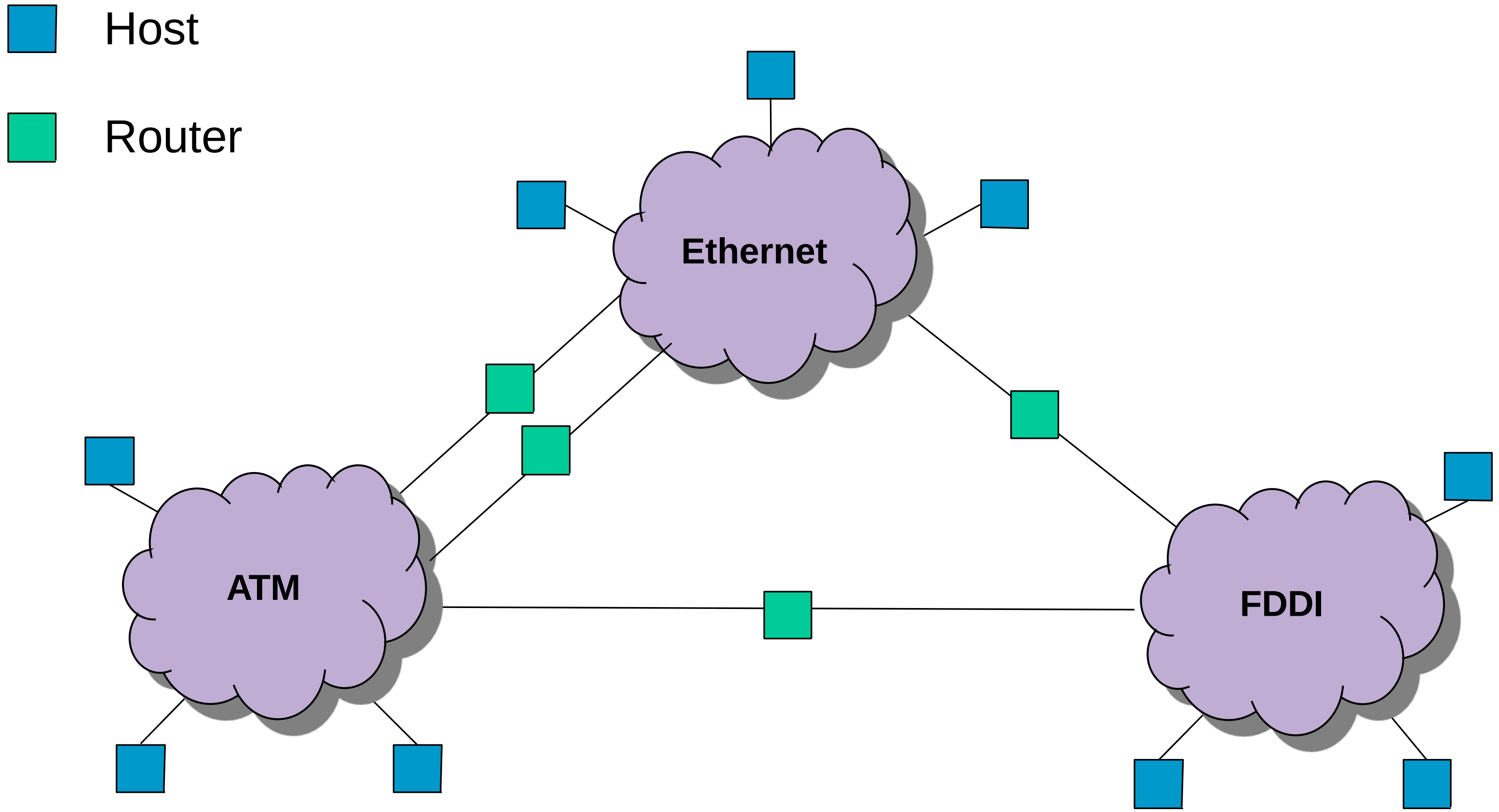
Internetworking
(Pet. and Davie, chapter 4)

Computer Networks
Dr. Jorge A. Cobb

Internetworking

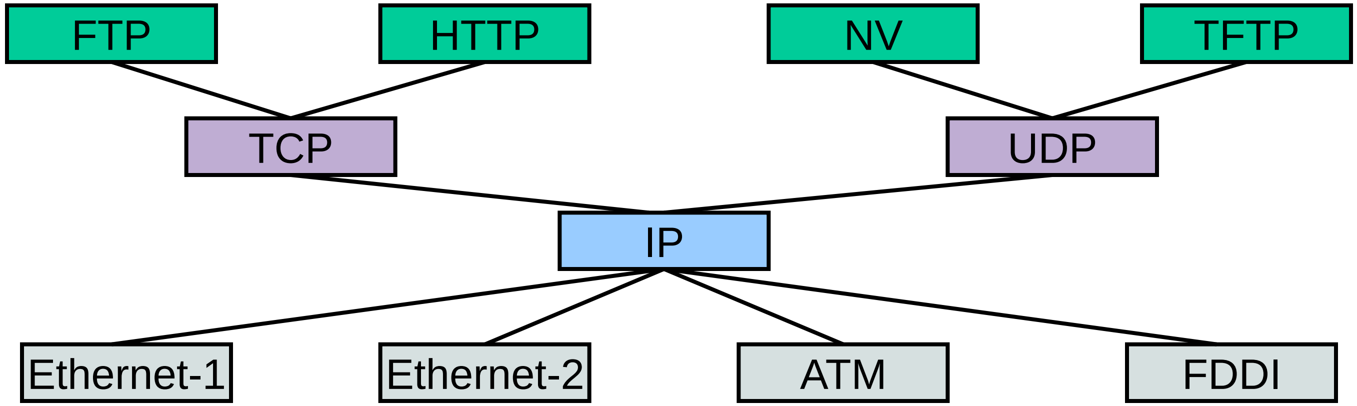


Basics of Internetworking

- **What is an internetwork?**
  - Gives an illusion of a single (direct link) network
  - Built on a set of distributed heterogeneous networks
  - Abstraction typically supported by software
- **Internetwork properties**
  - Supports heterogeneity: Hardware, OS, network type, and topology independent
  - Scales to global connectivity
- **Network (ATM, Ethernet, etc) properties**
  - Must be able to transfer messages between any two nodes in the network.
  - Preferably must support broadcast
- **The Internet is the specific global internetwork that grew out of ARPANET**

Internet Protocol (IP)

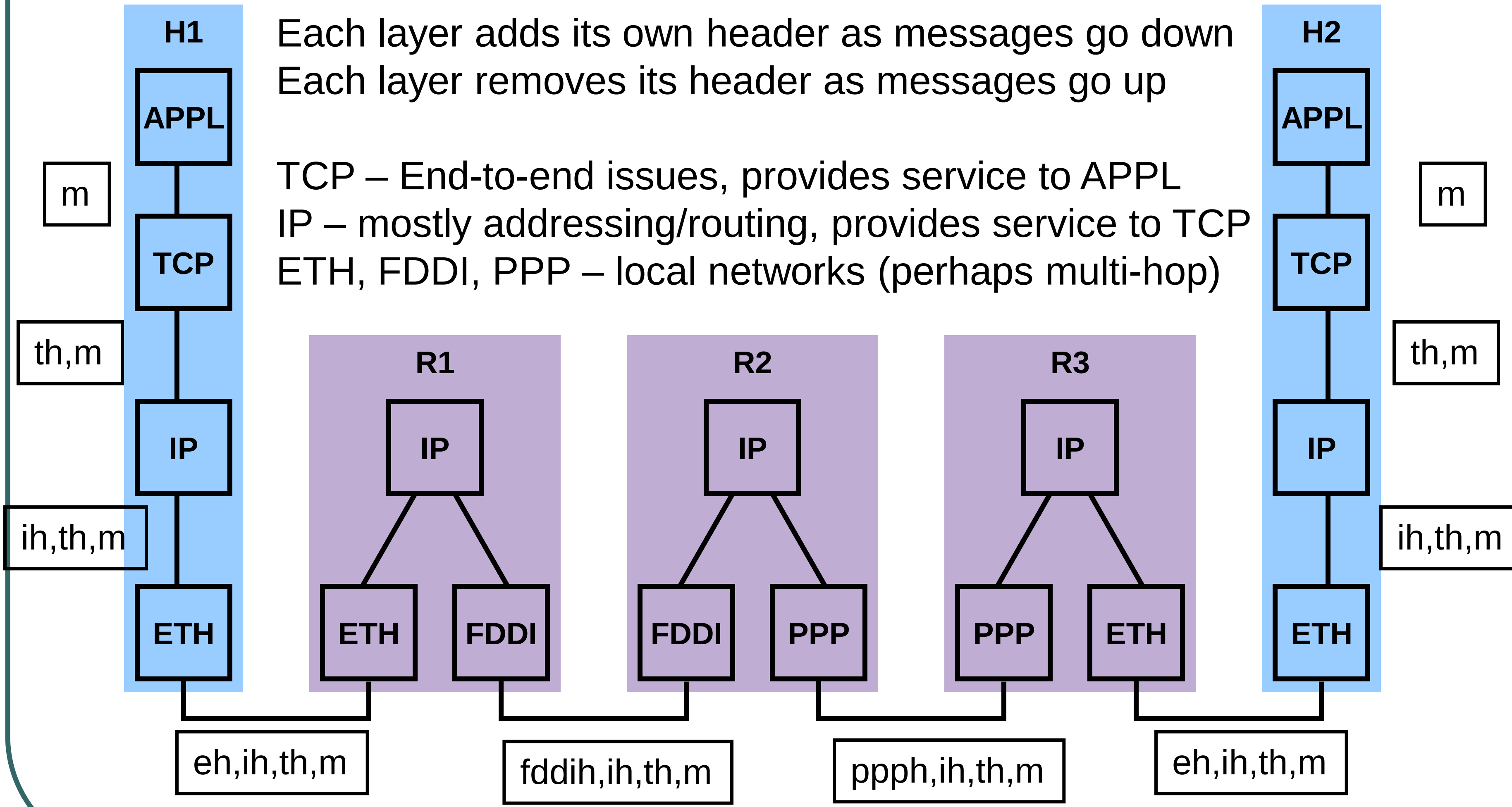
- Network-level protocol for the Internet
- Operates on all hosts and routers
- Protocol stack has an “hourglass” shape



Internetworking with IP

Each layer adds its own header as messages go down
Each layer removes its header as messages go up

TCP – End-to-end issues, provides service to APPL
IP – mostly addressing/routing, provides service to TCP
ETH, FDDI, PPP – local networks (perhaps multi-hop)



Internet Protocol (IP)

- What Services does IP provide?
  - Defines a global name (and address) space
  - Provides service to the transport layer (TCP, UDP)
    - Host-to-host connectivity (connectionless)
    - Best-effort packet delivery
- **Not** in IP service model
  - Delivery guarantees on bandwidth, delay or loss
- Delivery failure modes
  - Packet delayed for a very long time
  - Packet loss
  - Packet delivered more than once
  - Packets delivered out of order

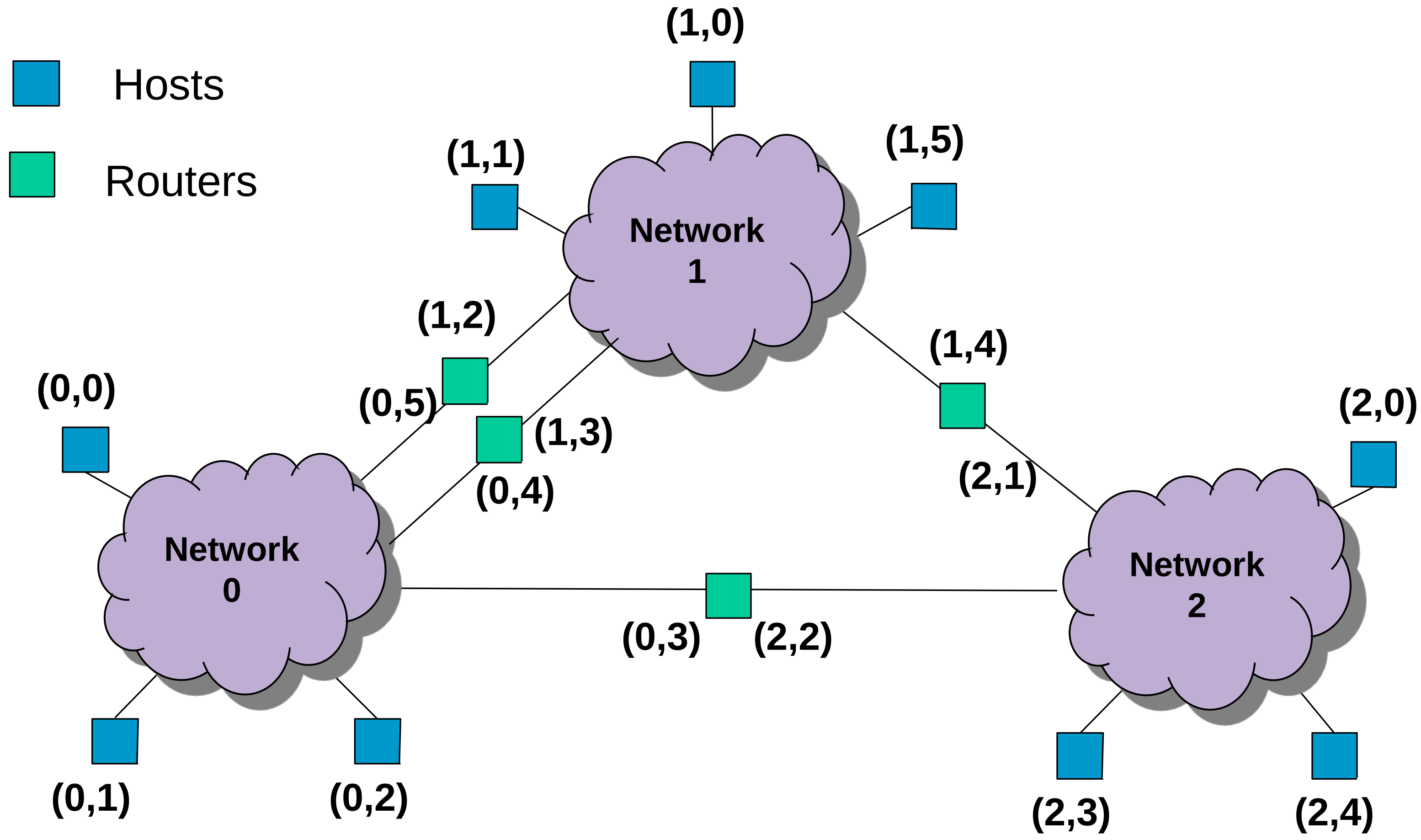
Simple Internetworking with IPv4

- Host addressing
- Forwarding
- Fragmentation and reassembly
- Error reporting/control messages

IPv4 Address Model (the first try …)

- Properties
  - 32-bit address
  - Hierarchical
    - (Network, host) hierarchy
    - Each network has a unique id in the globe
    - Each host within each network has a unique id within the network.
  - Maps to logically unique network adaptor
    - Hosts have (typically) one IP address
    - Routers have multiple IP addresses, one per attached network

Internetworking



Three Classes of Networks (and IP addresses within them)

Class A:

0	**Network (7 bits)**	**Host (24 bits)**

Class B:

1	**0**	**Network (14 bits)**	**Host (16 bits)**

Class C:

1	**1**	**0**	**Network (21 bits)**	**Host (8 bits)**

IPv4 Address Model

Class	Network ID	Host ID	# of Addresses	# of Networks
A	"0" + 7 bits	24 bit	2^{24} **- 2**	128
B	"10" + 14 bits	16 bit	65,536 **- 2**	2^{14}
C	"110" + 21 bits	8 bit	256 **- 2**	2^{21}
D	"1110" + Multicast Address		IP Multicast	
E	Future Use			

IPv4 Address Model

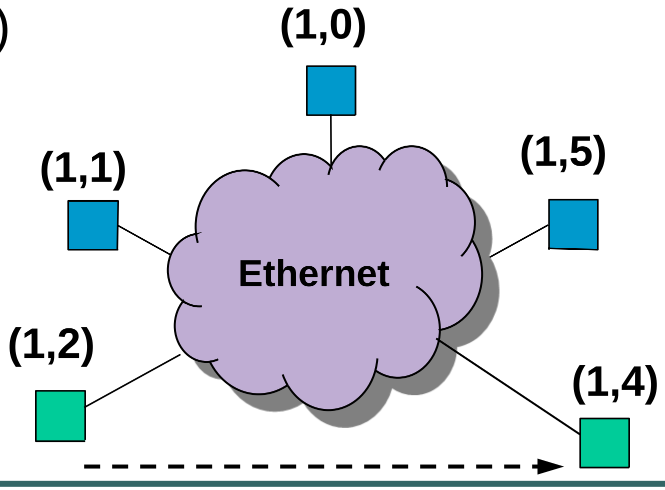
- IP addresses
  - Usually represented using decimal-dot notation (instead of hexadecimal, don't ask me why)
    - Host in class A network
      - 56.0.78.100 www.usps.gov
    - Host in class B network
      - 128.174.252.1 www.cs.uiuc.edu
    - Host in class C network
      - 198.182.196.56 www.linux.org
- Internet domain names
  - ASCII strings separated by periods
  - Provides some administrative hierarchy
    - host.subdomain.domain.domain_type (com, edu, gov, org, …)
    - host.domain.country (us, de, jp, …)

IPv4 Translation support

- Internet domain name to IP address (and vice-versa)
  - Assume your application knows the domain name of the destination
  - Your application must obtain the IP address of the destination before it can send data to it.
  - You call upon the Domain Name Service (DNS)
    - A hierarchy of servers.
    - Give your DNS server a domain name, and it returns to you the IP address (and vice-versa).
- What about physical addresses?

Translation to Physical Addresses

- Problem
  - An IP route can pass through many physical networks
    - E.g., Ethernet (recall that IP is just software)
  - Physical network source and destination addresses are needed at each hop along the route
- Router (1,2) wants to send an IP message to (1,4)
  - It needs the Ethernet (i.e., physical) address of (1,4) (how to get it?)

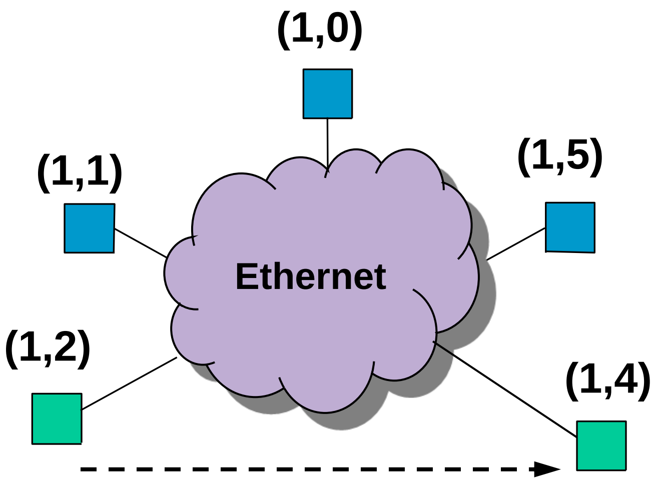


Solution – Address Translation

- Solution
  - Translate from IP address to physical address
  - This is done via the Address Resolution Protocol (ARP)
    - IP asks ARP to translate the IP address into a physical address
    - This is done as needed at each hop.
- Example …

Example

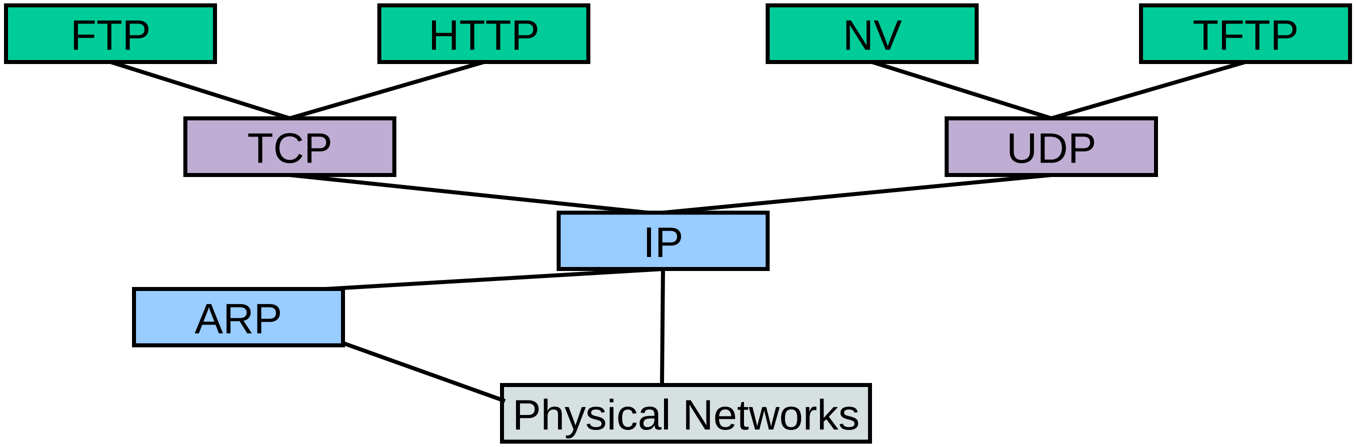
- Router (1,2) wants to send an IP message to (1,4)
  - It first finds out the Ethernet (i.e., physical) address of (1,4) (via ARP, more on ARP later)
- Router (1,2) gives the message and the Ethernet destination address to the Ethernet software (driver).
- The Ethernet software encapsulates the IP message (turns it into an Ethernet message) and sends it over the Ethernet hardware.
- The Ethernet hardware at the other router receives the message.
- The Ethernet software decapsulates the message, and gives it to the IP software.



IP to Physical Address Translation

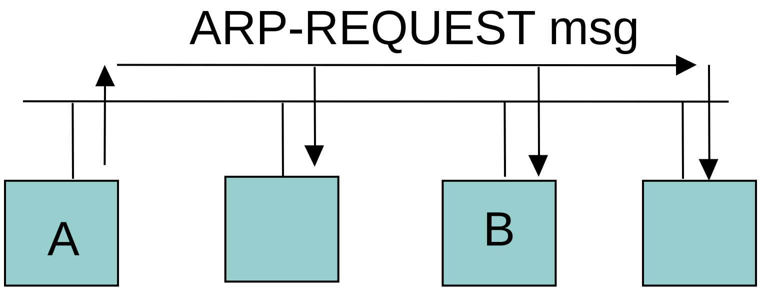
- Hard-coded: Encode physical address in IP address
    - E.g, map Ethernet addresses to IP addresses
    - Makes it impossible to associate IP address with topology (routing becomes too difficult)
- Centralized
  - Maintain a central repository and distribute to hosts
    - Bottleneck for queries and updates
- Automatically generated table
  - Use ARP to build table at each host as needed.
    - Take advantage of broadcast.
  - Use timeouts to clean up table (remove old unused entries)

ARP in the Protocol "Stack"



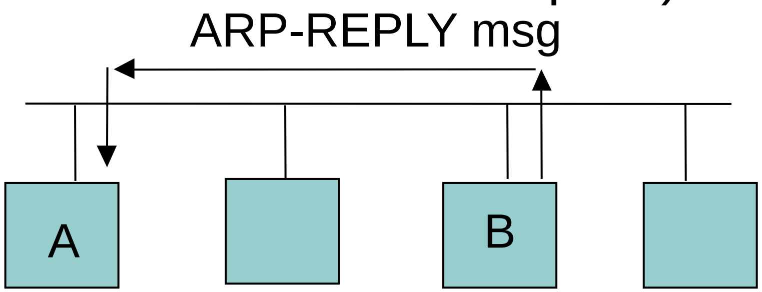
ARP Request

- A wants to send an IP message to B
- A *broadcasts* an ARP-REQUEST message on the physical network
  - What are the physical network source and destination addresses in the physical network header?
- ARP-REQUEST header contains
  - sender IP address field = A's IP address
  - sender phys. net. address field = A's phys. net. address
  - target IP address field = B's IP address
  - target phys. net. address field = empty (unknown)



ARP Reply

- **All machines** in the physical network receive the ARP request
  - Only B responds, because the target IP address is B's
- B responds with an ARP-REPLY *directly* to A.
  - (again, what are the src and dst addresses in the physical network header?)
- ARP-REPLY header contains:
  - sender IP address field = IP address of B
  - sender phys. net. address field = phys. net. address of B
  - target IP address field = IP address of A
  - target phys. net. address field = phys. net. address of A (B learned this from the ARP request)



ARP caching

- A learned the phys. net. address of B (from the reply)
- B *also* learned the phys. net. address of A (from the request)
- *All* machines on the network could learn A’s phys.net. address (since A’s request was a broadcast) <u>but in general this is not cached.</u>
- When a machine learns a new phys. net. address, it keeps it in a cache for future use, to avoid unnecessary ARP messages.
- What if no reply from B? A retransmits the request with exponentially increasing intervals (first wait 1 second, then 2 seconds, then 4, etc...)

Datagram Forwarding (routing) within IP

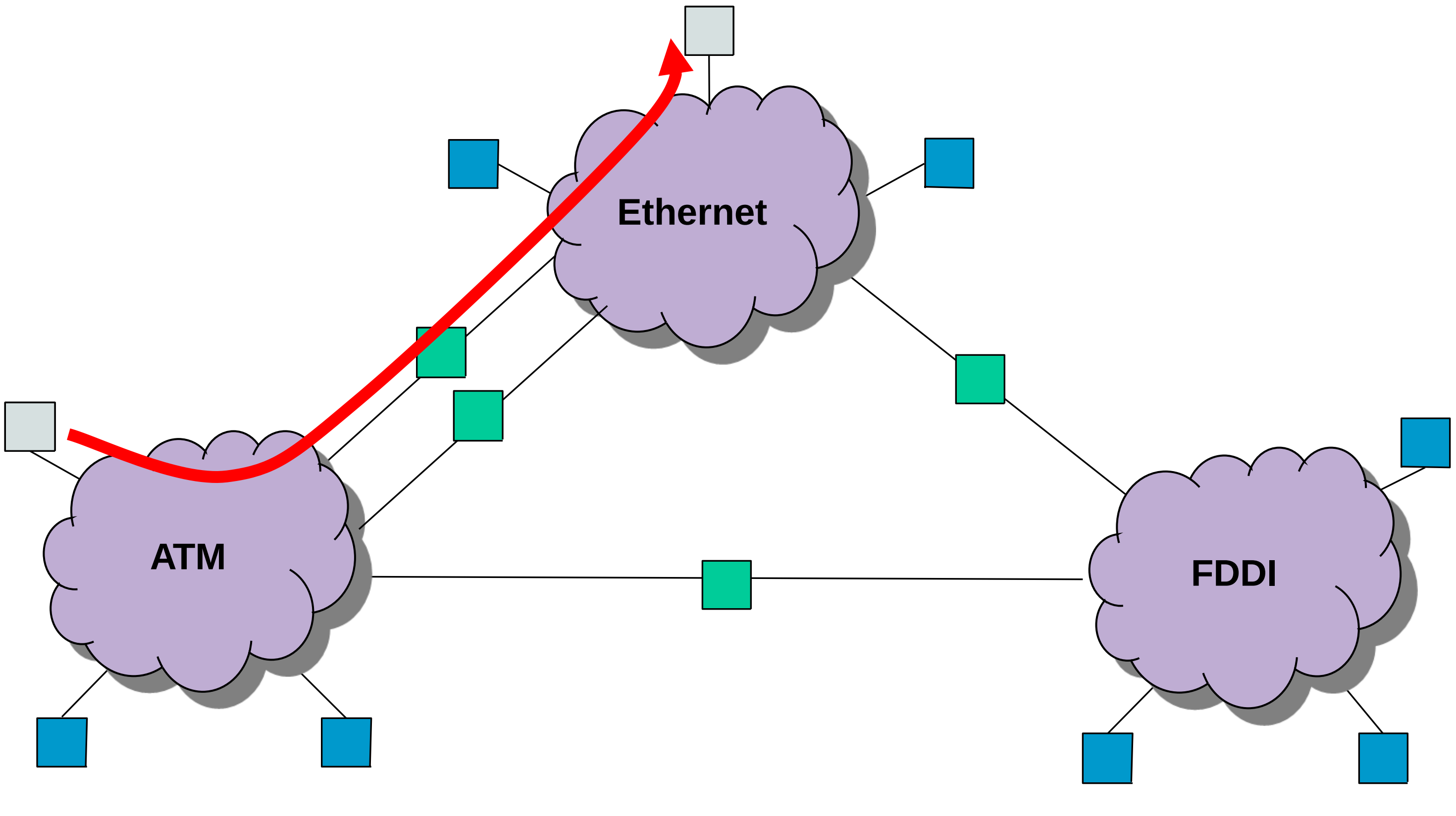
- What to do with a packet (generated or received?)
  - Send directly to destination if the destination is on the same network (how do you know?)
  - Send indirectly (via another router) to destination if it is located on a different network (i.e., use your routing table)

- Use ARP in both cases above to get physical address of next host/router

Datagram Forwarding cont …

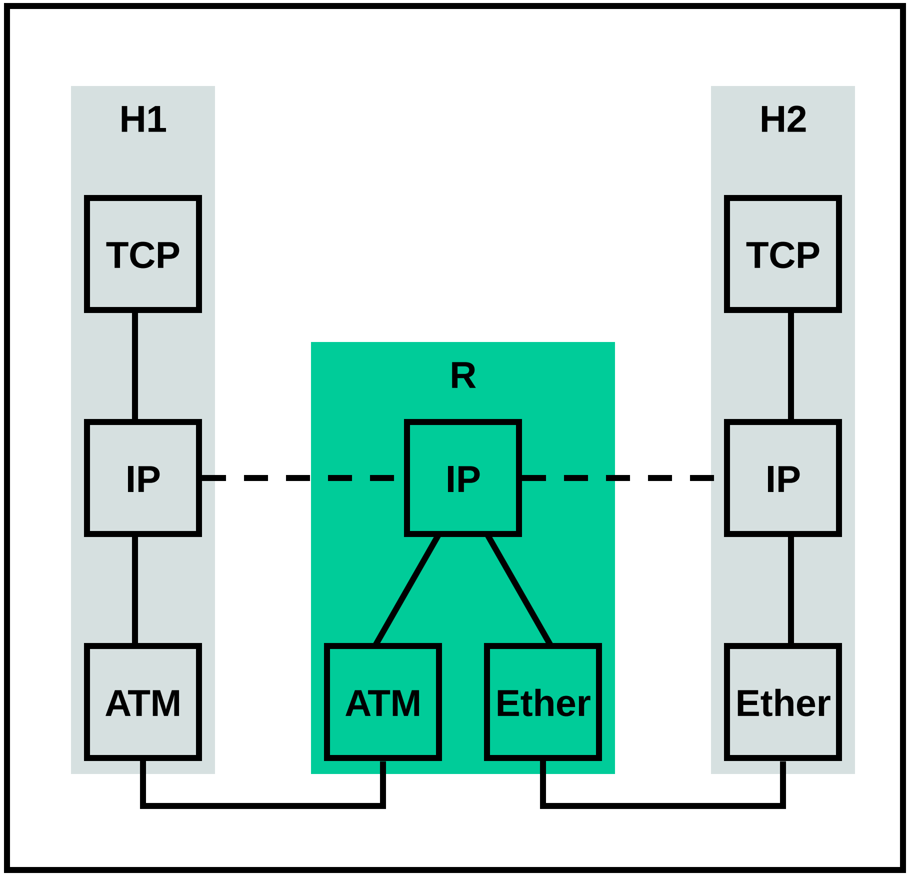
- Hosts and routers maintain routing tables (a.k.a. forwarding tables):
  - Contains list of <network #, next-hop> pairs
    - Match the network # of IP's destination address in the IP datagram against the network # in the list of pairs
    - Then forward packet to the corresponding next-hop
- Routing table often contains a default route
    - Pass unknown destination up the hierarchy to the next level
- Table is:
  - simple and static on hosts
  - complex and dynamic on routers

IP Message Transmission

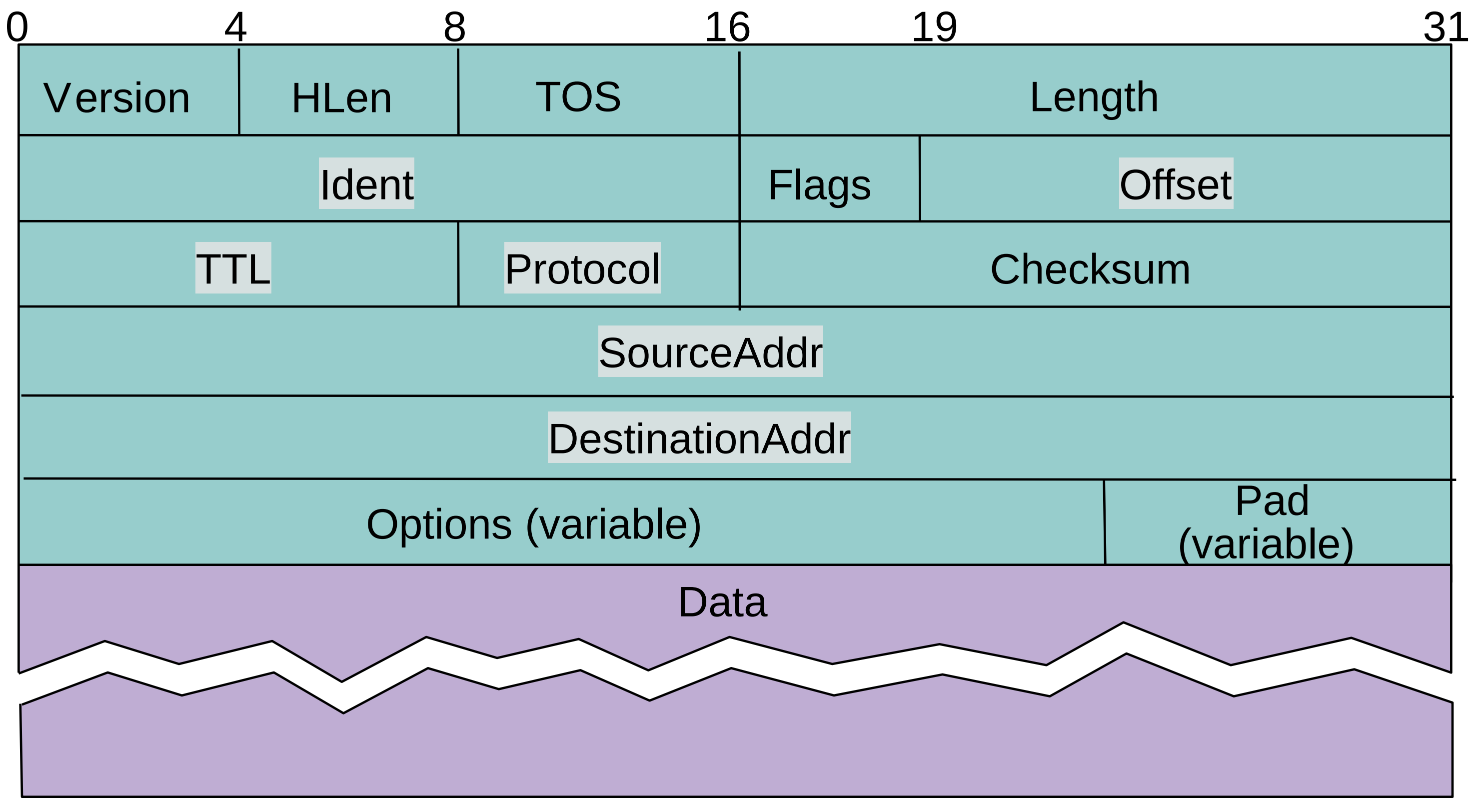


IP Message Transmission

- H1-TCP:
  - creates a packet
  - destination = IP addr of H2
- H1-IP:
  - dest is on a different network
  - so, IP forwards packet to default router R
  - IP looks up R's ATM address and sends packet via ATM
- R-IP:
  - R notices destination addr is on the same network as R.
  - IP looks up destination (H2) Ethernet address and sends packet via Ethernet



IP Packet Format



IP Packet Format

- 4-bit version
  - IPv4 = 4, IPv6 = 6
- 4-bit header length
  - Counted in words, minimum of 5
- 8-bit type of service field (TOS)
  - Mostly unused
- 16-bit data length
  - Counted in bytes

IP Packet Format

- Fragmentation support
  - 16-bit packet (datagram) ID
    - Increases by 1 for every packet sent by the host
    - If packet is later fragmented, all fragments from the same packet have the same ID
      - Each fragment is an IP packet in its own right
  - 3-bit flags in header
    - 1-bit is used to mark last fragment
  - 13-bit fragment offset into packet
    - Counted in words
- 8-bit time-to-live field (TTL)
  - Hop count decremented at each router
  - Packet is discard if TTL = 0

IP Packet Format

- 8-bit protocol field (who do you give the packet to?)
  - TCP = 6, UDP = 17
- 16-bit IP checksum on header (not the data!)
- 32-bit source IP address
- 32-bit destination IP address
- Options (variable sized)
  - Source-based routing
  - Record route (limited)
- Padding
  - Fill to 32-bit boundaries

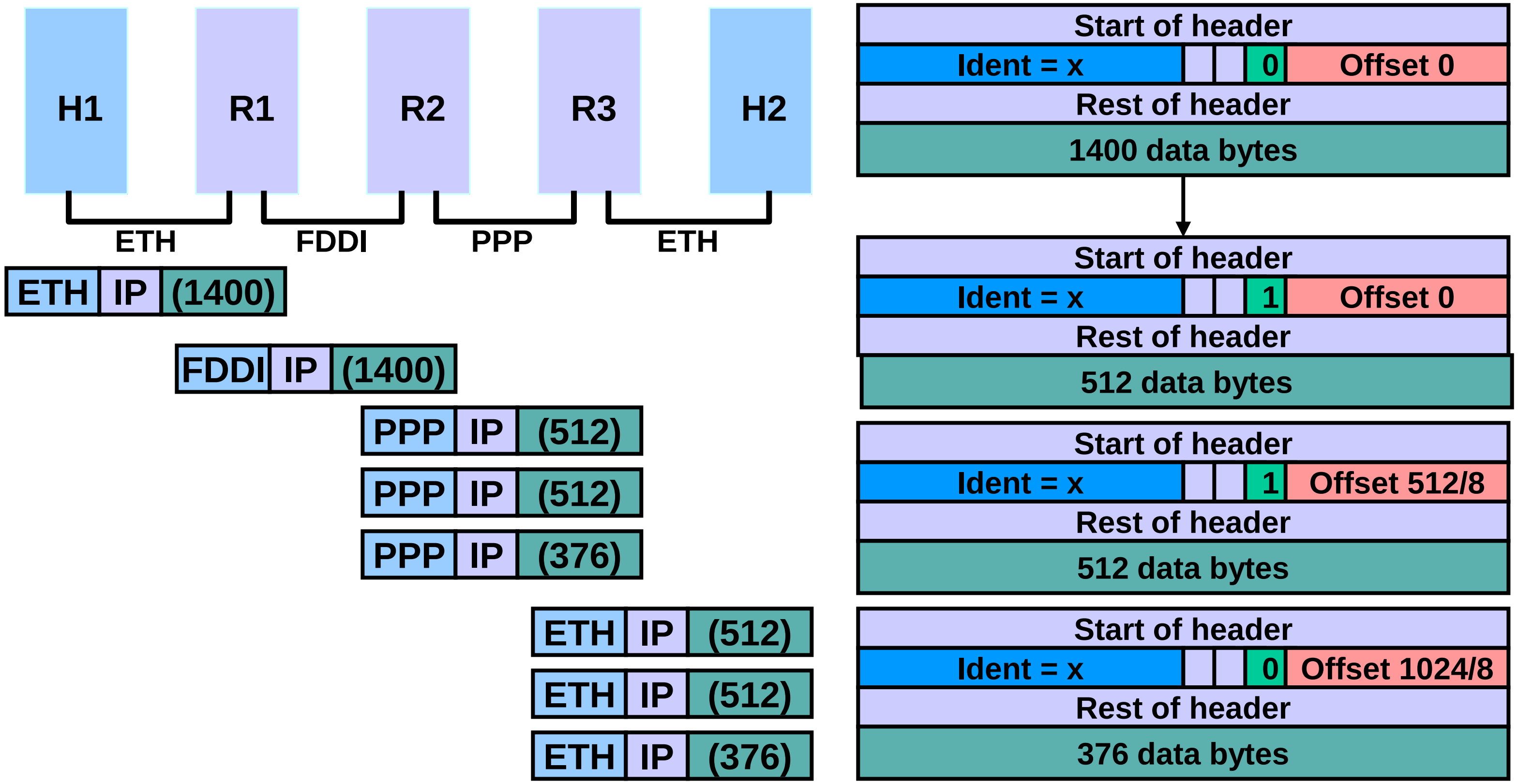
IP Fragmentation and Reassembly

- Problem
  - Different physical layers provide different limits on frame (packet) length
    - Known as the maximum transmission unit (MTU)
  - Source host does not know minimum value of MTU along the path (especially along dynamic routes)
- Solution
  - When necessary, split IP packet into acceptably sized packets prior to sending over physical link
  - Questions
    - Where should reassembly occur?
    - What happens when a fragment is damaged/lost?

IP Fragmentation and Reassembly

- Fragments are self-contained IP packets
- Reassemble at destination to minimize re-fragmentation
- Drop all fragments of a packet at the destination if one or more fragments are lost
- Avoid fragmentation at source host
  - Transport layer should send packets small enough to fit into one MTU of local physical network
  - Must consider IP header

IP Fragmentation and Reassembly



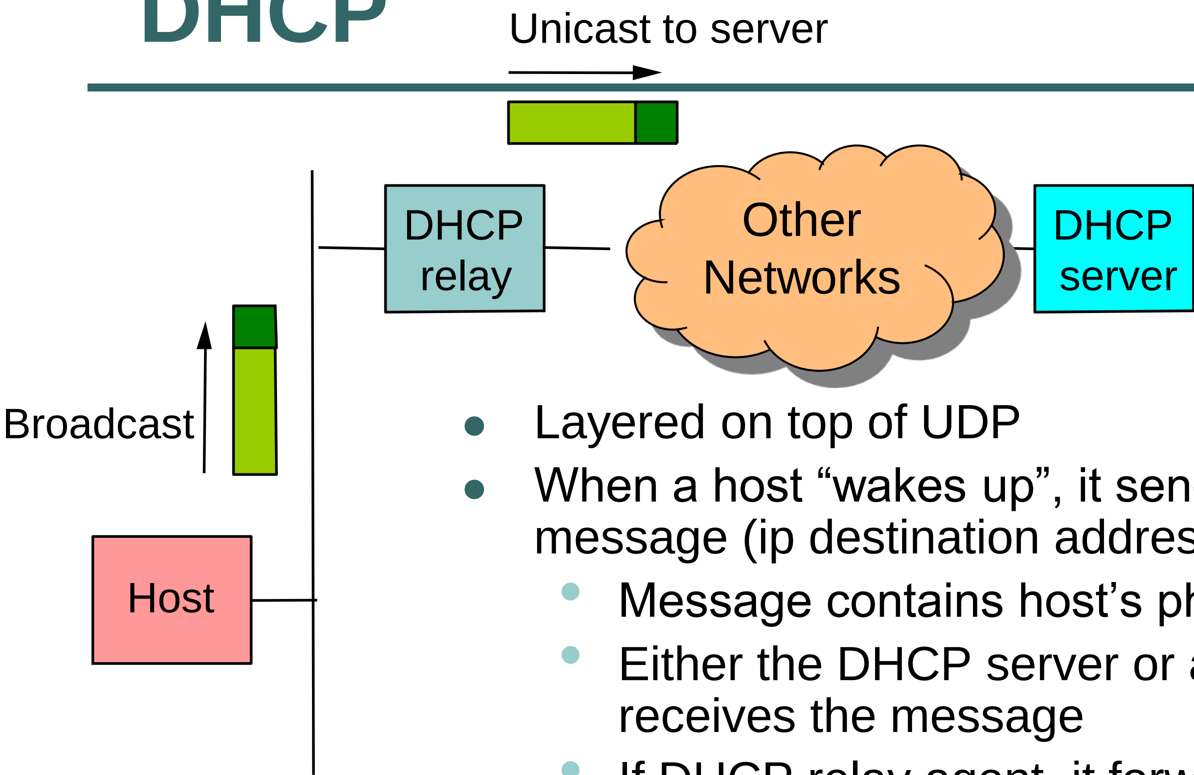
Host Configuration

- What configuration information does a host need?
  - Its IP address (leased for certain duration)
  - Subnet mask (later)
  - Default router address
  - DNS Server

Dynamic Host Configuration Protocol (DHCP)

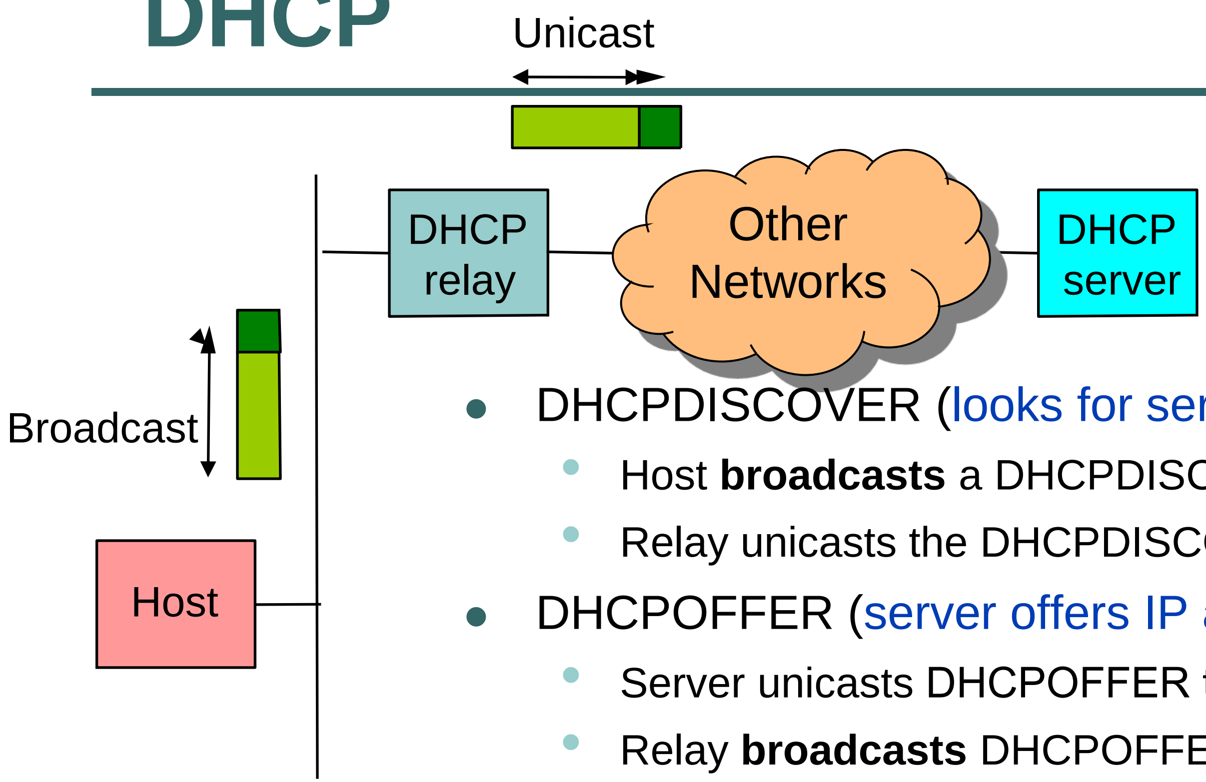
- Want: allow the automatic management and sharing of IP addresses
  - Servers manage a finite number of IP addresses
  - Addresses are leased to clients for finite leases
  - Renew lease if you need IP address longer
- Implementation by the DHCP protocol
  - It is a simple way to automate configuration information
  - Network administrator does not need to enter host IP address by hand
  - Good for large and/or dynamic networks

DHCP



- Layered on top of UDP
- When a host “wakes up”, it sends a DHCPDISCOVER message (ip destination address 255.255.255.255)
  - Message contains host’s physical address
  - Either the DHCP server or a DHCP relay agent receives the message
  - If DHCP relay agent, it forwards the message to the DHCP server
  - Server replies with an IP number and configuration information
- What happens if a host turns itself off without releasing the address?

DHCP



- DHCPDISCOVER (looks for server)
  - Host **broadcasts** a DHCPDISCOVER
  - Relay unicasts the DHCPDISCOVER to server
- DHCPOFFER (server offers IP address)
  - Server unicasts DHCPOFFER to relay's IP address
  - Relay **broadcasts** DHCPOFFER to host
- DHCPREQUEST (hosts requests offered IP address)
  - Host **broadcasts** DHCPREQUEST
  - Relay unicasts DHCPREQUEST to server
- DHCPACK (server acks, i.e. grants, IP address)
  - Server unicasts DHCPACK to relay's IP address
  - Relay **broadcasts** DHCPACK to host

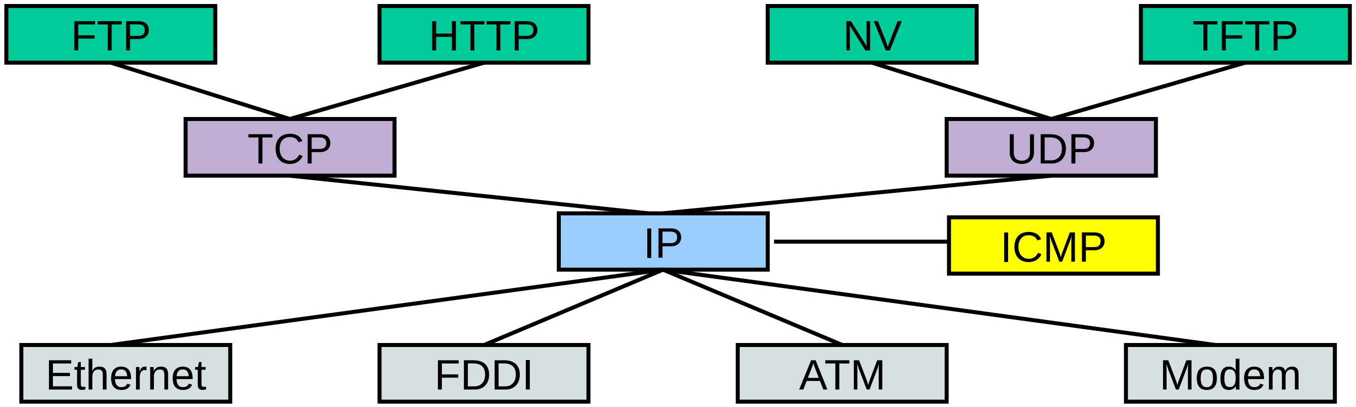
From Microsoft's web site

Source MAC addr	Dest MAC addr	Source IP addr	Dest IP addr	Packet Description
Client	Broadcast	0.0.0.0	255.255.255.255	DHCP Discover
DHCPsrvr	Broadcast	DHCPsrvr	255.255.255.255	DHCP Offer
Client	Broadcast	0.0.0.0	255.255.255.255	DHCP Request
DHCPsrvr	Broadcast	DHCPsrvr	255.255.255.255	DHCP ACK

Ethernet broadcasts are used to let other servers (if any) know of the exchange occurring between the host and the server.

Internet Control Message Protocol (ICMP)

- IP companion protocol
  - Handles error and control messages



ICMP

- Error Messages
  - Host unreachable
  - Reassembly failed
  - IP checksum failed
  - TTL exceeded (packet dropped)
  - Invalid header
- Control Messages
  - Echo/ping request and reply
  - Echo/ping request and reply with timestamps
  - Route redirect