

Chapter 2 – Link Layer *and* Chapter 4 – ARP

Elements

Logical link control
Medium access control
Physical medium

Ethernet

Historically meant:

10 MBits / Second
Bus interconnect
Baseband signaling
CSMA/CD MAC

Now refers to a collection of 10, 100, 1000 MBps standards that are capable of interoperating using Ethernet standard DIX or IEEE 802.2 frame formats.

Name and address mapping

Name resolution

Mapping: *jmw.cs.clemson.edu* → 130.127.48.24
Done by the "resolver" as part of DNS (to be discussed later).

MAC Address resolution

IP packets are routed using IP address *until* the dest LAN is reached.
Last hop is almost always over a shared medium (e-net, TR, ...) for economy
Final hop delivery is based on MAC address *not* IP address
That is, (IP) 130.127.48.24 must be mapped to (MAC)0:20:af:f:6f:3c
Done by ARP (Address Resolution Protocol)
after destination address is determined to be on *this* LAN

Logical Link Control Protocols (LLC's)

Two "standard" logical link level protocols (LLC's) have been used in the Ethernet

RFC 894 (DIX – Digital / Intel / Xerox)

IEEE 802.2 & IEEE 802.3 (RFC 1042)

RFC 894

Dest addr	Source addr	Packet type	Data	CRC
6	6	2	46–1500	4

Packet type

0x0800 IP Datagram
 0x0806 ARP Request / Reply
 0x8035 RARP Request / Reply

RFC 1042

<----- 802.2 LLC -----><-802.2 SNAP->									
Dest Addr	Source Addr	Length	DSAP 0xaa	SSAP 0xaa	Cntl 0x03	Org Code 0x00.	Type	Data	CRC
6	6	2	1	1	1	3	2	38–1492	4

Note that ambiguity is not possible because the large packet type fields are illegal lengths (unless large frames are in use!)

Packet type moves to SNAP (Sub-network access protocol) header in RFC 1042

Example RFC 1042 packet

```

00 b0 d0 24 fb 95 - dest
00 b0 d0 24 fc 27 - src
0016 - length
aa - dsap
aa - ssap
03 - ctl
000000 - org code
81 88 - type
62 6161 6161 6161 ..- data
    
```

ARP (Address resolution protocol)

Objectives of ARP

Map IP (network level) address to E-net (link level) address

Possible implementations:

Send *everything* by link broadcast and let the hosts sort it out.

Disadvantage: Lots of overhead on disinterested hosts

Send ARP requests for *every* packet.

Disadvantages: Lots of overhead on disinterested hosts and lots of excess network traffic

Send ARP requests only when address can't be resolved in ARP cache

A reasonable compromise because of locality of host references in network traffic.

Contents of the arp packet (Note that ARP assumes NEITHER IP nor ENet)

0	Hardware type	Ethernet for example
2	Protocol type –	IP for example
4	Hardware address len –	6
5	Protocol address len –	4
6	Operation	(1 = Arp req, 2 = Arp resp)
8	Sender hardware address	E-Net address
14	Sender protocol address	IP address
18	Target hardware address	E-Net address
24	Target protocol address	IP address

ARP implementation:

Each ARP request also contains the requestors IP and MAC addresses because the requestee will have to use it in the ARP reply.

Each system maintains an ARP cache

The *arp* command can be used to:

display the contents of the ARP cache

add *proxy* entries to it.

arp is available on Linux in /sbin and Solaris systems /usr/sbin

Example ARP exchange:

tcpdump -e -n

```
17:35:26.888798 eth0 < 0:b0:d0:11:91:c4 0:0:0:0:0:1 arp 60: arp who-has
130.127.48.113 tell 130.127.48.243
17:35:26.888817 eth0 > 0:0:0:0:0:0 0:90:27:57:b0:1b arp 42: arp reply 130.127.48.113
(0:90:27:57:b0:1b) is-at 0:90:27:57:b0:1b (0:b0:d0:11:91:c4)
17:35:26.899464 eth0 < 0:20:af:f:6f:cf 0:0:0:0:0:1 arp 60: arp who-has
130.127.48.113 tell 130.127.48.190
17:35:26.899478 eth0 > 0:0:0:0:0:0 0:90:27:57:b0:1b arp 42: arp reply 130.127.48.113
(0:90:27:57:b0:1b) is-at 0:90:27:57:b0:1b (0:20:af:f:6f:cf)
```

ARP command examples:

Sun Solaris:

```
shrads/utills ==> /usr/sbin/arp -a
```

Net to Media Table

Device	IP Address	Mask	Flags	Phys Addr
hme0	flik	255.255.255.255		08:00:69:0e:88:2c
hme0	jmw2	255.255.255.255		00:90:27:57:b0:1b
hme0	citron	255.255.255.255		08:00:20:90:15:e4

OS/2:

```
E:\] ==> arp -a
```

interface	hardware address	IP address	minutes since last use
0	0800202195dd	130.127.48.1	1
0	080020182467	130.127.48.24	7
0	0800207353b3	130.127.48.144	0

Linux:

```
arp -a
```

Address	HWtype	HWaddress	Flags	Mask	Iface
glint3-lane.atm.cs.clem	ether	00:00:77:97:C3:A5	C		lec0
jmw2-lane	ether	00:00:77:88:A0:15	C		lec0
killeen-lane.atm.cs.cle	ether	00:00:77:88:A1:15	C		lec0
waco-lane.atm.cs.clemso	ether	00:00:77:88:A5:A5	C		lec0
130.127.4.1	ether	00:D0:00:8C:9B:FC	C		lec2
glint2-lane.atm.cs.clem	ether	00:20:48:2E:00:EE	C		lec0
jmw9		(incomplete)			eth0
jmw9	*	*			MP lec0
jmw4.cs.clemson.edu	*	*			MP lec0
jmw4-lane.atm.cs.clemso	*	*			MP lec0
192.168.2.36	*	*			MP lec0
jmw6-lane.atm.cs.clemso	*	*			MP lec0
jmw5	*	*			MP lec0

Default is to delete entries 20 minutes old.

Changing an ethernet adapter or an IP address can result in a (temporarily) unreachable host *unless* gratuitous ARP is used or ARP caches are manually flushed.

Proxy ARP

A useful form of ARP used by routers or routing workstations.

A router announces its own MAC address with the IP address of a system "behind" it.

```
/sbin/arp -v -s 130.127.48.184 00:20:af:0f:6f:3c netmask 255.255.255.0 pub
```

Gratuitous ARP

When host 130.127.48.209 comes up it issues:

```
13:35:09.317396 arp who-has 130.127.48.209 tell 130.127.48.209
```

This has two beneficial effects:

- 1 – It allows detection of duplicate IP addresses
- 2 – It flushes any old ARP cache entries associated this IP address with another MAC address