ARP - Address Resolut
Protocol

TCP/IP class

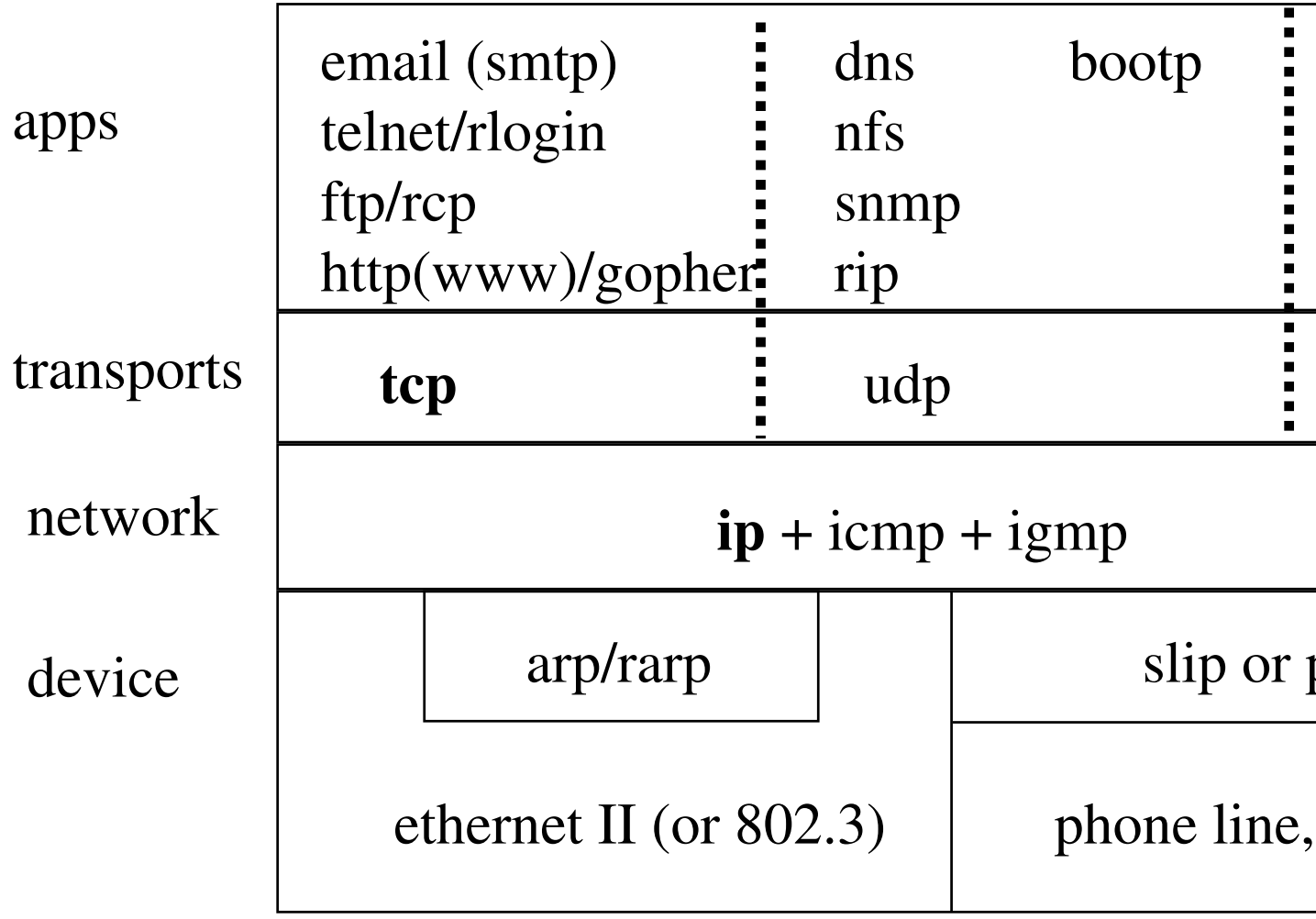
outline

- what's the problem
- how does it work?
- format/arp dump
- variations
  - proxy arp
  - gratuitous arp
- study questions

arp - the problem?

- problem: **how does ip address g mapped to ethernet address?**
- 2 machines on same enet can only con they know MAC/hw addr
- solutions:
  - configure addresses by hand (ouch!)
  - encode in IP address (48 bits in 32?)
  - use broadcast

Internet Protocols



solution - arp protocol

- rfc 826
- host A, wants to resolve IP addr B,
  - send BROADCAST arp request
  - get UNICAST arp reply from B
- same link only
- ethernet (or MAC) specific, although p
  designed to be extensible
- implemented in driver, not IP

% arp -a (SunOs)

 # arp -a
banshee.cs.pdx.edu (131.252.20.128) at 0:0:a7
pdx-gwy.cs.pdx.edu (131.252.20.1) at 0:0:c:
longshot.cs.pdx.edu (131.252.20.129) at 8:0:11
walt-suncs.cs.pdx.edu (131.252.21.2) at 8:0:20
walt-cs.cs.pdx.edu (131.252.20.2) at 8:0:20
connor.cs.pdx.edu (131.252.21.179) at 0:0:c0
dazzler.cs.pdx.edu (131.252.21.132) at 8:0:11
sprite.cs.pdx.edu (131.252.21.133) at 8:0:11

(DNS name,ip address,Ethernet address)

arp command functions

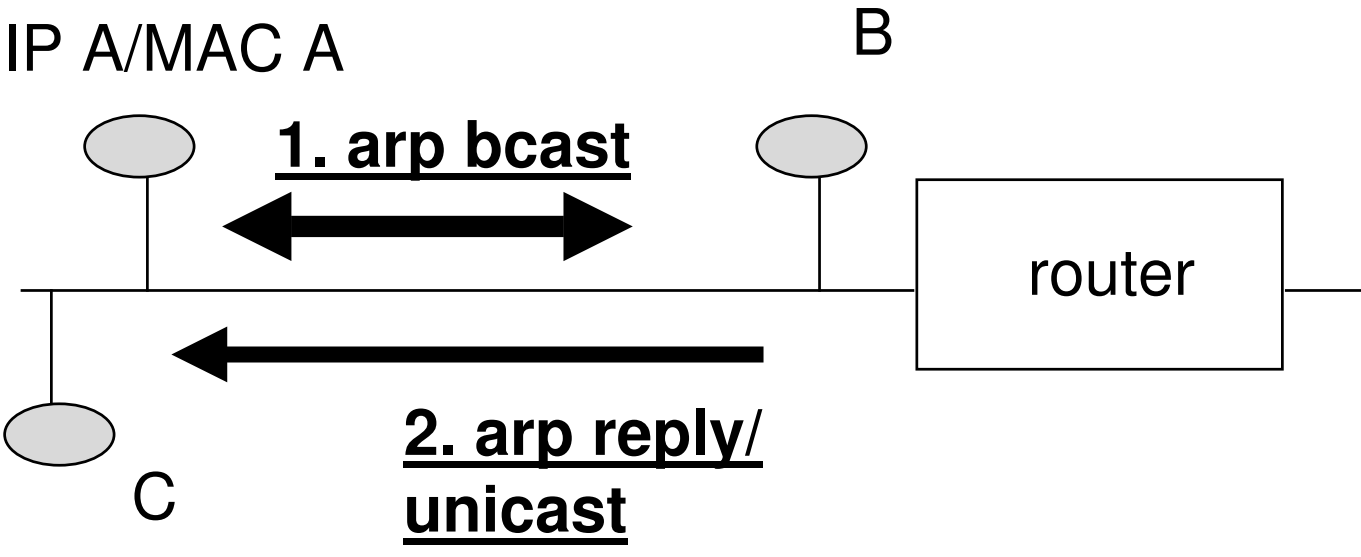
- ping someone and learn MAC ad
- debugging
- delete out of date ARP entry (you
  the IP address, and you don't wan
  OR somebody mucked up)

refinements

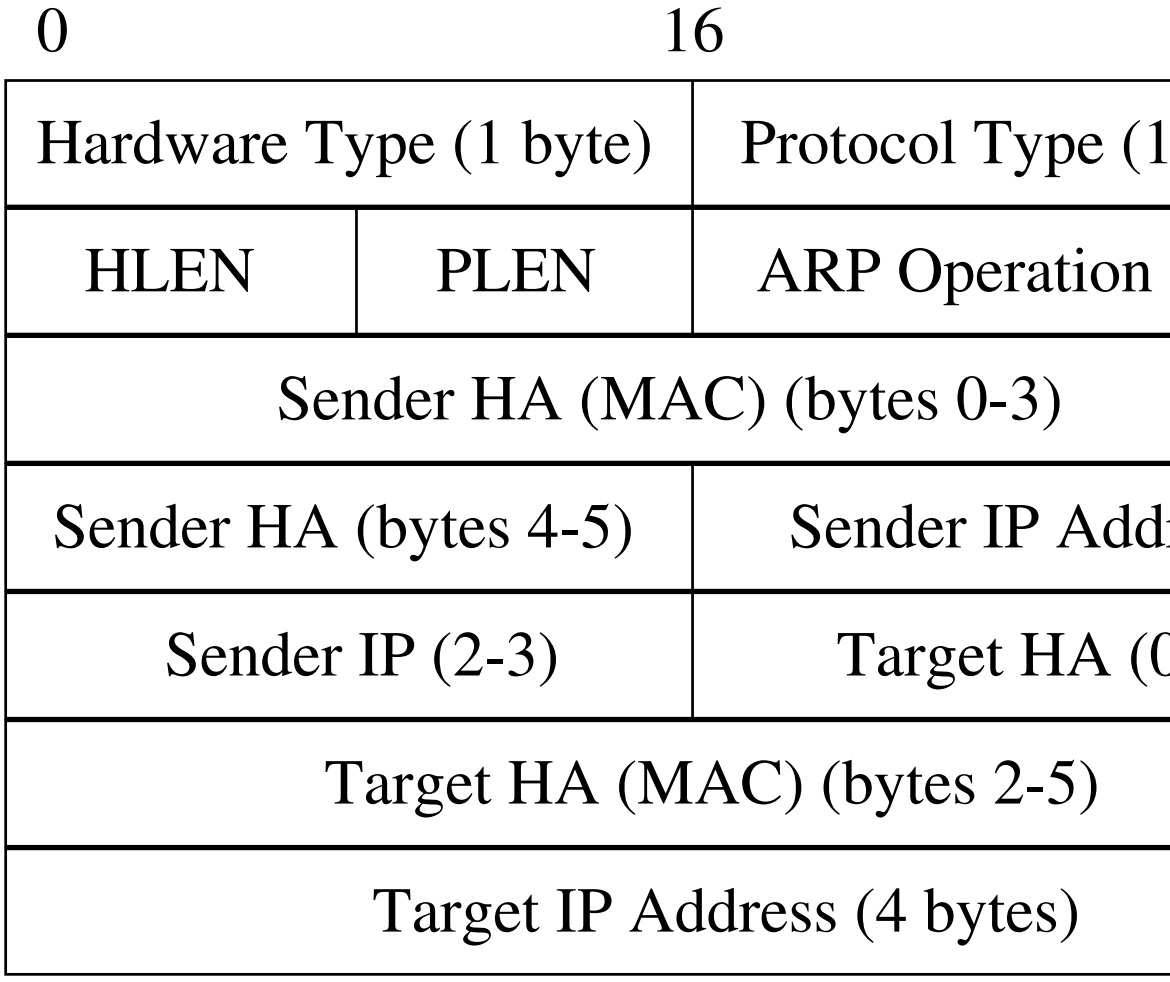
- o.s. will cache arp replies in **arp c**
  (ip , MAC, 20 minute timeou
  - don't need to do arp on every packe
- machine may store all arp broadc sender ip/mac mapping
- recv. machines can update their c
- why not broadcast all packets and :->

arp protocol

1. A to B, arp request/br
2. B to A, arp reply/unica



ARP packet header format



header format details

- header format is not fixed, somew
  dynamic (not used though)
- hw type, ethernet == 1
- protocol type, ip = 0x800
- hwlen, 6 (MAC), plen 4 (ip)
- operation: (used by rarp too)
  - 1: arp request, 2: arp reply
  - 3: rarp request, 4: rarp reply

more details

- sender hw addr, 6 bytes
  - the answer, if reply
- sender ip: 4 bytes
- target hw address: 6 bytes
  - 0 in request
- target ip: 4 bytes

Arp dump - echo request/reply

% *etherfind -x -between bob ray arp*

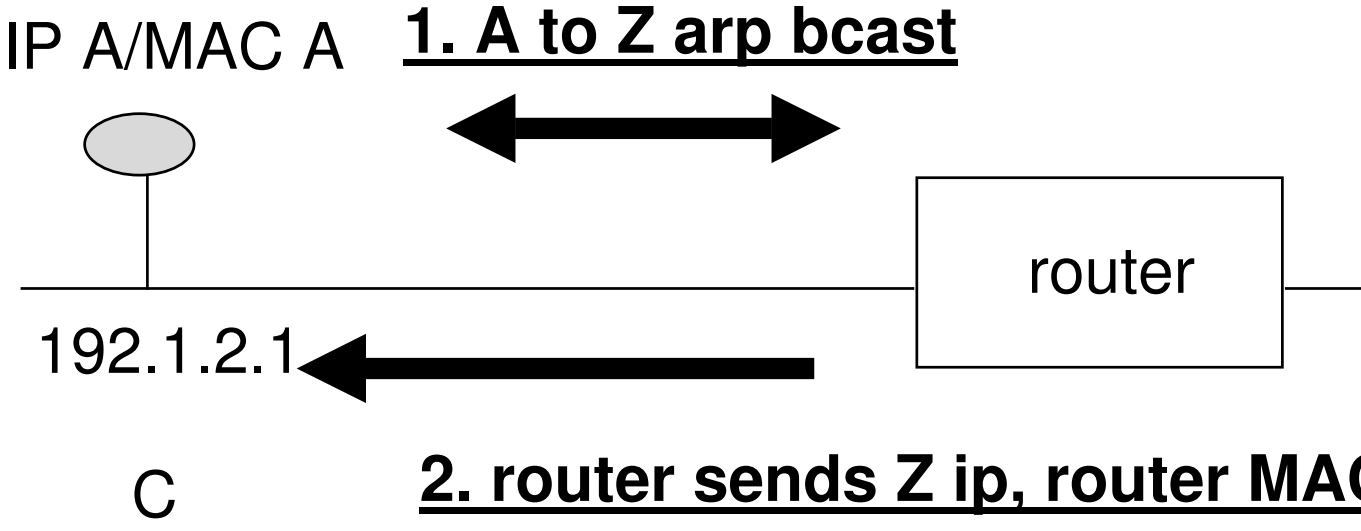
```
   60  arp             bob         ray
ff ff ff ff ff ff 00 80 f9 96 90 00 08 06
00 01 08 00 06 04 00 01 00 80 f9 96 90 00 8
06 57 00 00 00 00 00 00 8f b9 06 01 00 00 0
00 ...

   60  arp         ray             bob
00 80 f9 96 90 00 00 00 3c 00 19 56 08 06
00 01 08 00 06 04 00 02 00 00 3c 00 19 56
06 01 00 80 f9 96 90 00 8f b9 06 57 00 00 0
00 ...
```

proxy arp

- basic idea: machine A answers re
  machine B (that can't arp for som
  forwards packets to B somehow
  - machine A might have 2 IP address
    associated with one interface

proxy arp diagram

- if A to Z, router answers for Z



proxy arp pros/cons

- pros
  - same network numbers
  - can aid dumb host that can’t arp
  - remote serial host appears on same
    courtesy of terminal emulator/route
- cons
  - can drive you nuts -- debugging
  - not simple and not secure

gratuitous/promiscuous arp

- **grat arp** - at boot or change of ip addr
  broadcast arp request for YOURSELF
  - unix ifconfig does this
  - detect other boxes with same IP ad
  - allow recv boxes to cache your MA
- **promiscuous arp** - issue bcast arp to
  other's ideas of ip/mac mapping
  - **problem: no one guaranteed to b**

study questions

- in the picture of the arp protocol exam
  assuming A requests B's address, wha
  1. the router, 2. Z, 3. C play?
- if A was to telnet to Z, what arp packe
  appear on the 2 links (assume ethernet
- is arp a link layer protocol? if so, how
  explain its encapsulation (at the same
  ip...)

even more study questions

- could arp be used to allow an atta
  machine to masquerade as a attac
  machine on a net? think about pr
  too
- if two boxes with different IP net
  numbers are on the same net, car
  for each other?