### 计算机网络之漫谈

**一、前言**

我想写一系列文章来总结计算机网络，也作为拿宝的学习资料。因此，拿宝你是直接读者，在看这些文字的时候，就和老杨在你跟前给你说是一模一样的，你也可以把它当作小说读，或者就当作老杨发给你微信上很长的话，老杨相信，这样你会很容易理解网络中的一些诸如为什么计算机需要IP？没有IP可以吗？什么是路由器？什么是子网掩码？什么是MAC地址？TCP为什么是可靠传输？HTTP协议需要三次握手吗？为什么它又是无状态的？等问题。理解了这些问题之后，你会对你目前学习的node.js，wamp等知识有更深的理解。当然，由于个人能力有限，我会参考书籍和别人的资料，如果用到，我会给出原文出处。拿宝在看的过程中肯定会有哈哈大笑的时候，当然也会有眉头紧蹙，这时候我希望你能拿出笔大胆写下自己不理解的东西或者自己想要问的问题，我们可以后续讨论。好了，我知道你已经迫不及待想往下看了，我们继续^\_^。

1. **什么是网络？**

首先，为什么拿宝的微信消息能发送给老杨？拿宝现在已经可以咧着大嘴告诉我：消息是先到腾讯的微信服务器的，然后腾讯再遍历拿宝的好友列表，找到老杨，把消息发送给我。回答正确！但是也很明显，这个过程中拿宝和老杨的手机都是和腾讯服务器相连的。所以什么是网络呢？就是用物理设备将各个“局域网”相连组成的更大的“局域网”。小的局域网可以是家里的WiFi，或者陕西省你也可以看作是一个局域网。物理设备就是网线、光缆等。拿宝问：那美国和中国怎么办？老杨答：海底有电缆你怕啥。拿宝又问：我家和你家怎么办？老杨答：你家是广电的网，我家是电信的，你家用广电的线接到广电的服务器上，我家同理接到电信的服务器上，它们两个ISP（Internet Service Provider 互联网服务提供商）肯定也是相连的。下图1是2015年全球互联网跨国通信光缆的连接情况。

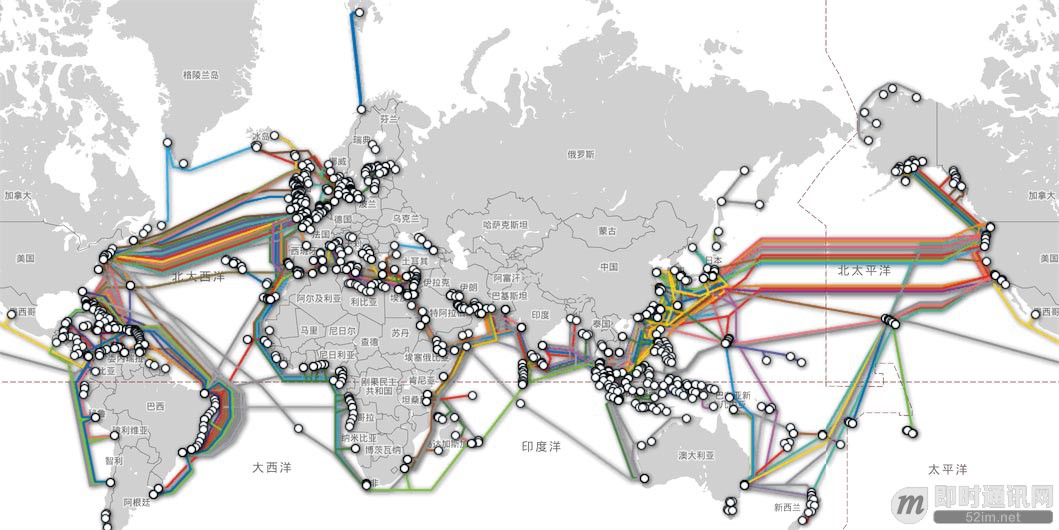


图1：2015年全球互联网跨国通信光缆的连接情况

其中密密麻麻的线就是各个国家之间的连接情况。所以网络没有什么神奇的，小到一台手机，大到一个国家如果想在目前我们所熟知的“互联网”上获取或者共享信息，那它就必须以某种物理方式接入到互联网。当然，只把这些通信设备连在一起肯定不行，必须得有一些约定或者说规矩来告诉不同型号的设备、不同规格的网络如何按照同样的方法处理相同的数据、网络上谁是谁？这就是网络协议了。早期的计算机网络，都是由各厂商自己规定一套协议，IBM，Apple，和MicroSoft都有自己的网络协议，比如IBM的两台电脑用网线连起来，互相说话能听懂。但是IBM和Apple的电脑连接起来说话就听不懂了，想想你和我微信聊天，我是IBM电脑，你是Apple电脑，你发送的消息到我这里显示不了或者解析成另一个意思，你说多坑。为了把全世界的所有不同类型的计算机都连接起来，就必须规定一套全球通用的协议，为了实现这个目标，互联网协议簇（Internet Protocol Suite）就成为了通用协议标准。Internet是由inter和net两个单词组合起来的，原意就是连接“网络”的网络，有了Internet，任何私有网络，只要支持这个协议，就可以联入互联网。互联网协议包含了上百种协议，但是最重要的两个协议是TCP和IP协议，而我们通常把基于TCP和IP协议的所有协议统称为”TCP/IP协议（蔟）”。因此，计算机网络说白了就是物理连接的“局域网”和工作于这个局域网上的“网络协议”。我们接下来会把重点放在讨论“网络协议”上面，你准备好了吗？

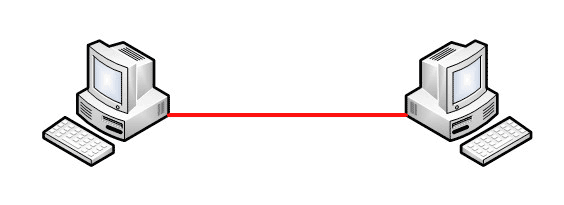
1. **OSI七层模型和TCP/IP四层模型**

在 **什么是网络？** 中拿宝已经知道计算机网络是物理连接的“局域网”和工作于这个局域网上的“网络协议”，并且我们的重心是网络协议。有关网络协议，按照目前的分层方式主要有两种，一种是OSI七层模型（忽略它，没什么卵用 :）），一种是TCP/IP四层模型。下面我们主要来看看它们的对应关系和工作在不同层的具体协议。

|  |  |  |
| --- | --- | --- |
| **OSI** | **TCP/IP** | **常见网络协议** |
| **应用层（Application）** | **应用层** | HTTP（超文本传输协议）  FTP（文件传输协议）  DNS（域名系统） |
| **表示层（Represent）** |
| **会话层（Conversation）** |
| **传输层（Transport）** | **传输层** | TCP（传输控制协议）  UDP（用户数据报协议） |
| **网络层（Internet）** | **网络层** | IP（网络协议）  ICMP（网络控制消息协议） |
| **数据链路层（Data Link）** | **数据链路层** | ARP（地址解析协议） |
| **物理层（Physical）** |

拿宝，再看一次上面的图片，确保你记住了每层对应的网络协议。现在你可能还有疑问：为什么要分层呢？答：说白了就是因为在网络传输中，所需要解决的问题不是同一个类型的或者层次的，比如要实现传递可靠数据（传输层），先要实现能传递数据（网络层），那最起码需要实现两台电脑之间能发数据包（数据链路层）。所以分层的解决方式无疑成为了首选，每一层都有自己需要解决的问题，下层协议为上层协议提供帮助。就像工厂的流水线一样，从最初的产品 -> 纸袋包装 ->礼品盒包装 ，而对面打开的次序刚好是相反的（数据包的拆包亦是如此）。下面我们要从协议最底部出发，沿着 数据链路层->网络层->传输层->应用层 的顺序依次分析这些协议。

1. **数据链路层**

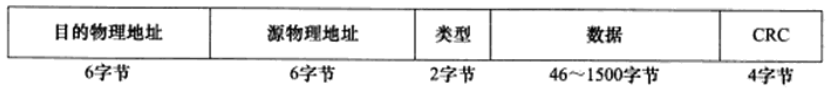
拿宝，我们从整个协议最底层开始，先来看看数据链路层主要都做了哪些事：****

首先上面两台电脑需要通信，先将它们物理连接起来，然后就可以传送高低电位了，高电位表示1，低电位表示0。但是单纯的0和1没有任何的意义，我们必须人为的规定解读方式，比如：多少个0和1作为一组信号？每个信号位有什么含义？......

拿宝，还记得不，我们前面说过互联网早期的时候好多公司都有自己的网络协议。但是最后逐渐被“以太网”占据了主导地位，以太网规定，一组电信号构成一个数据包，叫做“帧”。每一帧分为两个部分：标头（Head）和数据（Data）。

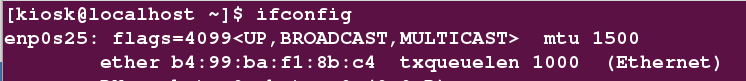


"标头"包含数据包的一些说明项，比如发送者（源物理地址）、接受者（目的物理地址）、数据类型、CRC校验码等等。"数据"则是数据包的具体内容。具体的帧结构如下图所示：



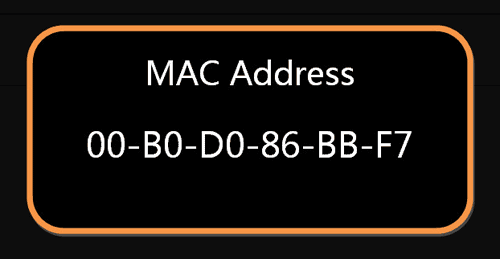
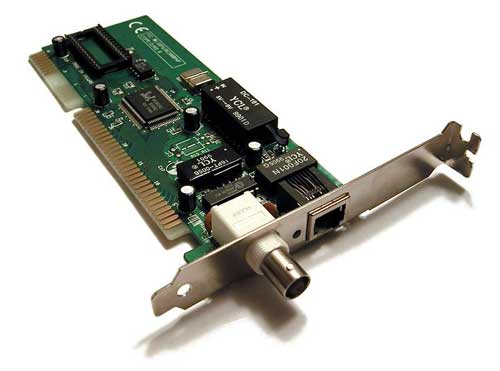
"标头"的长度，固定为18字节（6+6+2+4=18）。"数据"的长度，最短为46字节，最长为1500字节（MTU Maximum Transmission Unit 最大传输单元）。因此，整个"帧"最短为64字节，最长为1518字节。如果数据很长，就必须分割成多个帧进行发送。

拿宝可以使用ifconfig命令来查看自己网卡的MTU，下图是我的网卡情况。



从上面“帧”的结构中可以看到，“标头”包含了“源物理地址”和“目的物理地址”，其实这就是我们常说的MAC地址。以太网规定，连入网络的所有设备，都必须具有"网卡"接口。数据包必须是从一块网卡，传送到另一块网卡。网卡的地址，就是数据包的发送地址和接收地址，这叫做MAC地址。

每块网卡出厂的时候，都有一个全世界独一无二的MAC地址，长度是48个二进制位，通常用12个十六进制数表示。前6个十六进制数是厂商编号，后6个是该厂商的网卡流水号。有了MAC地址，就可以定位网卡和数据包的路径了。（同样，ifconfig命令就可以查看网卡的MAC地址，上张截图中有我的网卡MAC地址，你找找看）

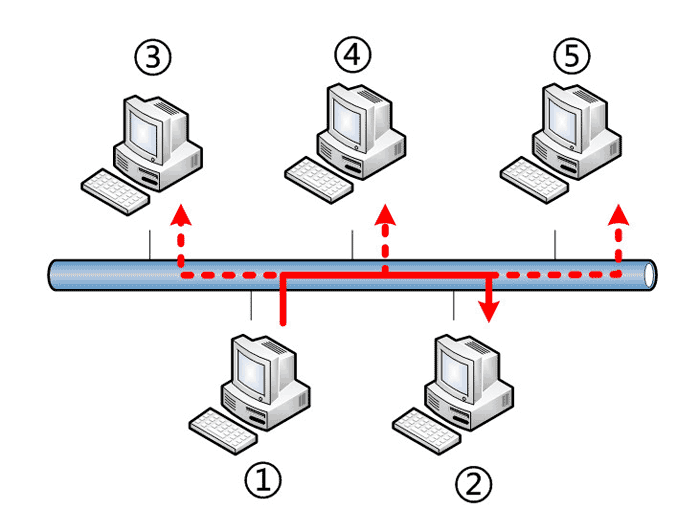


当然，现在有了地址也只是第一步，可是我的网卡怎么会知道拿宝网卡的MAC地址呢？

答：有一种ARP协议，可以解决这个问题。这个留到后面再说，这里只需要知道，以太网数据包必须知道接收方的MAC地址，然后才能发送。

其次，就算有了MAC地址，系统怎样才能把数据包准确送到接收方？

答：以太网采用了一种很"原始"的方式，它不是把数据包准确送到接收方，而是向本网络内**所有计算机**发送，让每台计算机自己判断，是否为接收方。



上图中，1号计算机向2号计算机发送一个数据包，同一个子网络的3号、4号、5号计算机都会收到这个包。它们读取这个包的"标头"，找到接收方的MAC地址，然后与自身的MAC地址相比较，如果两者相同，就接受这个包，做进一步处理，否则就丢弃这个包。这种发送方式就叫做"广播"（broadcasting）。

现在，我们假设全球的电脑都是用网线直接相连的，那有了数据包的定义、网卡的MAC地址、广播的发送方式，只要知道目的MAC地址我们就可以实现给世界各地任何电脑发送数据了。比如我可以直接广播拿宝网卡的MAC地址，发送“我爱你”，然后全球所有的网卡都会收到这条消息，但是只有你的网卡才会接受这条消息，也就是“我只爱你”。

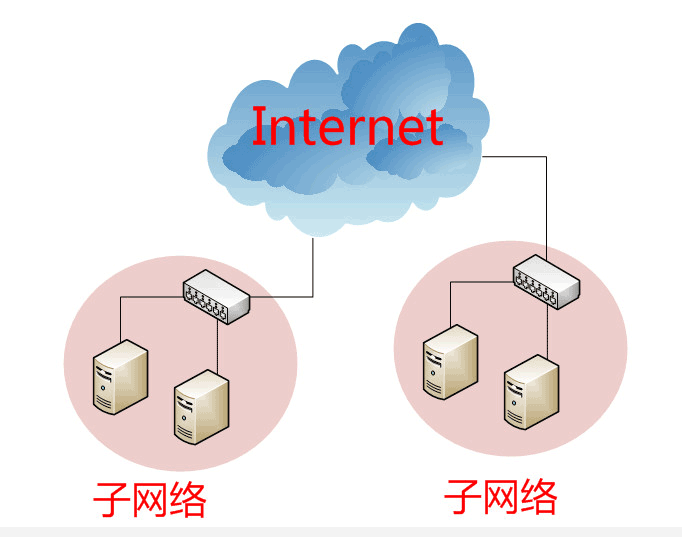
好，回过头来看看，我们现在已经从两台网线直接相连的电脑发展到全球电脑都能通信了，真棒，那请拿宝思考下广播的这种方式到底合适吗？请将自己的想法回复给我。

**五、网络层**

拿宝刚才责备老杨没有理你微信上回答的问题，还说打击了你的信心，我突然想起来今天还没有给你发文档呢。写了没？ 没有，那还睡觉？算了，还是要写完发给你才可以的。老杨现在在炕上办公，酷！废话少说，进入正题。

在 **数据链路层** 中，我们Get了数据包的定义（帧=标头+数据）、网卡的MAC地址、广播的发送方式三个重要概念。并且我们知道在技术上，依靠MAC地址和广播的发送的方式，我们其实可以实现全世界范围内的计算机互相通信（比如陇县和芝加哥）。留给拿宝的问题是：广播的方式到底合适吗？拿宝答：不能适用一般数据包的发送，会给网络造成很大的压力，是资源的浪费。回答正确！就是这样，世界上这么多计算机，两两通信如果所有人都会收到，网络上的数据量将是不可估计的，会造成灾难。

那实际上呢，目前的互联网在设计的时候，广播的方式是局限在发送者所在的子网络的。互联网是无数子网络共同组成的一个巨型网络，很难想象陇县和芝加哥的电脑会在同一个子网络，这几乎是不可能的。如下图是互联网的组成情况，拿宝，想想我们之前对网络的定义，是不是由多个局域网相连组成的，下面的子网络也就是局域网。



因此，必须找到一种方法，能够区分哪些MAC地址属于同一个子网络，哪些不是。如果是同一个子网络，就采用广播方式发送，否则就采用"路由"方式发送。（"路由"的意思，就是指如何向不同的子网络分发数据包，这是路由器干的活，我们先不涉及）遗憾的是，MAC地址本身无法做到这一点。它只与厂商有关，与所处网络无关。

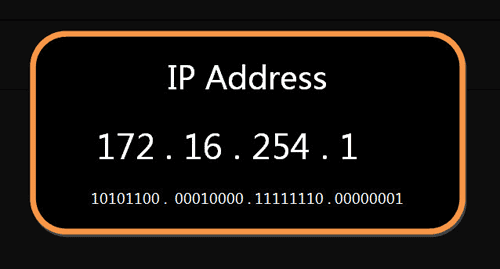
**这就导致了"网络层"的诞生。它的作用是引进一套新的地址，使得我们能够区分不同的计算机是否属于同一个子网络。这套地址就叫做"网络地址"，简称"网址"。**

于是，"网络层"出现以后，每台计算机有了两种地址，一种是MAC地址，另一种是网络地址。两种地址之间没有任何联系，MAC地址是绑定在网卡上的，网络地址则是管理员分配的，它们只是**随机组合**在一起（因此网址可以变化）。

网络地址帮助我们确定计算机所在的子网络，MAC地址则将数据包送到该子网络中的目标网卡。因此，从逻辑上可以推断，必定是先处理网络地址，然后再处理MAC地址。

规定网络地址的协议，叫做IP协议。它所定义的地址，就被称为**IP地址**。

目前，广泛采用的是IP协议第四版，简称IPv4。这个版本规定，网络地址由32个二进制位组成。

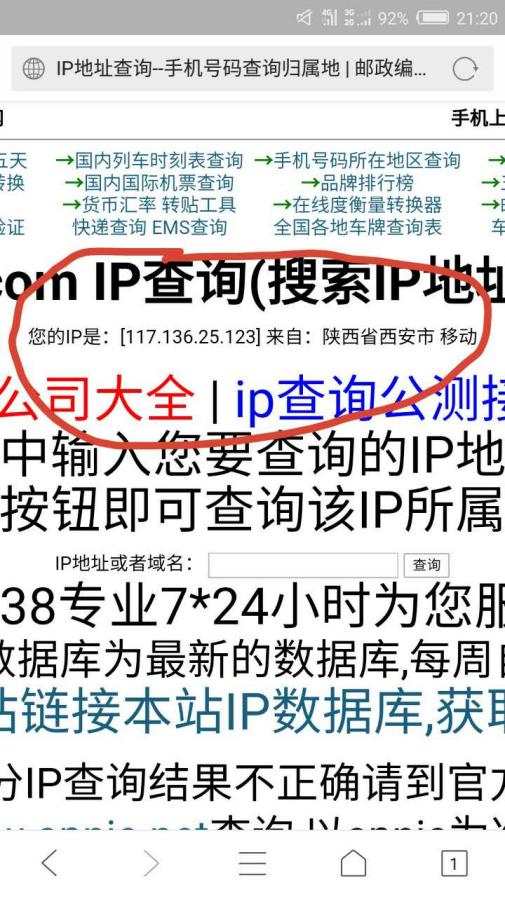
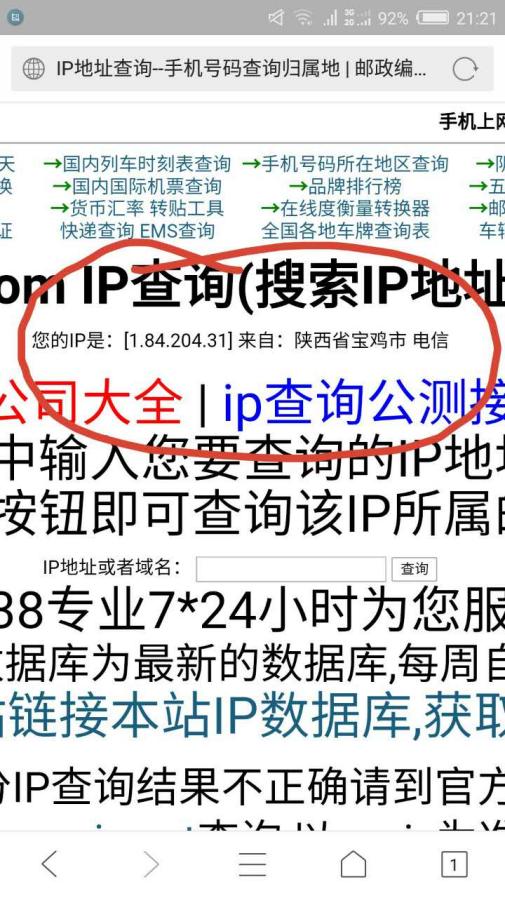


我们习惯用分成四段的十进制数表示IP地址，从0.0.0.0一直到255.255.255.255。

互联网上的每一台计算机，都会分配到一个IP地址。这个地址分成两个部分，前一部分代表网络，后一部分代表主机。比如，IP地址172.16.254.1，这是一个32位的地址，假定它的网络部分是前24位（172.16.254），那么主机部分就是后8位（最后的那个1）。处于同一个子网络的电脑，它们IP地址的网络部分必定是相同的，也就是说172.16.254.2应该与172.16.254.1处在同一个子网络。

**六、上网就需要IP吗？**

拿宝给老杨发送了消息，但是老杨去打球了，手机没有连接任何WiFi，但是老杨手机开的是移动4G，消息也收到了。仔细想想这个过程。拿宝连接的是家里的WiFi，消息经过路由器传到了腾讯的服务器，但是腾讯又怎么将消息转发到老杨手机的呢？是先发给移动公司然后移动公司再根据手机号转发给我？还是直接给我就行了？如果直接给的话它又是怎么确认我手机的地址的？（往下看之前好好想想这个问题......）好了，不卖关子了，其实这两个过程是一样的，都需要IP来唯一确定我手机地址，只是我从家里WiFi和移动4G拿到的IP不同罢了，腾讯只会根据IP找我的手机。下面两个图分别是我连接WiFi和移动4G时的目前IP（查询IP：<http://www.ip138.com）。>

移动4G WiFi

可以看到情况如下所示：

4G： 117.136.25.123 陕西省西安市 移动

WiFi：1.84.204.31 陕西省宝鸡市 电信

这下就更好理解了吧，无论我的手机在哪？连接WiFi还是4G，如果需要上网，那我必须获取一个IP，如果没有IP呢？比如在山里移动的信号塔没有修到的地方，那上网就难了，也就是拿不到IP。结合昨天说的，也就是这个IP确定了我在哪，然后腾讯将数据包路由给我，下来我们仔细研究研究IP这个东西。

1. **IP与子网掩码**

通过年前的文档[今天都初八了，捂脸]，拿宝现在已有的概念是每一台计算机如果需要接入互联网，都会分配到一个IP地址。这个地址分成两个部分，前一部分代表网络，后一部分代表主机。比如，IP地址172.16.254.1，这是一个32位的地址，假定它的网络部分是前24位（172.16.254），那么主机部分就是后8位（最后的那个1）。因此，处于同一个子网络的电脑，它们IP地址的网络部分必定是相同的，也就是说172.16.254.2应该与172.16.254.1处在同一个子网络（它们的网络号相同：172.16.254）。

但是，问题在于单单从IP地址，我们无法判断网络部分。还是以172.16.254.1为例，它的网络部分，到底是前24位，还是前16位，甚至前28位，从IP地址上是看不出来的。

那么，怎样才能从IP地址，判断两台计算机是否属于同一个子网络呢？这就要用到另一个参数"**子网掩码**"（subnet mask）。

所谓"子网掩码"，就是表示子网络特征的一个参数。它在形式上等同于IP地址，也是一个32位二进制数字，它的网络部分全部为1，主机部分全部为0。比如，IP地址172.16.254.1，如果已知网络部分是前24位，主机部分是后8位，那么子网络掩码就是11111111.11111111.11111111.00000000，写成十进制就是255.255.255.0。

知道"子网掩码"，我们就能判断，任意两个IP地址是否处在同一个子网络。方法是将两个IP地址与子网掩码分别进行AND运算（两个数位都为1，运算结果为1，否则为0），然后比较结果是否相同，如果是的话，就表明它们在同一个子网络中，否则就不是。

比如，已知IP地址172.16.254.1和172.16.254.233的子网掩码都是255.255.255.0，请问它们是否在同一个子网络？答：两者与子网掩码分别进行AND运算，结果都是172.16.254.0，因此它们在同一个子网络。

因此，我们总结一下IP协议的主要作用：一个是为每一台计算机分配IP地址，另一个是确定哪些地址在同一个子网络。

1. **IP数据包**

网络层从**网络层、上网就需要IP吗？**以及**IP与子网掩码**前前后后我们也说了三次了，IP这个东西絮絮叨叨的也一直在提。今天我们来解开IP协议的面纱，拿宝还记得我们之前在数据链路层说的物理帧的结构吗？就是这样：



其中Head叫标头，包含源和目的MAC地址，那时我们还没有IP的概念，但是我们用广播的方式也已经能实现全世界范围的通信，但之后我们否定了这种方式，因为会造成网络拥堵，浪费资源。因此就有了子网络，自此，MAC地址和IP地址就随机的结合在了一起，网络通信发送的数据包中也就有了IP地址。但是上面说的帧头部（Head）中只包含MAC地址，并没有IP地址的栏位，那么是否需要修改数据定义，再添加一个栏位呢？

回答是不需要，我们可以把IP数据包直接放进以太网数据包的"数据"部分，因此完全不用修改数据链路层帧的结构。这就是互联网分层结构的好处：**上层的变动完全不涉及下层的结构**。

我们先来说IP数据包，具体来说，IP数据包也分为"标头"和"数据"两个部分。



"标头"部分主要包括**版本、长度、IP地址**等信息，"数据"部分则是IP数据包的**具体内容**。IP数据包的"标头"部分的长度为20到60字节（下次我们仔细研究），整个数据包的总长度最大为65,535字节。因此，理论上，一个IP数据包的"数据"部分，最长为65,515（65,535-20）字节。前面说过，以太网数据包的"数据"部分，最长只有1500（MTU）字节。因此，如果IP数据包超过了1500字节，它就需要分割成几个以太网数据包，分开发送了。我们将它放进以太网数据包后，以太网数据包就变成了下面这样。

****

**上**面红色的是**网络层的IP数据包**，它又作为了**数据链路层“帧”**的数据部分。是不是就能感受到这种从上向下逐层包装的意思，上层的数据包作为下层的数据部分，下层加入自己的包头组成自己本层完整的数据包。再往下层传递又作为下层的数据包，直到数据链路层的帧，最后被转换成高低电位发送。下次我们研究下IP数据包的包头。

1. **ARP协议（补）**

拿宝，继续前行；

上次我们提到了，因为IP数据包是放在以太网数据包里发送的，所以我们必须同时知道两个地址，一个是对方的MAC地址，另一个是对方的IP地址。通常情况下，对方的IP地址是已知的（就像你要给我打电话，肯定知道我的号码），但是我们不知道它的MAC地址。所以，我们需要一种机制，能够从IP地址得到MAC地址。

这里又可以分成两种情况。第一种情况，如果两台主机不在同一个子网络，那么事实上没有办法得到对方的MAC地址，只能把数据包传送到两个子网络连接处的"网关"（gateway），让网关去处理。

第二种情况，如果两台主机在同一个子网络，那么我们可以用ARP协议，得到对方的MAC地址。ARP协议也是发出一个数据包（包含在以太网数据包中），其中包含它所要查询主机的IP地址，在对方的MAC地址这一栏，填的是FF:FF:FF:FF:FF:FF，表示这是一个"广播"地址。它所在子网络的每一台主机，都会收到这个数据包，从中取出IP地址，与自身的IP地址进行比较。如果两者相同，都做出回复，向对方报告自己的MAC地址，否则就丢弃这个包。

因此，有了ARP协议之后，我们就可以得到同一个子网络内的主机MAC地址，可以把数据包发送到任意一台主机之上了。

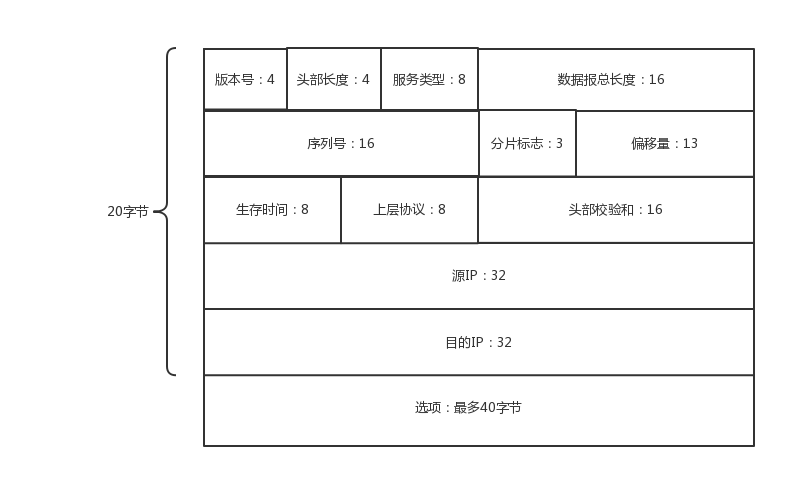
1. **IP数据包包头详解**

**拿宝，由于我没有得到你的回应，也不知道难度是否适合？但是我觉得还是一步一步从简单开始的，如果有看不懂需要调整或者停下来的需求，及时通知我！如果跟不上目前的进度，不要着急，先尝试看看前面的，找到自己的疑问，解决了就好了，前面几节我觉得你理解的都挺好的，请务必保持清醒头脑我们每节都在说什么。**

今天我们要研究IP数据包的包头，研究它的意义在于我们能了解到IP协议详细的细节。我们知道协议其实就是将数据按照一定格式组织起来，我们今天看的就是IP协议到底是怎么组织的。而包头才是重点，IP数据包的包头位置如下图红色 Head 所示：

****

如果忘记这个图的来历 :) ，请前往 **八、IP数据包** 查看。首先来一张IP数据包包头的字段图如下所示：



下面我们来看看每个字段的具体含义：

(1)**版本号**　占4位，指IP协议的版本。IPv4的值为 4

(2)**头部长度**　占4位，标识有多少个32bit字（4字节），4位最大为1111，能表示15，因此IP头部最长为15\*4=60字节。

(3)**服务类型**　占8位，可以设置不同的服务功能，例如小延迟和大吞吐量。

(4)**数据报总长度**　总长度指首部和数据之和的长度，单位为字节。总长度字段为16位，因此数据报的最大长度为2^16-1=65535字节。

　　(5)**序列号**　占16位。能唯一标识主机发送的每一个数据报。

(6)**分片标志**　三个bit位。第一位保留，未使用。第二位是DF（Don’t Fragment），如果为1，表示未发生分片。第三位是MF（More Fragment），如果为1，表示发生了分片，并且除了分片出的最后一个报文中此标志为0，其余报文中此标志均为1。

(7)**偏移量**　占13位。分片相对于原始IP数据报开始处的偏移。

　　(8)**生存时间**　占8位，生存时间字段常用的的英文缩写是TTL(Time To Live)，表明是数据报在网络中的寿命。每经过一个路由器时，就把 TTL值减1。当TTL值为0时，就丢弃这个数据报。

(9)**上层协议**　占8位，协议字段指出此数据报携带的数据是使用何种上层协议，以便使目的主机的IP层知道应将数据部分上交给上层哪个处理过程。

(10)**首部检验和**　占16位。由发送端填充，接收端对其CRC检验以确定头部是否损坏。

(11)**源IP地址**　 占32位。标识源IP。

(12)**目的IP地址**　占32位。标识目的IP。

1. **传输层**

拿宝，之前我们通过学习网络层和数据链路层知道了一个计算机有两个地址，MAC地址和IP地址，有了**MAC地址和IP地址**，我们已经可以在互联网上任意两台主机上建立通信。

现在面临的问题是，同一台主机上有许多程序都需要用到网络，比如，你一边浏览网页，一边与我QQ聊天。当一个数据包从互联网上发到你电脑的时候，它怎么知道，这是表示网页的内容，还是表示聊天的内容？

也就是说，我们还需要一个参数，表示这个数据包到底供哪个程序（进程）使用。是浏览器呢还是QQ？这个参数就叫做"端口"（port），它其实是每一个使用网卡的程序的编号。每个数据包都发到主机的特定端口（比如浏览器就监听80端口），所以不同的程序就能取到自己所需要的数据。

"端口"是0到65535之间的一个整数，正好16个二进制位。不管是浏览网页还是QQ聊天，应用程序会**随机选用**一个本地端口，然后与服务器的相应端口联系。

"传输层"的功能，就是建立"端口到端口"的通信（PORT）。相比之下，"网络层"的功能是建立"主机到主机"的通信（IP）。只要确定主机和端口，我们就能实现程序之间的交流。也就是我们讨论Node.js时说的，你启动Node.js服务器实际上干了什么？就是让你本机（127.0.0.1）在2333端口监听数据。127.0.0.1确定你本机，2333确定这个程序是Node.js。对，就是这么简单。

**十二、最简单的传输层协议——UDP协议**

拿宝，看到没？**最简单**的传输层协议，懂我意思了没？

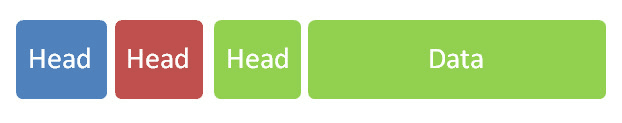
昨天咱们讨论了如果需要确定一个计算机上的不同网络程序（比如QQ和浏览器），需要**端口**的标识，但是IP头部和帧的头部都没有端口的标识字段，需要新的协议。和前面IP协议的实现套路一样，我们需要一个空间来存放端口号，因此就有了传输层的协议TCP和UDP。最简单的实现就是UDP协议，它的格式几乎就是在数据前面，加上端口号。UDP数据包，也是由"标头"和"数据"两部分组成。



"标头"部分主要定义了源端口和接收端口，"数据"部分就是具体的内容。具体的Head结构如下所示：



然后，把整个UDP数据包放入IP数据包的"**数据**"部分，而前面说过，IP数据包又是放在以太网数据包之中的，所以整个以太网数据包现在变成了下面这样：

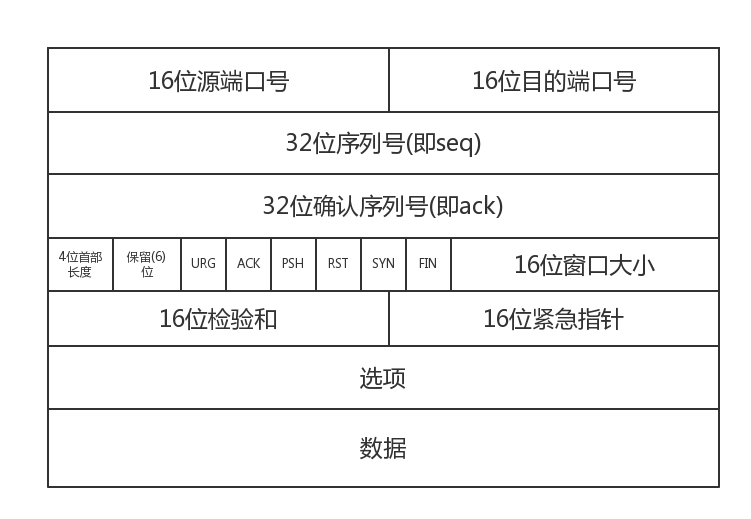


目前我们的数据包更加完善了，也更加能体现层层封装的意思。绿色头部是UDP头部，红色头部是IP头部，蓝色头部是帧的头部，上层**标头+数据**作为下层的**数据**。

1. **可靠的TCP协议**

拿宝，昨天我们讨论了UDP协议，UDP协议的优点是比较简单（它的数据包包头几乎只有端口和长度），容易实现，但是缺点是可靠性较差，一旦数据包发出，无法知道对方是否收到。为了解决这个问题，提高网络可靠性，TCP协议就诞生了。这个协议非常复杂，但可以近似认为，它就是有确认机制的UDP协议，每发出一个数据包都要求确认。如果有一个数据包遗失，就收不到确认，发出方就知道有必要重发这个数据包了。

因此，TCP协议能够确保数据不会遗失。它的缺点是过程复杂（TCP三次握手、四次挥手等）、实现困难（数据重传机制、流量控制机制等）、消耗较多的资源。TCP数据包和UDP数据包一样，都是内嵌在IP数据包的"数据"部分。TCP数据包也是由“标头”和“数据”组成的。当然，“标头”是重点。下面我们一起来看看TCP数据包的“标头”。



明显比UDP的数据包“标头”复杂多了，我们下面来依次解释每个字段的含义。我尽量以最简单的方式来解释每个字段，目前你大概知道是干什么的就行了。

**(1)源端口与目的端口**: 各占16位，分别写入源端口号和目的端口号。  
 **(2)32位序列号**: 占32位，能唯一标识一次通信中的数据包序号，可以看作是一个数据包的ID。

**(3)32位确认序列号**: 占32位，作为确认收到使用，比如如果想确定上面**32位序列号**，就把此值填为其值加1。

**(4)首部长度**: 占4位.表示整个“标头”的长度。  
 **(5)URG**: 占1位，紧急指针标志位,当URG=1时,表明紧急指针字段有效.它告诉系统中有紧急数据,应当尽快传送,这时不会按照原来的排队序列来传送.而会将紧急数据插入到本报文段数据的最前面。  
 **(6)ACK**: 占1位，当ACK=1时,确认序列号才有效,当ACK=0时,确认序号ack无效。

**(7)PSH**: 占1位，推送操作,很少用,没有了解。  
 **(8)RST**: 占1位，当RST=1时,表明TCP连接出现严重错误,此时必须释放连接。 **(9)SYN**: 占1位，请求连接的标志。  
 **(10)FIN**: 占1位，释放链接的标志.   
 **(11)16位窗口的大小**：占16位，表示期望对方发送的字节数。

**(12)16位检验和**: 占16位，用于检验包数据是否正确。  
 **(13)紧急指针**: 占16位，只有当URG=1时的时候,紧急指针才有效,它指出紧急数据的字节数。