# 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？-网络编程/专项技术区 - 即时通讯开发者社区!


## 1、前言

标题虽然是为了解释有了 IP 地址，为什么还要用 MAC 地址，但是本文的重点在于理解为什么要有 IP 这样的东西。本文对读者的定位是知道 MAC 地址是什么，IP 地址是什么。

## 2、关于作者


**翟志军**，个人博客地址：https://showme.codes/，Github：https://github.com/zacker330。感谢作者的原创分享。

作者的另一篇《即时通讯安全篇（七）：如果这样来理解HTTPS，一篇就够了》也写的非常好，有兴趣的读者可以深读之。

## 3、系列文章

**本文是系列文章中的第9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）》
- 《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？》（本文）

**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》

# 4、书上说的

***基本概念：***

如今的网络是分层来实现的，就像是搭积木一样，先设计某个特定功能的模块，然后把模块拼起来组成整个网络。局域网也不例外，一般来说，在组网上我们使用的是IEEE802参考模型，从下至上分为：物理层、媒体接入控制层（MAC），逻辑链路控制层（LLC）。

标识网络中的一台计算机，一般至少有三种方法，最常用的是域名地址、IP地址和MAC地址，分别对应应用层、网络层、物理层。网络管理一般就是在网络层针对IP地址进行管理，但由于一台计算机的IP地址可以由用户自行设定，管理起来相对困难，MAC地址一般不可更改，所以把IP地址同MAC地址组合到一起管理就成为常见的管理方式。

***什么是MAC地址?***

MAC地址就是在媒体接入层上使用的地址，也叫物理地址、硬件地址或链路地址，由网络设备制造商生产时写在硬件内部。MAC地址与网络无关，也即无论将带有这个地址的硬件（如网卡、集线器、路由器等）接入到网络的何处，都有相同的MAC地址，它由厂商写在网卡的BIOS里。MAC地址可采用6字节（48比特）或2字节（16比

特）这两种中的任意一种。但随着局域网规模越来越大，一般都采用6字节的MAC地址。这个48比特都有其规定的意义，前24位是由生产网卡的厂商向IEEE申请的厂商地址，目前的价格是1000美元买一个地址块，后24位由厂商自行分配，这样的分配使得世界上任意一个拥有48位MAC 地址的网卡都有唯一的标识。另外，2字节的MAC地址不用网卡厂商申请。

MAC地址通常表示为12个16进制数，每2个16进制数之间用冒号隔开，如：08:00:20:0A:8C:6D就是一个MAC地址，其中前6位16进制数08:00:20代表网络硬件制造商的编号，它由IEEE分配，而后6位16进制数0A:8C:6D代表该制造商所制造的某个网络产品（如网卡）的系列号。每个网络制造商必须确保它所制造的每个以太网设备都具有相同的前三字节以及不同的后三个字节。这样就可保证世界上每个以太网设备都具有唯一的MAC 地址。

***什么是IP地址？***

IP地址是指互联网协议地址（英语：Internet Protocol Address，又译为网际协议地址），是IP Address的缩写。IP地址是IP协议提供的一种统一的地址格式，它为互联网上的每一个网络和每一台主机分配一个逻辑地址，以此来屏蔽物理地址的差异。

***为什么要用到MAC地址？***

这是由组网方式决定的，如今比较流行的接入Internet的方式（也是未来发展的方向）是把主机通过局域网组织在

一起，然后再通过交换机和 Internet相连接。这样一来就出现了如何区分具体用户，防止盗用的问题。由于IP只是逻辑上标识，任何人都随意修改，因此不能用来标识用户；而 MAC地址则不然，它是固化在网卡里面的。从理论上讲，除非盗来硬件（网卡），否则是没有办法冒名顶替的（注意：其实也可以盗用，后面将介绍）。

基于MAC地址的这种特点，局域网采用了用MAC地址来标识具体用户的方法。注意：具体实现：在交换机内部通过“表”的方式把MAC地址和IP地址一一对应，也就是所说的IP、MAC绑定。

**具体的通信方式：**接收过程，当有发给本地局域网内一台主机的数据包时，交换机接收下来，然后把数据包中的IP地址按照“表”中的对应关系映射成MAC地址，转发到对应的MAC地址的主机上，这样一来，即使某台主机盗用了这个IP地址，但由于他没有这个MAC地址，因此也不会收到数据包。发送过程和接收过程类似，限于篇幅不叙述。

综上可知，只有IP而没有对应的MAC地址在这种局域网内是不能上网的，于是解决了IP盗用问题。

***IP地址与MAC地址的区别是什么？***

IP地址基于逻辑，比较灵活，不受硬件限制，也容易记忆。MAC地址在一定程度上与硬件一致，基于物理，能够标识具体。这两种地址各有好处，使用时也因条件而采取不同的地址。

***MAC地址涉及到的安全问题：***

从上面的介绍可以知道，这种标识方式只是MAC地址基于的，如果有人能够更改MAC地址，就可以盗用IP免费上网了，目前网上针对小区宽带的盗用MAC地址免费上网方式就是基于此这种思路。如果想盗用别人的IP地址，除了IP地址还要知道对应的MAC地址。举个例子，获得局域网内某台主机的MAC地址，比如想得到局域网内名为TARGET主机的MAC地址，先用PING命令：PING TARGET，这样在我们主机上面的ARP表的缓存中就会留下目标地址和MAC映射的记录，然后通过ARP A命令来查询ARP表，这样就得到了指定主机的MAC地址。最后用ARP -s IP 网卡MAC地址，命令把网关的IP地址和它的MAC地址映射起来就可以了。

如果要得到其它网段内的MAC地址，那么可以用工具软件来实现，我觉得Windows优化大师中自带的工具不错，点击*“系统性能优化”→“系统安全优化”→“附加工具”→“集群Ping”*，可以成批的扫出MAC地址并可以保存到文件。

**小知识：**

ARP(Address Resolution Protocol)是地址解析协议，ARP是一种将IP地址转化成物理地址的协议。从IP地址到物理地址的映射有两种方式：表格方式和非表格方式。ARP 具体说来就是将网络层（IP层，也就是相当于OSI的第三层）地址解析为数据连接层（MAC层，也就是相当于OSI的第二层）的MAC地址。ARP协议是通过IP地址来获得MAC地址的。

**ARP原理：**

要向主机B发送报文，会查询本地的ARP缓存表，找到B的IP地址对应的MAC地址后就会进行数据传输。如果未找到，则广播A一个 ARP请求报文（携带主机A的IP地址Ia——物理地址Pa），请求IP地址为Ib的主机B回答物理地址Pb。网上所有主机包括B都收到ARP请求，但只有主机B识别自己的IP地址，于是向A主机发回一个ARP响应报文。其中就包含有B的MAC地址，A接收到B的应答后，就会更新本地的ARP缓存。接着使用这个MAC地址发送数据（由网卡附加MAC地址）。

因此，本地高速缓存的这个ARP表是本地网络流通的基础，而且这个缓存是动态的。ARP表：为了回忆通信的速度，最近常用的MAC地址与IP的转换不用依靠交换机来进行，而是在本机上建立一个用来记录常用主机IP－MAC映射表，即ARP表。

# 5、最通俗的解释

看完上一节中各种书籍里对IP地址、MAC地址的理解介绍和说明，还是很蒙逼，那么请继续看完本节吧。

## 5.1网络洪荒时代

一开始时，网络中的机器并不多。大家都连到同一个集线器就可以了，就可以实现互通。这时，机器 A 发消息到机

器 B，消息头里附上机器 B 的MAC，集线器收到消息后就广播给所有连到集线器的机器。

机器 C 收到消息，发现消息里的 MAC 地址和自己的不一样，就丢弃。机器B发现消息里的 MAC 地址和自己一样，就收到下并解析。


这样机制带来问题很明显：首先每次广播，给所在网络带来不必要的浪费。所以，就出现了交换机。它能识别消息里的目标 MAC 地址后，直接就消息丢到机器 B 所连接的端口中。另一个角度，交换机必须记住所有的 MAC 地址和端口之间的关系。

这样的机制在网络规规模小的时候是高效的。但是当网络规模扩大到全球的时候，不可能让一台交换机记录下全球这么多的网络设备，也不可能让全球的机器连接到一台交换机上。

## 5.2如果是多台交换机呢?

想像一下，你是斯坦福的学生，你的电脑 x 的网络直连的是学校的交换机，而学校的交换机又连美国国家网络交换机。而美国国家网络交换机又直接的是中国国家网络交换机，中国服务器 y 直连的是中国国家交换机。

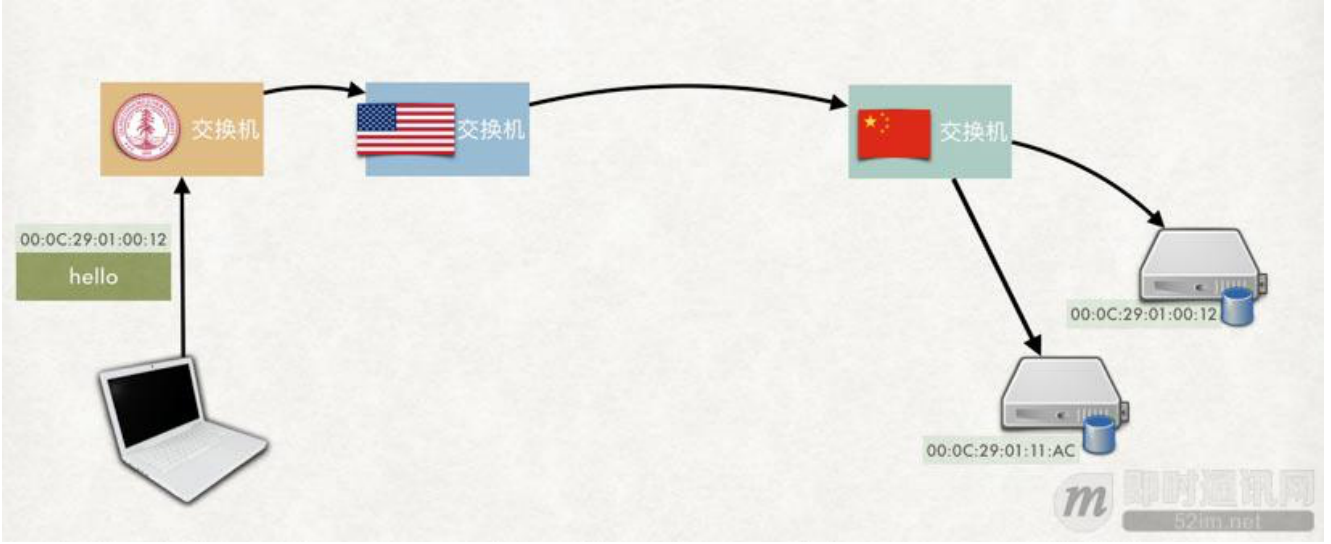
你想访问中国的服务器 y 中的资源。你了解到服务器 y 的 MAC 地址是00:0C:29:01:00:12，所以你在消息里附上这个 MAC 地址。

学校交换机收到消息后，拿到 MAC 地址后就愣了，这是要发给谁啊？因为中国服务器 y 并不是直连学校交换机的。这时，学校交换机有一个选择，就是收到不明的 MAC 地址时，一律转发给默认端口。斯坦福交换机就将消息转给美国国家交换机。

美国国家交换机同样发愣了，因为没有这条 MAC 地址对应的端口。它又直接向默认端口：中国国家网络交换机。

中国国家网络交换机收到消息，发现自己记录了 MAC 地址 对应的是服务器 y。就直接将你这位斯坦福学生的消息转发到服务器 y 所连接的端口。

最终，我们的服务器 y 终于收到来自美国斯坦福学生的资源访问请求。




那么，我们的服务器 y 如何将相应的资源返回给学生呢？将消息中的源MAC 地址作为响应消息的目标 MAC 地址发送给中国国家交换机不就可以了？同样的机制，只不过是把源地址和目标地址反一下。

这下，我们是不是完美实现使用交换机组建美国网络和中国网络的互通？

但是美国和中国并不能代表全世界。其他国家也需要加入这个大网络。当日本国家交换机也接入美国国家交换机后，斯坦福学生的消息从学校到达美国国家交换机后就需要进行广播所有直连自己的端口了，因为这时，它没有对外的所谓默认端口了。这里有点烧脑，容各位同学一点时间思考。

## 5.3小结

也就是说，当两个网络互接时，MAC 地址 + 交换机还能

解决问题广播问题，但是两个以上的网络互连时，MAC 地址 + 交换机就没有办法解决广播问题了。

这时，我们面临的问题就是无法使用现有的技术—— MAC 地址 + 交换机——解决多网络互连的问题了。所以，需要发明一种新的技术。

而 IP 协议就是就是解决此问题的一项技术。

事实上，IP协议的产生并不只是为解决上述的“广播问题”。还解决了很多其他网络传输过程会遇到的问题，比如一次传输的消息过大时，如何对消息进行分组等问题。

**好了，如果以上内容你还是没有完全理解，那么以下3篇文章你必须好好读读（再不懂的话，真没救了..）：**

- 《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）》
- 《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》

# 6、写在最后

由于历史原因，MAC 地址及相关技术先出现，但是后来

发现它并不能解决所有（已知）的问题，所以，先驱们发明了 IP 地址及相关技术来解决。

另一个角度，个人认为，由于 MAC 地址没有办法表达网络中的子网的概念，而 IP 地址可以。如果网络互换设备（比如路由器）能从目标 MAC 地址中分析出目标网络，而不是只是目标主机，IP 地址还会出现吗？

**有另一个有趣的问题：**如果历史反过来，一开始就使用的是 IP 地址，而不是 MAC 地址，我们现在的网络世界会怎么样？

## 附录：更多网络编程方面的文章

**[1] 网络编程基础资料：**

《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》
《技术往事：改变世界的TCP/IP协议（珍贵多图、手机慎点）》
《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）》

《UDP中一个包的大小最大能多大？》
《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》
《技术扫盲：新一代基于UDP的低延时网络传输层协议——QUIC详解》
《让互联网更快：新一代QUIC协议在腾讯的技术实践分享》
《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》
《聊聊iOS中网络编程长连接的那些事》
《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》
《IPv6技术详解：基本概念、应用现状、技术实践（上篇）》
《IPv6技术详解：基本概念、应用现状、技术实践（下篇）》
《从HTTP/0.9到HTTP/2：一文读懂HTTP协议的历史演变和设计思路》
《以网游服务端的网络接入层设计为例，理解实时通信的技术挑战》
《迈向高阶：优秀Android程序员必知必会的网络基础》
>> 更多同类文章 ……

**[2] NIO异步网络编程资料：**

《Java新一代网络编程模型AIO原理及Linux系统AIO介绍》
《有关“为何选择Netty”的11个疑问及解答》
《开源NIO框架八卦——到底是先有MINA还是先有Netty?》
《选Netty还是Mina：深入研究与对比（一）》
《选Netty还是Mina：深入研究与对比（二）》
《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》
《Netty 4.x学习（一）：ByteBuf详解》
《Netty 4.x学习（二）：Channel和Pipeline详解》
《Netty 4.x学习（三）：线程模型详解》
《Apache Mina框架高级篇（一）：IoFilter详解》
《Apache Mina框架高级篇（二）：IoHandler详解》
《MINA2 线程原理总结（含简单测试实例）》
《Apache MINA2.0 开发指南（中文版）[附件下载]》
《MINA、Netty的源代码（在线阅读版）已整理发布》
《解决MINA数据传输中TCP的粘包、缺包问题（有源码）》
《解决Mina中多个同类型Filter实例共存的问题》
《实践总结：Netty3.x升级Netty4.x遇到的那些坑（线程篇）》
《实践总结：Netty3.x VS Netty4.x的线程模型》

《详解Netty的安全性：原理介绍、代码演示（上篇）》
《详解Netty的安全性：原理介绍、代码演示（下篇）》
《详解Netty的优雅退出机制和原理》
《NIO框架详解：Netty的高性能之道》
《Twitter：如何使用Netty 4来减少JVM的GC开销（译文）》
《绝对干货：基于Netty实现海量接入的推送服务技术要点》
《Netty干货分享：京东京麦的生产级TCP网关技术实践总结》
《新手入门：目前为止最透彻的的Netty高性能原理和框架架构解析》
>> 更多同类文章 ……