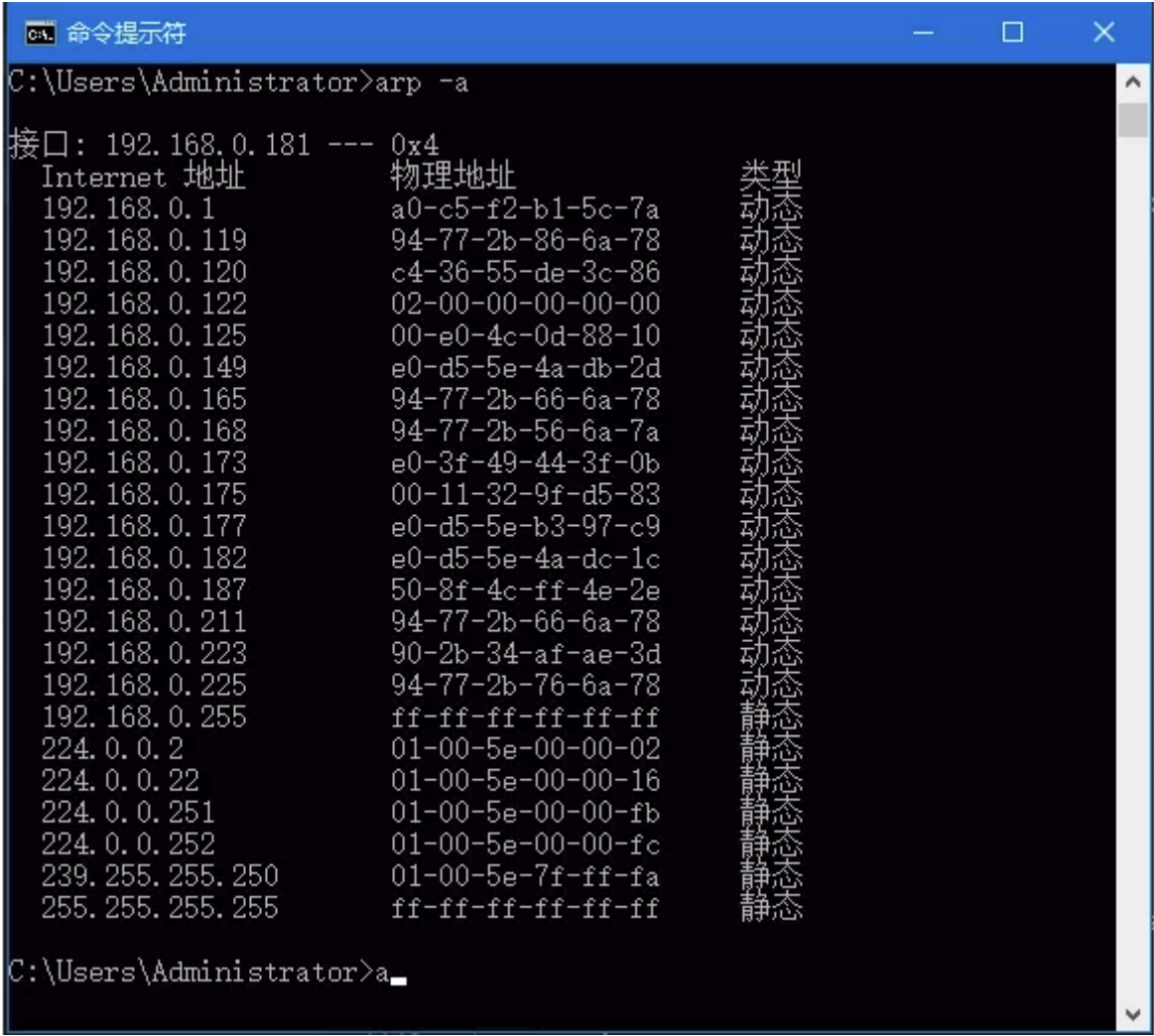


title: ARP协议原理 author: 杰杰 top: false cover: false toc: true mathjax: false date: 2019-10-15 22:56:12 img: coverImg: password: summary: tags: - TCP/IP - 网络 - LwIP categories: TCP/IP

引言

从一篇文章中，我们知道计算机中会维护一个ARP缓存表，这个表记录着IP地址与MAC地址的映射关系，我们可以通过在电脑的控制台通过arp -a指令查看一下我们自己计算机的ARP缓存表：



那么什么是ARP协议呢？

初识ARP

ARP协议是地址解析协议（Address Resolution Protocol）是通过解析IP地址得到MAC地址的，是一个在网络协议包中极其重要的网络传输协议，它与网卡有着极其密切的关系，在TCP/IP分层结构中，把ARP划分为网络层，为什么呢，因为在网络层看来，源主机与目标主机是通过IP地址进行识别的，而所有的数据传输又依赖网卡底层硬件，即链路层，那么就需要将这些IP地址转换为链路层可以识别的东西，在所有的链路中都有着自己的一套寻址机制，如在以太网中使用MAC地址进行寻址，以标识不同的主机，那么就需要有一个协议将IP地址转换为MAC地址，由此就出现了ARP协议，所有ARP协议在网络层被应用，它是网络层与链路层连接的重要枢

纽，每当有一个数据要发送的时候都需要在通过**ARP协议**将**IP地址转换成MAC地址**，在IP层及其以上的层次看来，他们只标识IP地址，从不跟硬件打交道，就像我一样，我做应用层的工作，而不会去写底层驱动，得专门有个同事将驱动写完给我，我只需要知道他提供的**API接口**就行了，而我就专心处理我的工作，我相信他能把驱动写好，我只需要直接调用即可。

ARP缓存表

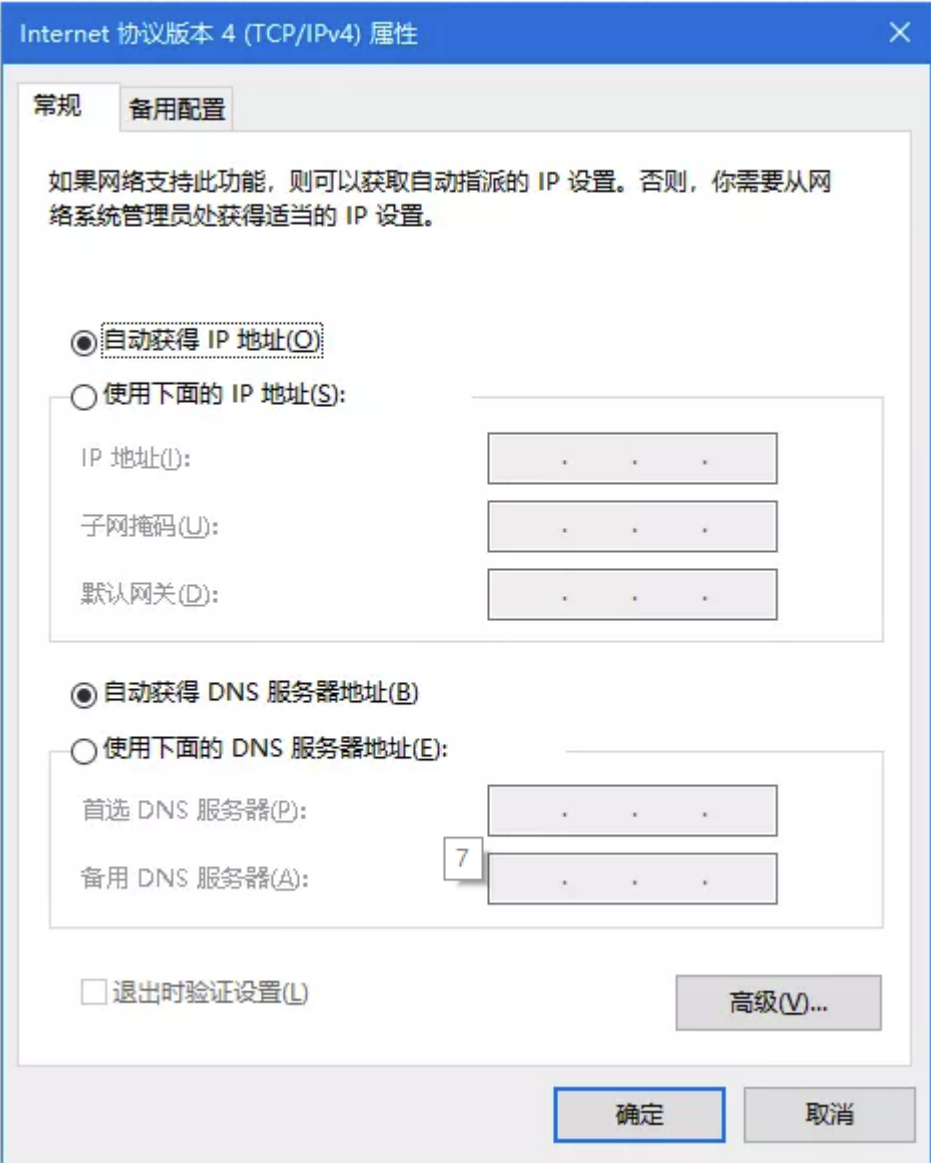
既然已经解释了**ARP协议**的用途及重要性，那么它是如何工作的？为了实现**IP地址与MAC地址的查询与转换**，**ARP协议**引入了**ARP缓存表**的概念，每台主机或路由器在**维护**着一个**ARP缓存表（ARP table）**，这个表包含**IP地址到MAC地址的映射关系**，表中记录了**<IP地址，MAC地址>**对，我称之为**ARP表项**，如我们前面那张图所展示的一样，他们是主机**最近**运行时获得关于其他主机的**IP地址到MAC地址**的映射，当需要发送数据的时候，主机就会根据数据报中的**目标IP地址**信息，然后在**ARP缓存表**中进行查找对应的**MAC地址**，最后通过网卡将数据发送出去。**ARP缓存表**包含一个**寿命值（TTL，也称作生存时间）**，它将记录每个**ARP表项**的生存时间，生存时间到了就会从缓存表中删除。从一个表项放置到**ARP缓存表**中开始，一个表项通常的生存时间一般是**10分钟**吗，当然，这些生存时间是可以任意设置的，我们一般使用默认即可。

一句话总结ARP协议的工作

ARP协议的主要工作就是建立、查询、更新、删除**ARP表项**。

ARP表项的建立

当主机开机的时候，**ARP缓存表**肯定是空的，那么怎么一步步**建立 ARP表项** 呢？如果此时想发送数据怎么办呢？因为没有**ARP表项**，说明就找不到**IP地址到MAC地址的映射关系**，这样子也就无法发送数据了。其实协议栈的实现还是很强大的，如果没有**ARP表项**，那么主机就会去建立**ARP表项**。以我们的电脑为例，在开机的时候，我们的电脑肯定是没有**ARP表项**的（或者说应该说只有路由器的**ARP表项**，因为可能是动态获取IP地址）：

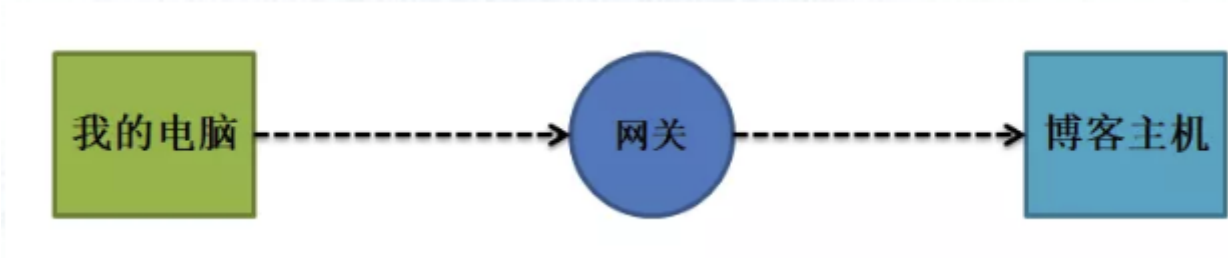


对于局域网

如果我想向局域网中的某个电脑发送一个数据，那么我的电脑就会从已有的ARP缓存表中寻找这个IP地址对应的物理地址的ARP表项，然后直接将数据写入以太网数据帧中让网卡进行发送即可，而如果没有找到这个IP地址，那么这个数据就没法立即发送，电脑会先在局域网广播一个ARP请求（目标MAC地址为FF-FF-FF-FF-FF-FF），广播的ARP请求发出后，处于同一局域网内的所有主机都会接收到这个请求，如果目标IP地址与接收到ARP请求的主机自身IP地址吻合就会返回一个ARP应答，告诉请求者（即我的电脑）自身的MAC地址，当我的电脑收到这个ARP应答后，就去建立一个ARP表项，并且重新将数据发送出去。ARP协议的核心就是对缓存表的操作，发送数据包的时候，查找ARP缓存表以得到对应的MAC地址，在ARP缓存表中的TTL即将过期的时候更新缓存表以保证ARP表项有效，此外ARP协议还需要不断处理来自局域网中其他主机的ARP请求。

对于公网

比如我的电脑向访问我的博客：<https://jiejetop.cn> 因为我的博客主机是处于外网的，那么我的电脑在访问的时候，也是找不到缓存表的，那它只能通过网关，让网关将数据转发到我的博客主机上，即：



因为我的电脑与博客主机不在一个网段，电脑查询自己的路由表，知道如果想和博客主机通信则必须通过网关（gateway）来中转，所以会在与网关直连的网卡上请求网关的MAC地址，因为电脑要把发给博客主机的数据先发给网关，当合法以太帧到达网关并且顺利接收，网关会将数据递交给IP层，IP层查询路由表，找到与博客主机直连的接口（假设是直连的，实际上肯定不是直连的），网关会发一个ARP请求到博客主机上，请求它的MAC地址，网关收到应答后将建立新的ARP表项并将开始维护ARP缓存表，然后完成最终的通信。

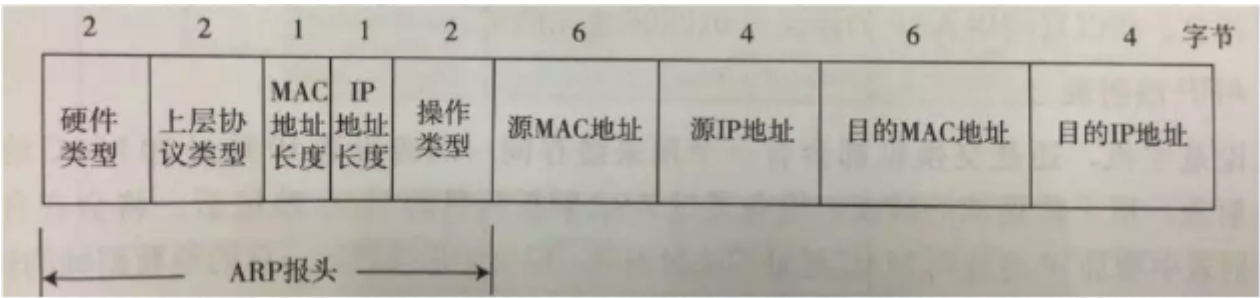
ARP缓存表的超时处理

ARP是动态处理的，现在总结一下：ARP表项的生存时间一般为5-10分钟（LwIP中默认是5分钟），而ARP请求的等待时间是5秒钟，当这些时间到达后，就会更新ARP表项，如果在物理链路层无法连通则会删除表项。因此每个协议栈的实现都必须维护着一个定时器（超时机制）来管理ARP缓存表，在必要的时候更新及删除ARP表项，关于怎么处理的我们就不多追究，有兴趣的可以看LwIP的etharp_tmr()源码。

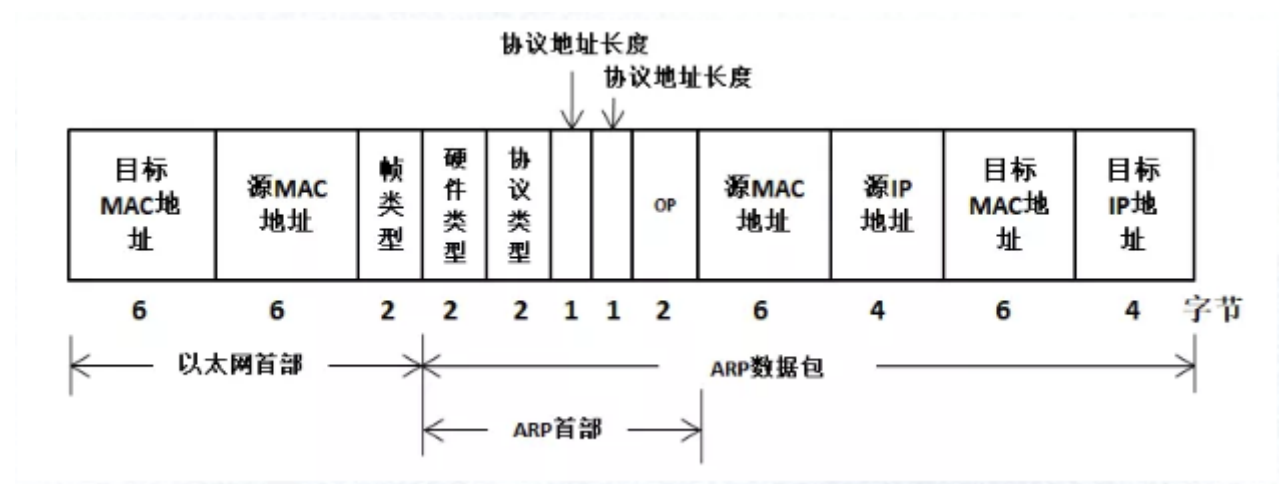
说点题外话：因为ARP协议是一个动态的协议，很多网络攻击都是利用ARP协议进行的，如ARP欺骗，ARP洪水攻击等等，而且这种攻击是很难防御的，当然也有办法，直接将动态的ARP缓存表设置为静态就行了，但是这就违背了ARP协议的动态地址解析特性。

ARP报文

ARP的请求与应答都是依赖ARP报文结构进行的，ARP报文是放在以太网数据帧中进行发送的，下面是ARP报文的格式：



当它封装在以太网帧中的格式：



在ARP表建立前，主机并不知道目标MAC地址，所以在一开始的时候只能通过广播的方式将ARP请求包发送出去，处于同一局域网的主机都能接收到广播的数据包。所以一开始目标MAC地址是FF-FF-FF-FF-FF-FF，而以太网首部的帧类型是有多种，对于ARP数据包来说，其值为0x0806，对于IP数据报来说，其值为0x0800。在ARP首部一开始的2个字节存储的是硬件类型，表示要知道目标网卡的硬件类型，其中，值为1表示以太网地址；接下来还有2字节的协议类型，其中，0x0800表示IP协议，其他还可能是ICMP/IGMP协议等；接下来有1个字节表示硬件地址长度，指出该报文中硬件地址的长度，对于以太网硬件类型，它的值为6；还有1字节的协议地址长度，如果是ARP协议、IP协议等，该值为4；ARP首部最后的op字段用于记录ARP操作的类型，分别是：

- ARP请求，其值为1。
- ARP应答，其值为2。
- RARP请求，其值为3。
- RARP应答，其值为4。

我们只关心ARP的请求与应答即可，RARP是逆地址解析协议，在这里我们就不用去了解，它在网络中基本已经被淘汰，用于主机在启动的时候获得自己的IP地址。对于ARP首部后面的四个字段分别是源MAC地址、源IP地址、目标MAC地址、目标IP地址，这些就是比较简单的了。为了加深理解，我们使用wireshark网络抓包工具形象地讲解报文格式与内容，关于wireshark网络抓包工具的使用方式我就不做过多讲解，网上教程一大把，打开工具，然后抓取电脑网络中的数据。

Wireshark · Packet 160 · 以太网

> Frame 160: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0

> Ethernet II, Src: OrayCom_01:5c:7a (a0:c5:f2:b1:5c:7a), Dst: Giga-Byt_b3:9a:cf (e0:d5:5e:b3:9a:cf)

> Destination: Giga-Byt_b3:9a:cf (e0:d5:5e:b3:9a:cf) 目标MAC地址

> Source: OrayCom_01:5c:7a (a0:c5:f2:b1:5c:7a) 源MAC地址

Type: ARP (0x0806) 帧类型

Padding: 00000000000000000000000000000000

以太网帧首部

> Address Resolution Protocol (request)

Hardware type: Ethernet (1) 硬件类型

Protocol type: IPv4 (0x0800) 协议类型

Hardware size: 6 硬件地址长度

Protocol size: 4 IP地址长度

Opcode: request (1) op

Sender MAC address: OrayCom_01:5c:7a (a0:c5:f2:b1:5c:7a) 源MAC地址

Sender IP address: 192.168.0.1 源MAC地址

Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00) 目标MAC地址

Target IP address: 192.168.0.181 目标IP地址

ARP报文

0000 e0 d5 5e b3 9a cf a0 c5 f2 b1 5c 7a 08 06 00 01 ...^.....\z....

0010 08 00 06 04 00 01 a0 c5 f2 b1 5c 7a c0 a8 00 01 \z....

0020 00 00 00 00 00 00 c0 a8 00 b5 00 00 00 00 00 00

0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Close

Help

Wireshark · Packet 161 · 以太网

> Frame 161: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

> Ethernet II, Src: Giga-Byt_b3:9a:cf (e0:d5:5e:b3:9a:cf), Dst: OrayCom_01:5c:7a (a0:c5:f2:b1:5c:7a)

> Destination: OrayCom_01:5c:7a (a0:c5:f2:b1:5c:7a) 目标MAC地址

> Source: Giga-Byt_b3:9a:cf (e0:d5:5e:b3:9a:cf) 源MAC地址

Type: ARP (0x0806) 帧类型

以太网帧首部

> Address Resolution Protocol (reply)

Hardware type: Ethernet (1) 硬件类型

Protocol type: IPv4 (0x0800) 协议类型

Hardware size: 6 硬件地址长度

Protocol size: 4 IP地址长度

Opcode: reply (2) op字段为2, 是ARP应答包

Sender MAC address: Giga-Byt_b3:9a:cf (e0:d5:5e:b3:9a:cf) 源MAC地址

Sender IP address: 192.168.0.181 源MAC地址

Target MAC address: OrayCom_01:5c:7a (a0:c5:f2:b1:5c:7a) 目标MAC地址

Target IP address: 192.168.0.1 目标IP地址

ARP报文

0000 a0 c5 f2 b1 5c 7a e0 d5 5e b3 9a cf 08 06 00 01\z...^.....

0010 08 00 06 04 00 02 e0 d5 5e b3 9a cf c0 a8 00 b5^.....

0020 a0 c5 f2 b1 5c 7a c0 a8 00 01\z...

Close

Help

未完待续... 下一篇LwIP中ARP协议的实现

喜欢就关注我吧！



相关代码可以在公众号后台获取。