

網絡七層協議的通俗理解

魯魯檳 PHP在線 今天

一、需求

1.1、物理層

作用是負責傳送0和1的電信號。

科學家要解決的第一個問題是，兩個硬件之間怎麼通信。具體就是一台發些比特流，然後另一台能收到。

於是，科學家發明了物理層：主要定義物理設備標準，如網線的接口類型、光纖的接口類型、各種傳輸介質的傳輸速率等。它的主要作用是傳輸比特流(就是由1、0轉化為電流強弱來進行傳輸，到達目的地後在轉化為1、0，也就是我們常說的數模轉換與模數轉換)。這一層的數據叫做比特。

調製解調器是一種計算機硬件，它能把計算機的數字信號翻譯成可沿普通電話線傳送的模擬信號，而這些模擬信號又可被線路另一端的另一個調製解調器接收，並譯成計算機可懂的語言。這一簡單過程完成了兩台計算機間的通信。

中繼器 (**RP repeater**) 工作於OSI的物理層，是局域網上所有節點的中心，它的作用是放大信號，補償信號衰減，支持遠距離的通信。中繼器。工作於物理層，只是起到擴展傳輸距離的作用，對高層協議是透明的。實際上，通過中繼器連接起來的網絡相當於同一條電線組成的更大的網絡。中繼器也能把不同傳輸介質 (**10Base 5**和**10Base 2**) 的網絡連在一起，多用在數據鏈路層以上相同的局域網的互連中。

1.2、數據鏈路層

確定了0和1的分組方式。

現在通過電線我能發數據流了，但是，我還希望通過無線電波，通過其它介質來傳輸。然後我還要保證傳輸過去的比特流是正確的，要有糾錯功能。

於是，發明了數據鏈路層：定義瞭如何讓格式化數據以進行傳輸，以及如何讓控制對物理介質的訪問。這一層通常還提供錯誤檢測和糾正，以確保數據的可靠傳輸。

1.3、網絡層

它的作用是引進一套新的地址，使得我們能夠區分不同的計算機是否屬於同一個子網絡。這套地址就叫做"網絡地址"，簡稱"網址"。

傳輸層只是解決了打包的問題。但是如果我有多台計算機，怎麼找到我要發的那台？或者，A要給F發信息，中間要經過B，C，D,E，但是中間還有好多節點如KJZY。我怎麼選擇最佳路徑？這就是路由要做的事。

於是，發明了網絡層。即路由器，交換機那些具有尋址功能的設備所實現的功能。這一層定義的是IP地址，通過IP地址尋址。所以產生了IP協議。

1.4、傳輸層

"傳輸層"的功能，就是建立"端口到端口"的通信。相比之下，"網絡層"的功能是建立"主機到主機"的通信。只要確定主機和端口，我們就能實現程序之間的交流。

現在我能發正確的發比特流數據到另一台計算機了，但是當我發大量數據時候，可能需要好長時間，例如一個視頻格式的，網絡會中斷好多次（事實上，即使有了物理層和數據鏈路層，網絡還是經常中斷，只是中斷的時間是毫秒級別的）。那麼，我還須要保證傳輸大量文件時的準確性。於是，我要對發出去的數據進行封裝。就像發快遞一樣，一個個地發。

於是，先發明了傳輸層（傳輸層在OSI模型中，是在網絡層上面）。例如TCP，是用於發大量數據的，我發了1萬個包出去，另一台電腦就要告訴我是否接受到了1萬個包，如果缺了3個包，就告訴我是第1001，234，8888個包丟了，那我再發一次。這樣，就能保證對方把這個視頻完整接收了。

例如UDP，是用于发送少量数据的。我发20个包出去，一般不会丢包，所以，我不管你收到多少个。在多人互动游戏，也经常用UDP协议，因为一般都是简单的信息，而且有广播的需求。如果用TCP，效率就很低，因为它会不停地告诉主机我收到了20个包，或者我收到了18个包，再发我两个！如果同时有1万台计算机都这样做，那么用TCP反而会降低效率，还不如用UDP，主机发出去就算了，丢几个包你就卡一下，算了，下次再发包你再更新。TCP协议是会绑定IP和端口的协议，下面会介绍IP协议。

1.5、会话层

会话层的作用就是建立和管理应用程序之间的通信。

现在我们已经保证给正确的计算机，发送正确的封装过后的信息了。但是用户级别的体验好不好？难道我每次都要调用TCP去打包，然后调用IP协议去找路由，自己去发？当然不行，所以我们要建立一个自动收发包，自动寻址的功能。

于是，发明了会话层。会话层的作用就是建立和管理应用程序之间的通信。

1.6、表示层



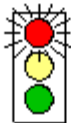

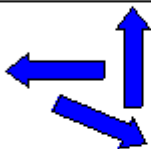

帮我们解决不同系统之间的通信语法问题。

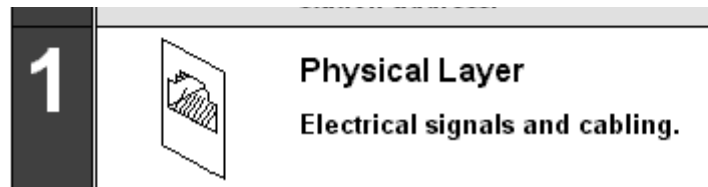
现在我能保证应用程序自动收发包和寻址了。但是我要用Linux给window发包，两个系统语法不一致，就像安装包一样，exe是不能在linux下用的，shell在window下也是不能直接运行的。于是需要表示层（presentation），帮我们解决不同系统之间的通信语法问题。

1.7、应用层

OK，现在所有必要条件都准备好了，我们可以写个android程序，web程序去实现需求把。



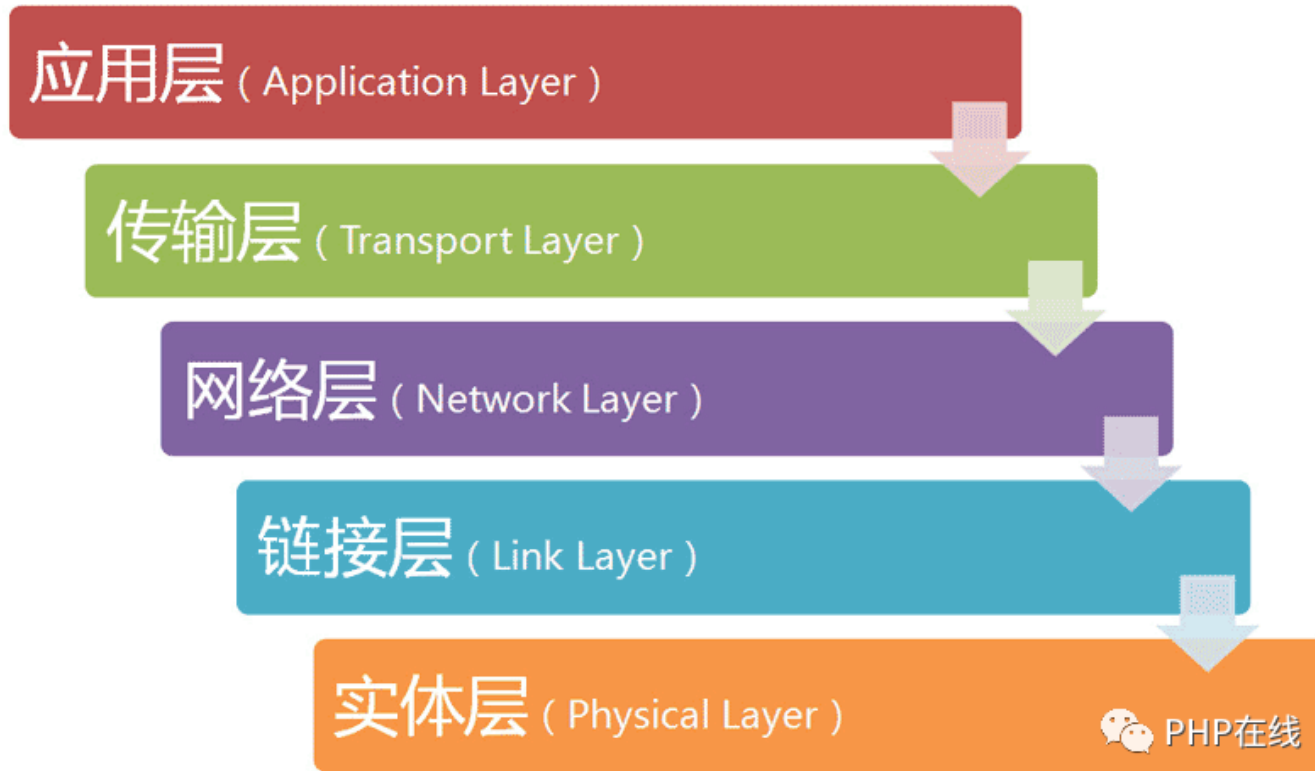
OSI MODEL			TCP / IP
7	 Application Layer Type of communication: E-mail, file transfer, client/server.		FTP, Telnet, HTTP, SNMP, DNS, OSPF, RIP, Ping, Traceroute
6	 Presentation Layer Encryption, data conversion: ASCII to EBCDIC, BCD to binary, etc.		
5	 Session Layer Starts, stops session. Maintains order.		
4	 Transport Layer Ensures delivery of entire file or message.		TCP (delivery ensured) UDP (delivery NOT ensured)
3	 Network Layer Routes data to different LANs and WANs based on network address.		IP (ICMP, IGMP, ARP, RARP)
2	 Data Link (MAC) Layer Transmits packets from node to node based on station address.		

 PHP在线

二、概述

2.1、五层模型

互联网的核心是一系列协议，总称为"互联网协议" (Internet Protocol Suite)。它们对电脑如何连接和组网，做出了详尽的规定。理解了这些协议，就理解了互联网的原理。互联网的实现，分成好几层。每一层都有自己的功能，就像建筑物一样，每一层都靠下一层支持。用户接触到的，只是最上面的一层，根本没有感觉到下面的层。要理解互联网，必须从最下层开始，自下而上理解每一层的功能。如何分层有不同的模型，有的模型分七层，有的分四层。我觉得，把互联网分成五层，比较容易解释。



如上图所示，最底下的一层叫做"实体层" (Physical Layer)，最上面的一层叫做"应用层" (Application Layer)，中间的三层（自下而上）分别是"链接层" (Link Layer)、"网络层" (Network Layer) 和"传输层" (Transport Layer)。越下面的层，越靠近硬件；越上面的层，越靠近用户。它们叫什么名字，其实并不重要。只需要知道，互联网分成若干层就可以了。

2.2、层与协议

每一层都是为了完成一种功能。为了实现这些功能，就需要大家都遵守共同的规则。

大家都遵守的规则，就叫做"协议" (protocol)。互联网的每一层，都定义了很多协议。

这些协议的总称，就叫做"互联网协议" (Internet Protocol Suite)。它们是互联网的核心，下面介绍每一层的功能，主要就是介绍每一层的主要协议。

2.3、实体层

我们从最底下的一层开始。电脑要组网，第一件事要干什么？当然是先把电脑连起来，可以用光缆、电缆、双绞线、无线电波等方式。



这就叫做"实体层"，它就是把电脑连接起来的物理手段。它主要规定了网络的一些电气特性，作用是负责传送0和1的电信号。

三、链接层

3.1、定义

单纯的0和1没有任何意义，必须规定解读方式：多少个电信号算一组？每个信号位有何意义？

这就是"链接层"的功能，它在"实体层"的上方，确定了0和1的分组方式。

3.2、以太网协议

早期的时候，每家公司都有自己的电信号分组方式。逐渐地，一种叫做"以太网" (Ethernet) 的协议，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" (Frame)。每一帧分成两个部分：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、接受者、数据类型等等；"数据"则是数据包的具体内容。

"标头"的长度，固定为18字节。"数据"的长度，最短为46字节，最长为1500字节。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。

- IEEE 802.1——通用网络概念及网桥等
- IEEE 802.2——逻辑链路控制等
- IEEE 802.3——以太网
- IEEE 802.4——ARCnet总线结构及访问方法,物理层规定
- IEEE 802.5——Token Ring访问方法及物理层规定等
- IEEE 802.6——城域网的访问方法及物理层规定
- IEEE 802.7——宽带局域网
- IEEE 802.8——光纤局域网(FDDI)
- IEEE 802.9——ISDN局域网
- IEEE 802.10——网络的安全
- IEEE 802.11——无线局域网

3.3、MAC地址

上面提到，以太网数据包的"标头"，包含了发送者和接受者的信息。那么，发送者和接受者是如何标识呢？

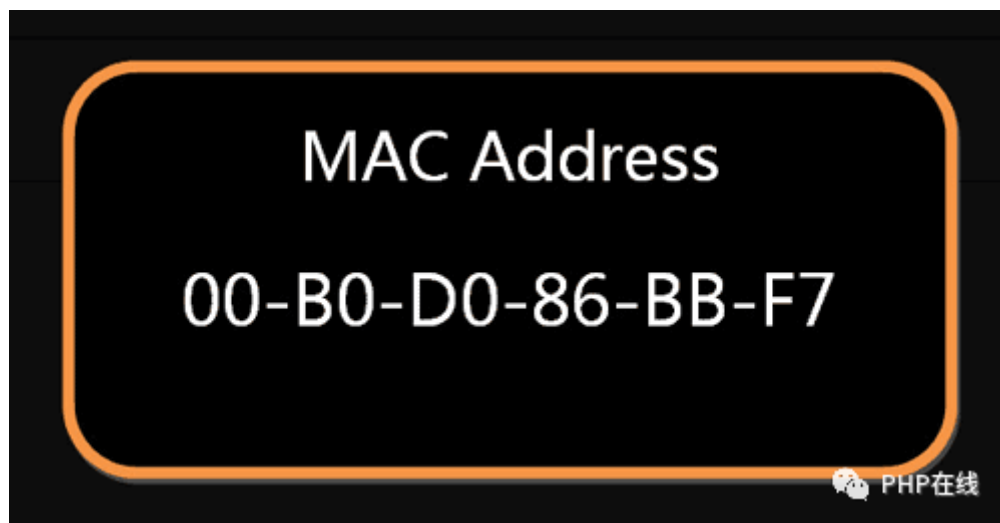
以太网规定，连入网络的所有设备，都必须具有"网卡"接口。数据包必须是从一块网卡，传送到另一块网卡。网卡的地址，就是数据包的发送地址和接收地址，这叫做MAC地址。

MAC地址 (Media Access Control Address)，直译为媒体访问控制地址，也称为局域网地址 (LAN Address)，以太网地址 (Ethernet Address) 或物理地址 (Physical Address)，它是一个用来确认网上设备位置的地址。



PHP在线

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。



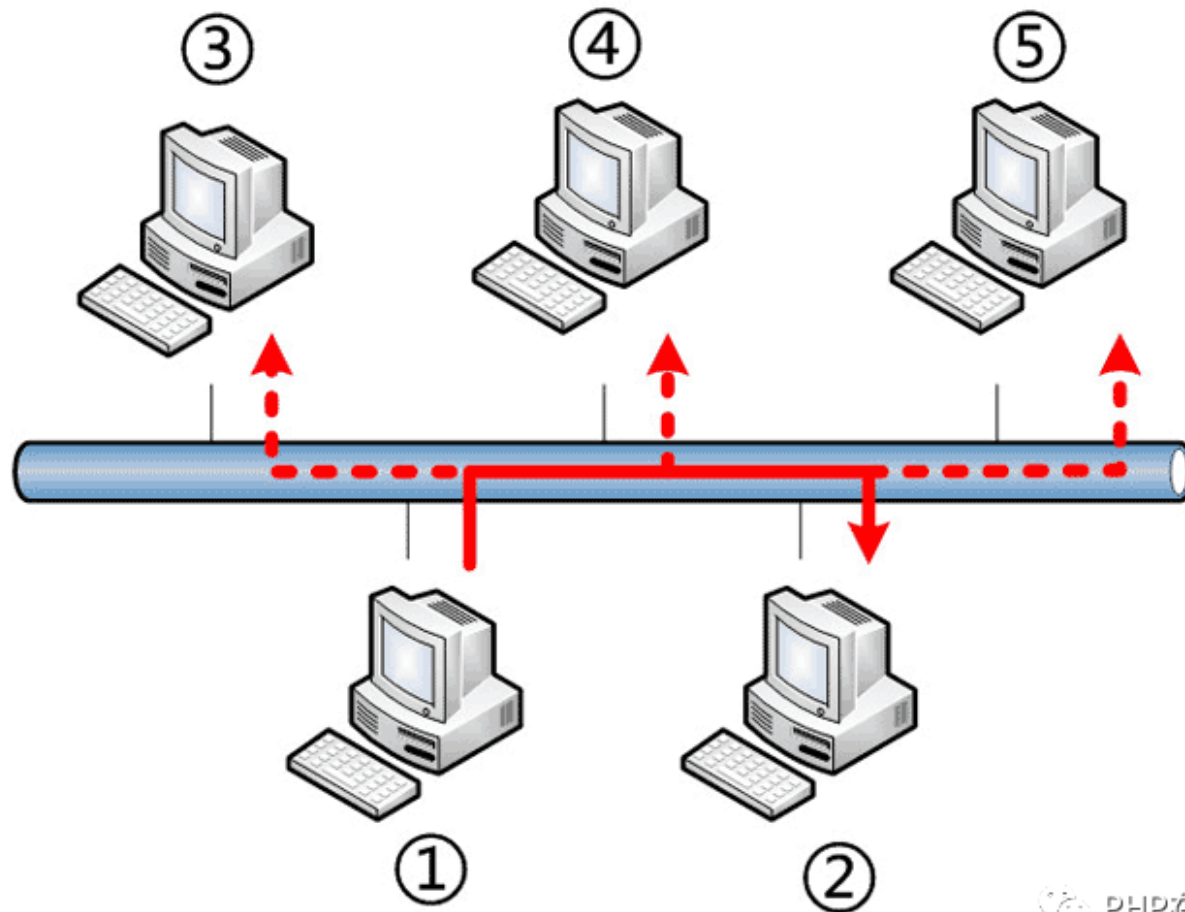
PHP在线

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。

3.4、广播

定义地址只是第一步，后面还有更多的步骤。

- 首先，一块网卡怎么会知道另一块网卡的MAC地址？回答是有一种ARP协议，可以解决这个问题。这个留到后面介绍，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？回答是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)。

有了数据包的定义、网卡的MAC地址、广播的发送方式，"链接层"就可以在多台计算机之间传送数据了。

计算机网络的基本分类方法主要有两种：

- 一种是根据网络所使用的传输技术：计算机网络可以分为广播式网络和点对点式网络
- 另一种是根据覆盖范围与规模：计算机网络可以分为局域网、城域网和广域网。

3.5、PPP 协议

在数据链路层有两个重要的协议，即HDLC协议和PPP协议。

- 高级数据链路控制 (High-Level Data Link Control或简称HDLC)，是一个在同步网上传输数据、面向比特的数据链路层协议，它是由国际标准化组织制订的。

HDLC协议是面向比特的，而PPP协议则是面向字节的，HDLC的帧采用开头跟结尾都是01111110作为帧的边界，这样当接收方接收到一串比特的时候可以根据它来判断该帧从哪里开始，到哪里结束，但是，假如在两个标志字段之间的比特串中恰好出现了01111110比特串，那该怎么办呢，HDLC采用零比特填充法，所谓零比特填充法就是每当出现5个1的时候就给它添加一个0进去，而接收方接收到数据时凡出现5个1的时候去掉其后面一个0，这样就能很好地确定帧。

- 点对点协议 (英语：Point-to-Point Protocol，PPP) 工作在数据链路层 (以OSI参考模型的观点)。它通常用在两节点间建立直接连接，并可以提供连接认证、传输加密 (使用ECP，RFC 1968) 以及压缩。

PPP协议本来也是跟HDLC协议一样，把01111110作为边界符 (一般称为标志符)，但是因为PPP协议是面向字节的，所以这里不说01111110，而是说用7E作为边界符。PPP协议在同步传输链路中也是采用零比特填充法，而在异步传输链路中则采用特殊的字符填充法。

- HDLC在控制字段中提供了可靠的确认机制，因此它可以实现可靠传输，而PPP则不提供可靠传输，要靠上层实现保证其正确性，因此，曾经在误码率比较高的链路中，HDLC曾起到了极大的作用，但随着技术的发展，在数据链路层出现差错的概率不大，因此现在全世界使用得最多的数据链路层协议是PPP协议。

3.6、交换机

交换机 (Switch) 意为“开关”是一种用于电 (光) 信号转发的网络设备。它可以为接入交换机的任意两个网络节点提供独享的电信号通路。最常见的交换机是以太网交换机。交换机工作于OSI参考模型的第二层，即数据链路层。交换机内部的CPU会在每个端口成功连接时，通过将MAC地址和端口对应，形成一张MAC表。在今后的通讯中，发往该MAC地址的数据包将仅送往其对应的端口，而不是所有的端口。因此，交换机可用于划分数据链路层广播，即冲突域；但它不能划分网络层广播，即广播域。

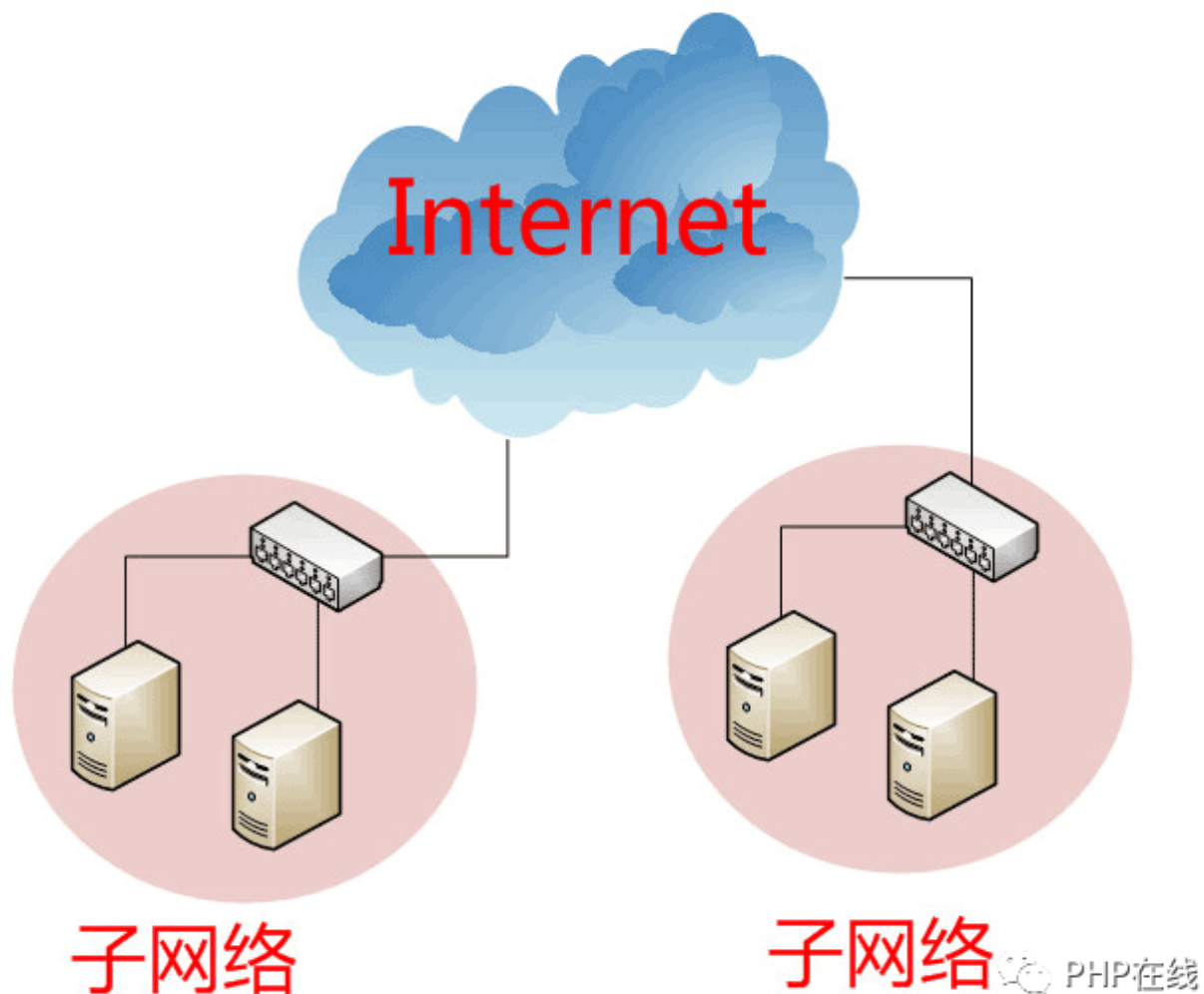
四、网络层

4.1、网络层的由来

以太网协议，依靠MAC地址发送数据。理论上，单单依靠MAC地址，上海的网卡就可以找到洛杉矶的网卡了，技术上是可以实现。

但是，这样做有一个重大的缺点。以太网采用广播方式发送数据包，所有成员人手一“包”，不仅效率低，而且局限在发送者所在的子网络。也就是说，如果两台计算机不在同一个子网络，广播是传不过去的。这种设计是合理的，否则互联网上每一台计算机都会收到所有包，那会引起灾难。

互联网是无数子网络共同组成的一个巨型网络，很像想象上海和洛杉矶的电脑会在同一个子网络，这几乎是不可能的。



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，这是一个很大的主题，本文不涉及。）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。

ICMP是（Internet Control Message Protocol）Internet控制报文协议。它是TCP/IP协议族的一个子协议，用于在IP主机、路由器之间传递控制消息。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。这些控制消息虽然并不传输用户数据，但是对于用户数据的传递起着重要的作用。常用的ping用到的就是该协议。

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”。

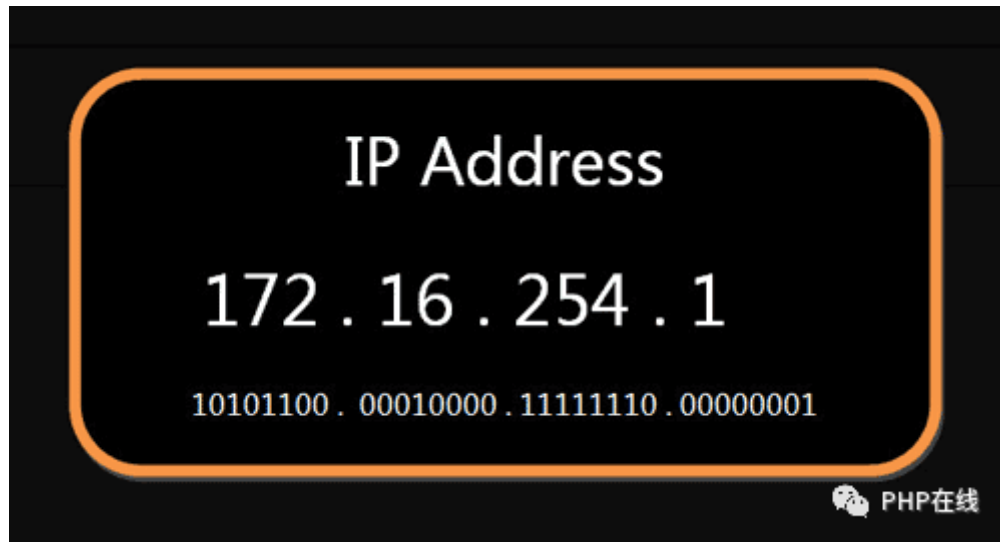
于是，“网络层”出现以后，每台计算机有了两种地址，一种是MAC地址，另一种是网络地址。两种地址之间没有任何联系，MAC地址是绑定在网卡上的，网络地址则是管理员分配的，它们只是随机组合在一起。

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。

4.2、IP协议

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。



习惯上，我们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。

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？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。

总结一下，IP协议的作用主要有两个，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。

- A类IP地址：一个A类IP地址由1字节的网络地址和3字节主机地址组成。以0开头，第一个字节范围：0~127 (1.0.0.0 - 126.255.255.255) ；
- B类IP地址：一个B类IP地址由2个字节的网络地址和2字节的主机地址组成。以10开头，第一个字节范围：128~191 (128.0.0.0 - 191.255.255.255) ；
- C类IP地址：一个C类IP地址由3字节的网络地址和1字节的主机地址组成。以110开头，第一个字节范围：192~223 (192.0.0.0 - 223.255.255.255) ；

4.3、IP数据包

根据IP协议发送的数据，就叫做IP数据包。不难想象，其中必定包括IP地址信息。

但是前面说过，以太网数据包只包含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，因此完全不用修改以太网的规格。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。

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。它放进以太网数据包后，以太网数据包就变成了下面这样。



IP数据包的"标头"部分的长度为20到60字节，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字节。前面说过，以太网数据包的"数据"部分，最长只有1500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。

4.4、ARP协议

关于"网络层"，还有最后一点需要说明。

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（后文会解释），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。

ARP 协议：地址解析协议，即ARP（Address Resolution Protocol），是根据IP地址获取物理地址的一个TCP/IP协议。

总之，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。

五、传输层

5.1、传输层的由来

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。

接下来的问题是，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，一边与朋友在线聊天。当一个数据包从互联网上发来的时候，你怎么知道，它是表示网页的内容，还是表示在线聊天的内容？

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。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，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。0到1023的端口被系统占用，用户只能选用大于1023的端口。不管是浏览网页还是在线聊天，应用程序会随机选用一个端口，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。相比之下，“网络层”的功能是建立“主机到主机”的通信。只要确定主机和端口，我们就能实现程序之间的交流。

因此，Unix系统就把主机+端口，叫做“套接字”（socket）。有了它，就可以进行网络应用程序开发了。

5.2、UDP协议

现在，我们必须在数据包中加入端口信息，这就需要新的协议。最简单的实现叫做UDP协议，它的格式几乎就是在数据前面，加上端口号。

UDP用户数据包协议（User Datagram Protocol），又称使用者资料包协议，是一个简单的面向数据报的传输层协议。

UDP数据包，也是由“标头”和“数据”两部分组成。



“标头”部分主要定义了发出端口和接收端口，“数据”部分就是具体的内容。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：



UDP数据包非常简单，“标头”部分一共只有8个字节，总长度不超过65,535字节，正好放进一个IP数据包。

5.3、TCP协议

UDP协议的优点是比较简单，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。

传输控制协议（英语：Transmission Control Protocol）是一种面向连接的、可靠的、基于字节流的传输层通信协议，由IETF的RFC 793定义。

因此，TCP协议能够确保数据不会遗失。它的缺点是过程复杂、实现困难、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数据包没有长度限制，理论上可以无限长，但是为了保证网络的效率，通常TCP数据包的长度不会超过IP数据包的长度，以确保单个TCP数据包不必再分割。

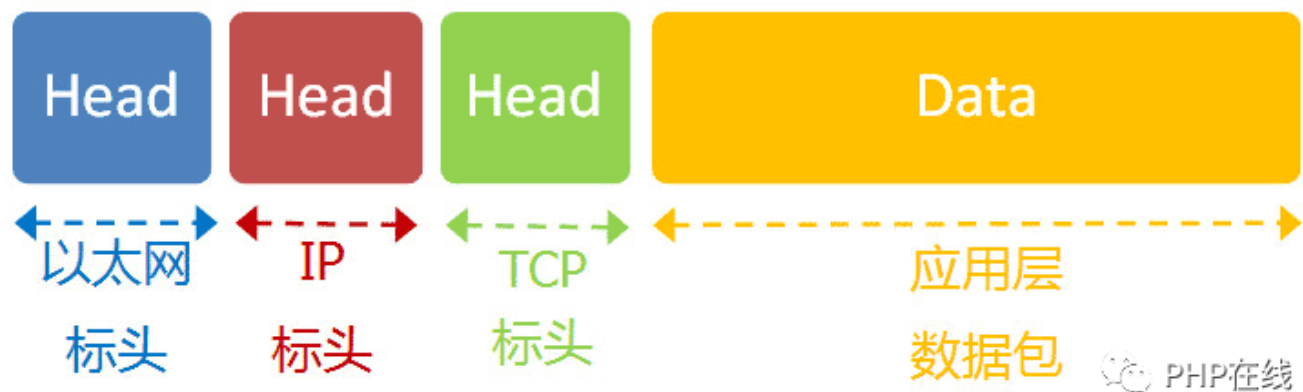
六、应用层

应用程序收到“传输层”的数据，接下来就要进行解读。由于互联网是开放架构，数据来源五花八门，必须事先规定好格式，否则根本无法解读。

“应用层”的作用，就是规定应用程序的数据格式。

举例来说，TCP协议可以为各种各样的程序传递数据，比如Email、WWW、FTP等等。那么，必须有不同协议规定电子邮件、网页、FTP数据的格式，这些应用程序协议就构成了“应用层”。

这是最高的一层，直接面对用户。它的数据就放在TCP数据包的“数据”部分。因此，现在的以太网的数据包就变成下面这样。



至此，整个互联网的五层结构，自下而上全部讲完了。这是从系统的角度，解释互联网是如何构成的。接下来，从用户的角度，自上而下看看这个结构是如何发挥作用，完成一次网络数据交换的。

七、一个小结

7.1、小结

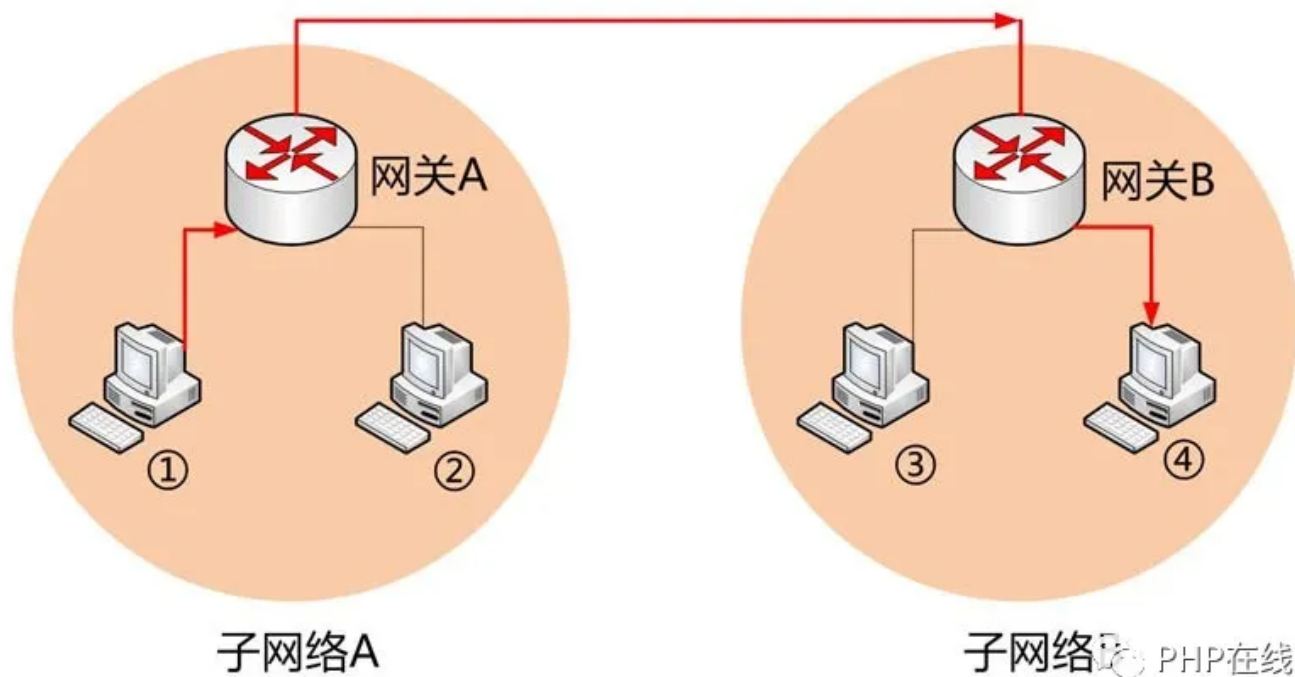
先对前面的内容，做一个小结。

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，基本上是上图那样子。

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。

7.2、网关协议

- 内部网关协议是在自治系统内部运行的路由协议，主要包括：RIP、EIGRP、OSPF、ISIS；

- 外部网关协议是在自治系统之间使用的路由协议，有BGP；

八、用户的上网设置

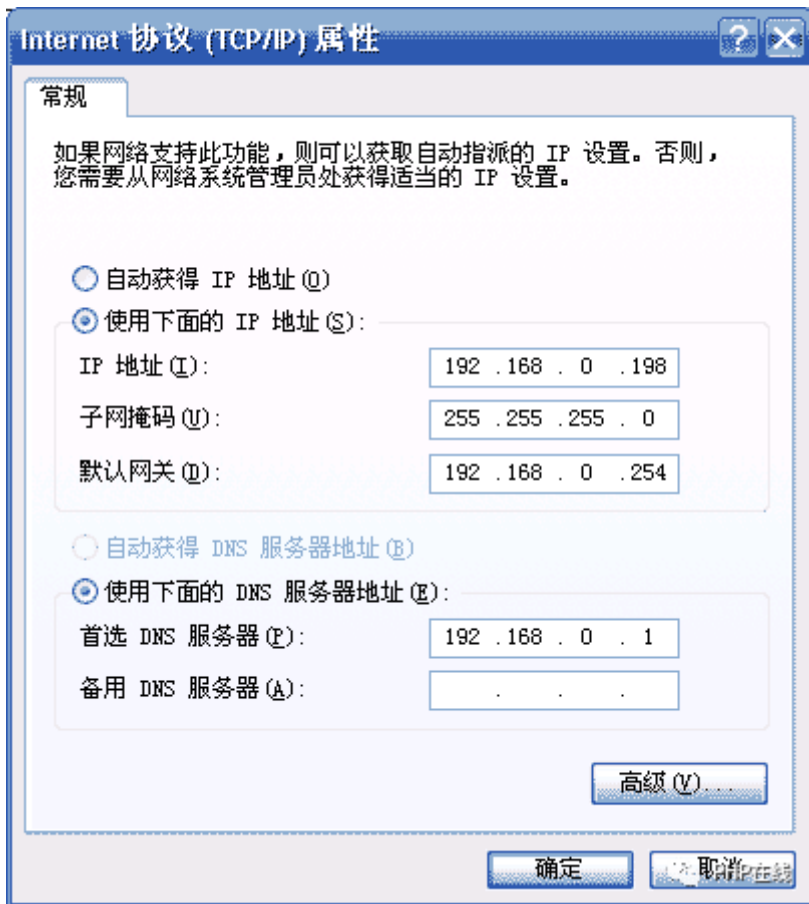
8.1 、静态IP地址

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"。

8.2、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。

动态主机设置协议（英语：Dynamic Host Configuration Protocol，DHCP）是一个局域网的网络协议，使用UDP协议工作，主要有两个用途：用于内部网或网络服务提供商自动分配IP地址；给用户用于内部网管理员作为对所有计算机作中央管理的手段。

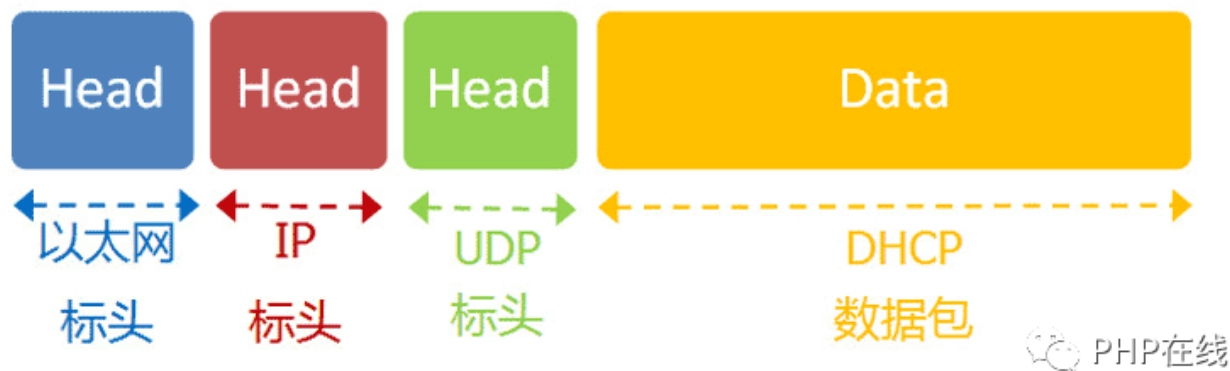
这个协议规定，每一个子网络中，有一台计算机负责管理本网络的所有IP地址，它叫做"DHCP服务器"。新的计算机加入网络，必须向"DHCP服务器"发送一个"DHCP请求"数据包，申请IP地址和相关的网络参数。

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。

8.3、DHCP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：



- 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。
- 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。
- 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。

8.4、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。

九、一个实例：访问网页

9.1、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。

9.2、DNS协议

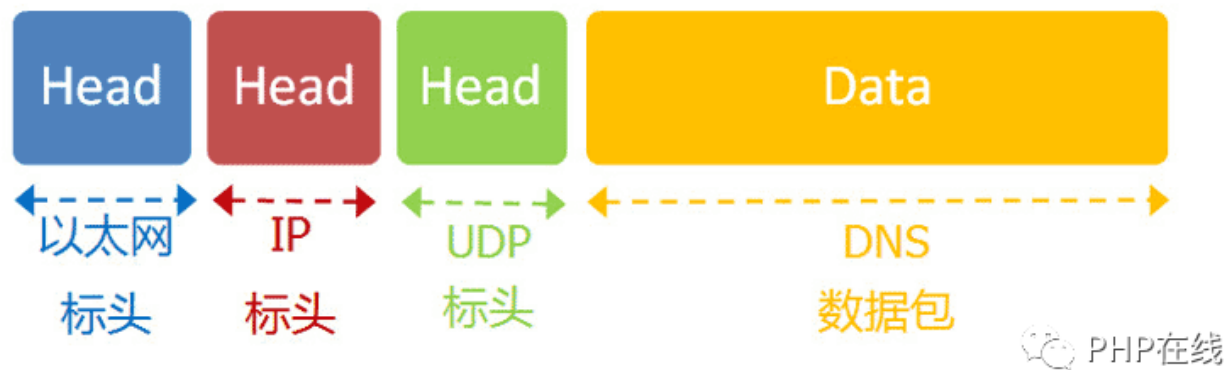
我们知道，发送数据包，必须要知道对方的IP地址。但是，现在，我们只知道网址www.google.com，不知道它的IP地址。

网域名称系统（英文：Domain Name System，缩写：DNS）是互联网的一项服务。它作为将域名和IP地址相互映射的一个分布式数据库，能够使人更方便地访问互联网。DNS使用TCP和UDP端口53。

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）。

当一个主机请求查询域名时：

- 先查本地缓存
- 没有缓存请求本地域名服务器
- 本地域名服务器没有请求根域名服务器



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。

9.3、子网掩码

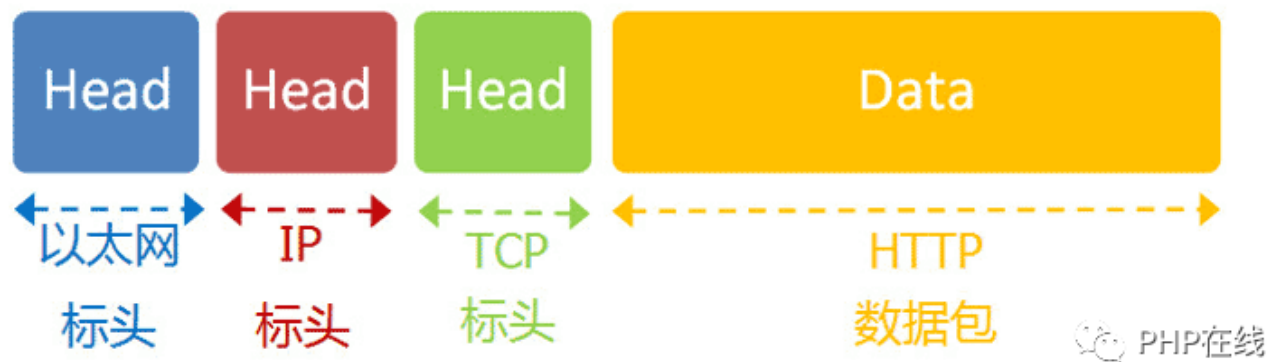
接下来，我们要判断，这个IP地址是不是在同一个子网络，这就要用到子网掩码。

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。

9.4、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: www.google.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 6.1) .....
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
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。

9.5、TCP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。

9.6、IP协议

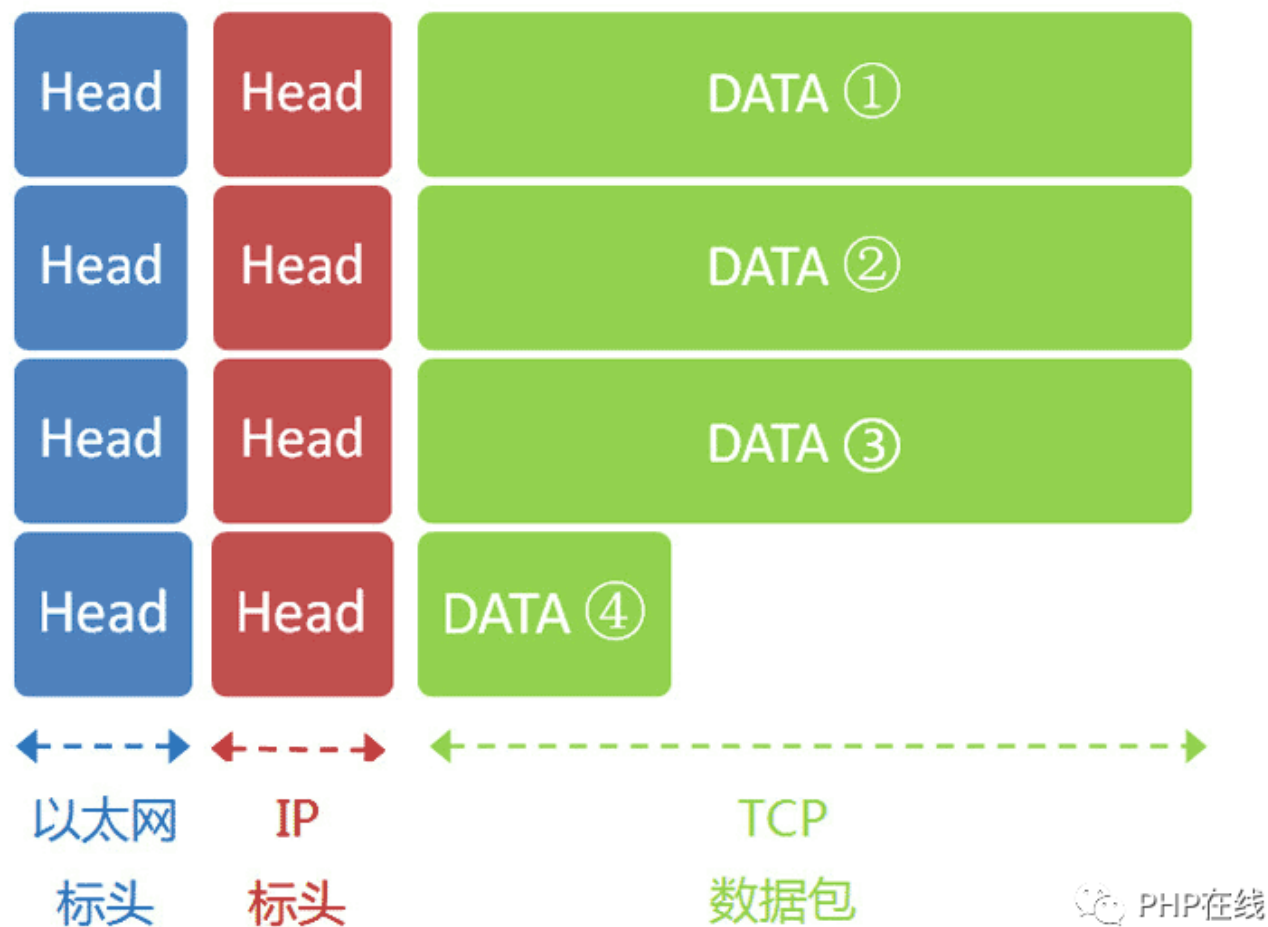
然后，TCP数据包再嵌入IP数据包。IP数据包需要设置双方的IP地址，这是已知的，发送方是192.168.1.100（本机），接收方是172.194.72.105（Google）。

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。

9.7、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。



9.8、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。

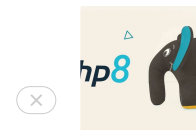
这个例子就到此为止，虽然经过了简化，但它大致上反映了互联网协议的整个通信过程。



喜欢此内容的人还喜欢

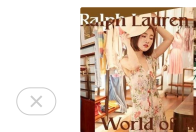
认识PHP8

PHP在线



神仙地儿：一次满足你三个愿望 | @World of Ralph Lauren

周小晨



成都49中男生墜亡，33分鐘監控曝光：這一點，才是最大的真相！

木棉說



