Linux Networking: Address Resolution Protocol

David Morgan

“Hardware address” to “Protocol address” translation

- Network layer and up use one addressing scheme
- Data link and down use (if any) another
- Network-up: “protocol” addresses
- Datalink-down: “hardware” addresses

“Hardware” vs “Protocol” addresses

- Protocol addresses
  - software abstractions
  - apps use them to identify destination computers
  - hardware cannot locate a computer using one
- Hardware addresses
  - applications don’t use them
  - hardware can locate a computer using one
  - but only within same physical net (computers on common medium)

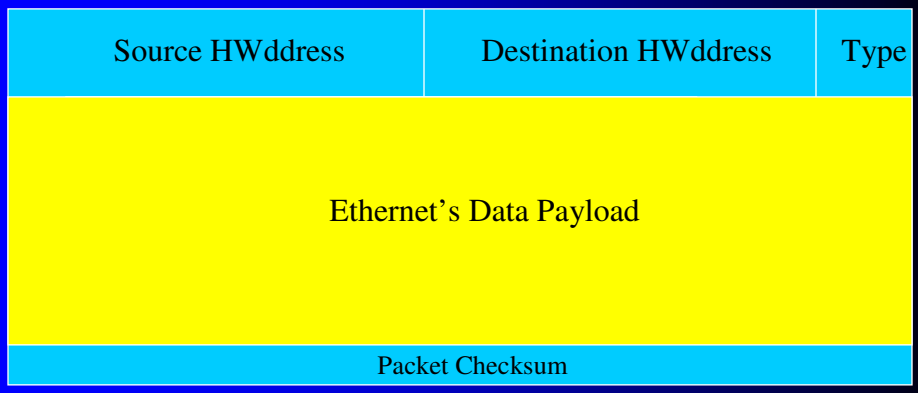
Example

- IP addresses
  - 32-bit numbers
  - telnet/ftp/http use them to identify destination computers
  - ethernet cannot locate a computer using one
- Ethernet addresses
  - 48-bit numbers
  - telnet/ftp/http don’t use them
  - ethernet can locate a computer on the common coax or hub using one

Translation necessary

- Given an IP destination, what is the matching ethernet address?
- Address Resolution Protocol finds out (resolves)

Ethernet frame structure



Ethernet NICs' reading habits

-- frames that NICs read

- Frames with the NIC's own address
- Frames with the address FF:FF:FF:FF:FF:FF
- Others ignored (payload never read)

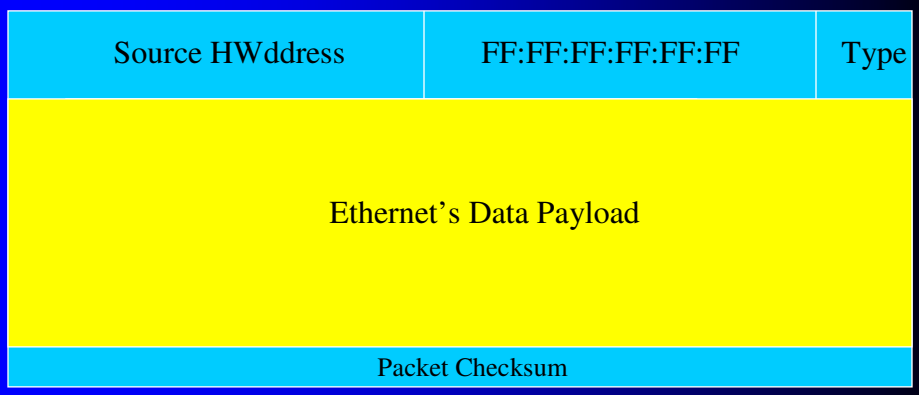
Quick quiz

1. What address gets a frame read by all receiving NICs?
2. What is that address called?

Answers to quiz

1. FF:FF:FF:FF:FF:FF
2. the broadcast address

Ethernet broadcast



How could we translate?

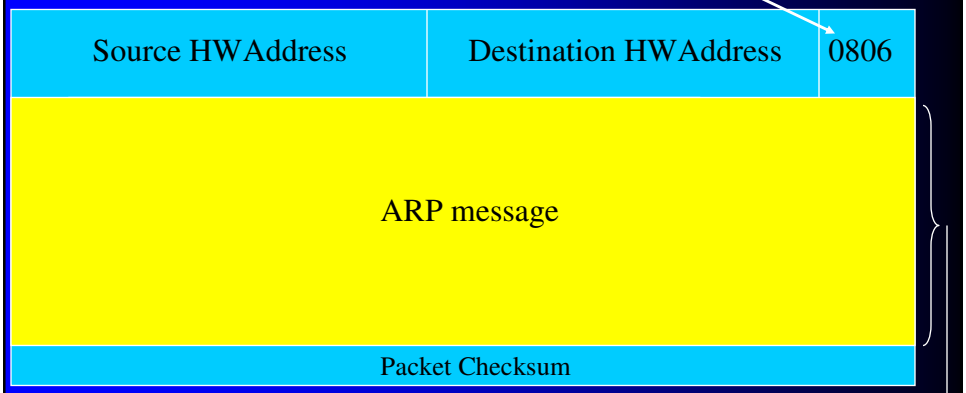
- Table lookup
  - bindings/mappings kept in memory table
- Message exchange
  - dynamic message exchange across network
- ARP uses both

A lookup table

IP address	Ethernet address
192.168.3.1	00:80:C8:E2:AF:61
192.168.3.2	00:A0:CC:D2:F0:42
192.168.3.3	00:40:05:A3:42:26
192.168.3.4	0A:07:4B:12:82:36
192.168.3.5	0A:77:81:0E:52:FA

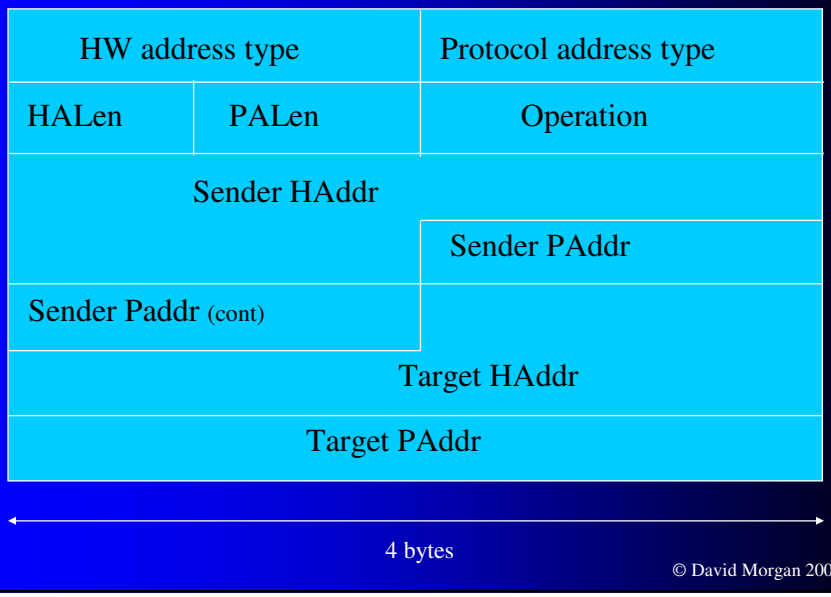
… or how about message exchange?

Ethernet carrying ARP

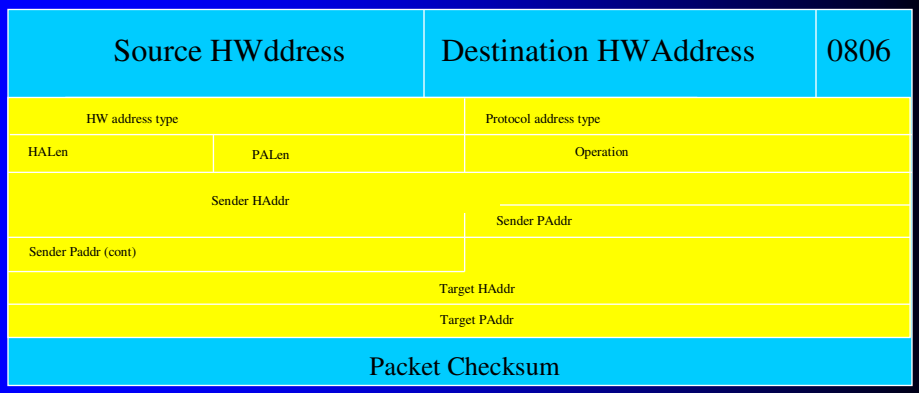


Ethernet's payload may be an Address Resolution Protocol message

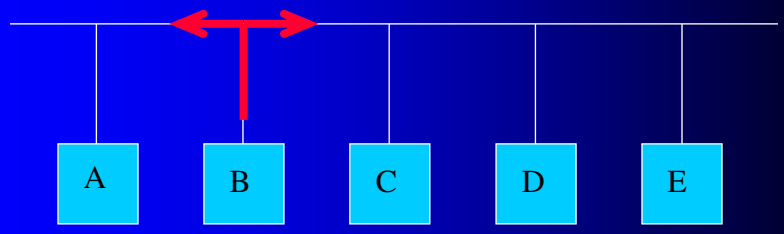
ARP message structure



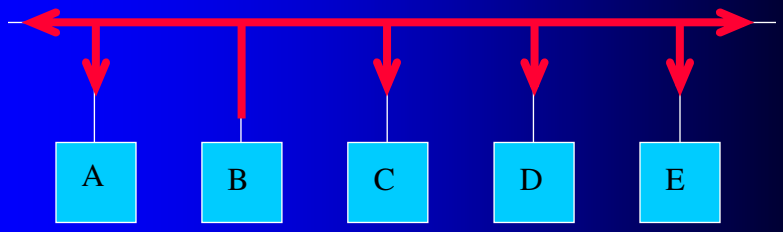
Ethernet carrying ARP



B arps (seeks) D

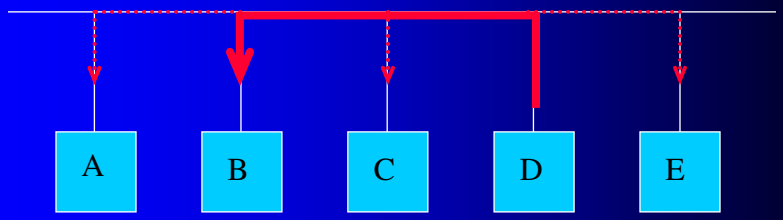


B's arp request is broadcast…



…reaches everybody; everybody reads it, nobody ignores it

D's arp reply is direct to B…

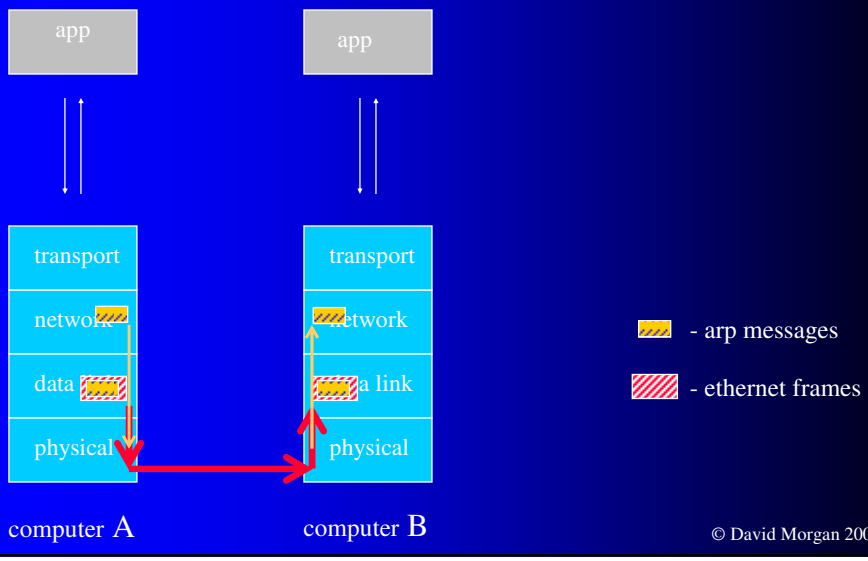


…reaches everybody; B reads it, everybody else ignores it

What does “ignore” mean?
what does “read” mean?
who reads and ignores?

- ethernet and arp – separate software entities
- they operate independently
- B’s ethernet may ignore (discard) or accept (to arp)
- B’s arp may then ignore (discard) or reply

ethernet, arp are independent



Disposition of communiqués

B's request:

	at A	at C	at D	at E
eth	reads	reads	reads	reads
arp	ignores	ignores	reads	ignores

D's reply:

with **hub**

	at A	at B	at C	at E
eth	ignores	reads	ignores	ignores
arp	n/a	reads	n/a	n/a

with **switch**

	at A	at B	at C	at E
eth	n/a	reads	n/a	n/a
arp	n/a	reads	n/a	n/a

Caching arp responses

- arp is inefficient
- takes 3 frames to transfer 1 packet
- packets between host pairs occur in bunches
- so arp caches a table of recent arp'd bindings in memory
- subsequent packets use table, not message exchange

Cached arp table

```
[root@EMACH1 david]# arp -n
Address          HWtype  HWaddress           Flags Mask  Iface
192.168.3.1      ether   00:80:C8:E2:AF:61   C           eth0
192.168.3.3      ether   00:40:05:A3:42:26   C           eth0
64.130.228.62    ether   00:10:E8:09:6E:80   C           eth1
```


Operation essentials: arp request

- target receives, reads broadcast frame
- caches sender's addr binding
- compares target IP with his own
  - quit if no match, otherwise…
- compose arp response
  - reverse sender, target addr bindings
  - insert ethernet addr into Sender Haddr field
  - insert "2" (response) in operation field
  - send

Operation essentials: arp reply

- target receives, reads unicast frame
- caches sender's addr binding
- uses its hardware address to frame and send protocol packet to sender (remember, arp reply "sender" is protocol's intended "recipient")