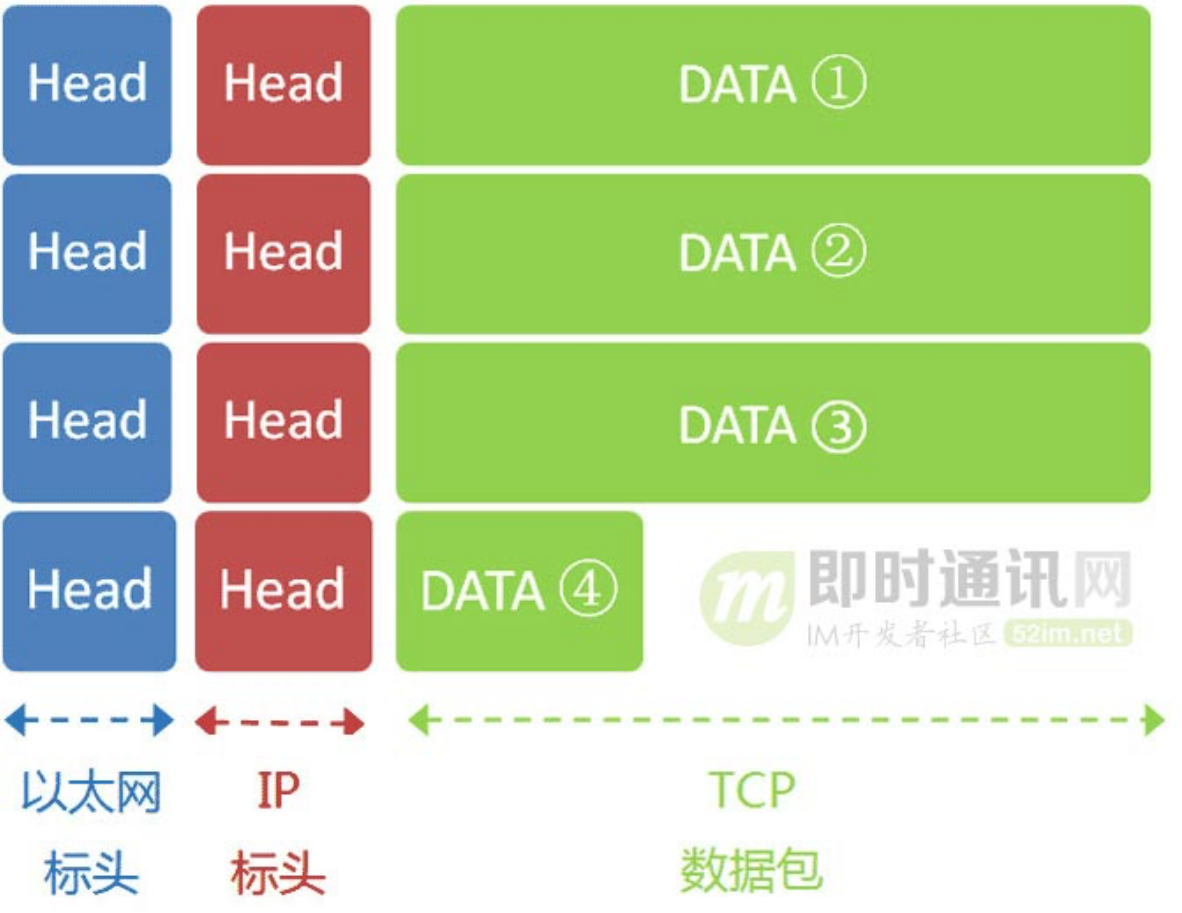
6.6IP协议

然后，TCP数据包再嵌入IP数据包。IP数据包需要设置双方的IP地址，这是已知的，发送方是192.168.1.100（本机），接收方是172.194.72.105（Google）。IP数据包的标头长度为20字节，加上嵌入的TCP数据包，总长度变为5000字节。

6.7以太网协议

最后，IP数据包嵌入以太网数据包。以太网数据包需要设置双方的MAC地址，发送方为本机的网卡MAC地址，接收方为网关192.168.1.1的MAC地址（通过ARP协议得到）。

以太网数据包的数据部分，最大长度为1500字节，而现在的IP数据包长度为5000字节。因此，IP数据包必须分割成四个包。因为每个包都有自己的IP标头（20字节），所以四个包的IP数据包的长度分别为1500、1500、1500、560。



6.8服务器端响应

经过多个网关的转发，Google的服务器172.194.72.105，收到了这四个以太网数据包。根据IP标头的序号，Google将四个包拼起来，取出完整的TCP数据包，然后读出里面的"HTTP请求"，接着做出"HTTP响应"，再用TCP协议发回来。

本机收到HTTP响应以后，就可以将网页显示出来，完成一次网络通信。

这个例子就到此为止，虽然经过了简化，但它大致上反映了互联网协议的整个通信过程。

（— 全文完 —）（原文链接：点此进入）

附录：更多网络编程资料

《技术往事：改变世界的TCP/IP协议（珍贵多图、手机慎点）》
《UDP中一个包的大小最大能多大？》
《Java新一代网络编程模型AIO原理及Linux系统AIO介绍》
《NIO框架入门(一)：服务端基于Netty4的UDP双向通信Demo演示》
《NIO框架入门(二)：服务端基于MINA2的UDP双向通信Demo演示》
《NIO框架入门(三)：iOS与MINA2、Netty4的跨平台UDP双向通信实战》
《NIO框架入门(四)：Android与MINA2、Netty4的跨平台UDP双向通信实战》
《P2P技术详解(一)：NAT详解——详细原理、P2P简介》
《P2P技术详解(二)：P2P中的NAT穿越(打洞)方案详解》
《P2P技术详解(三)：P2P技术之STUN、TURN、ICE详解》
《通俗易懂：快速理解P2P技术中的NAT穿透原理》
>> 更多同类文章 ……