


Simple Internetworking



Hui Chen, Ph.D.

Department of Engineering & Computer Science
Virginia State University
Petersburg, VA 23806

Acknowledgements

- ❑ Some pictures used in this presentation were obtained from the Internet
- ❑ The instructor used the following references
 - Larry L. Peterson and Bruce S. Davie, Computer Networks: A Systems Approach, 5th Edition, Elsevier, 2011
 - Andrew S. Tanenbaum, Computer Networks, 5th Edition, Prentice-Hall, 2010
 - James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach, 5th Ed., Addison Wesley, 2009
 - Larry L. Peterson's (<http://www.cs.princeton.edu/~llp/>) Computer Networks class web site

Outline

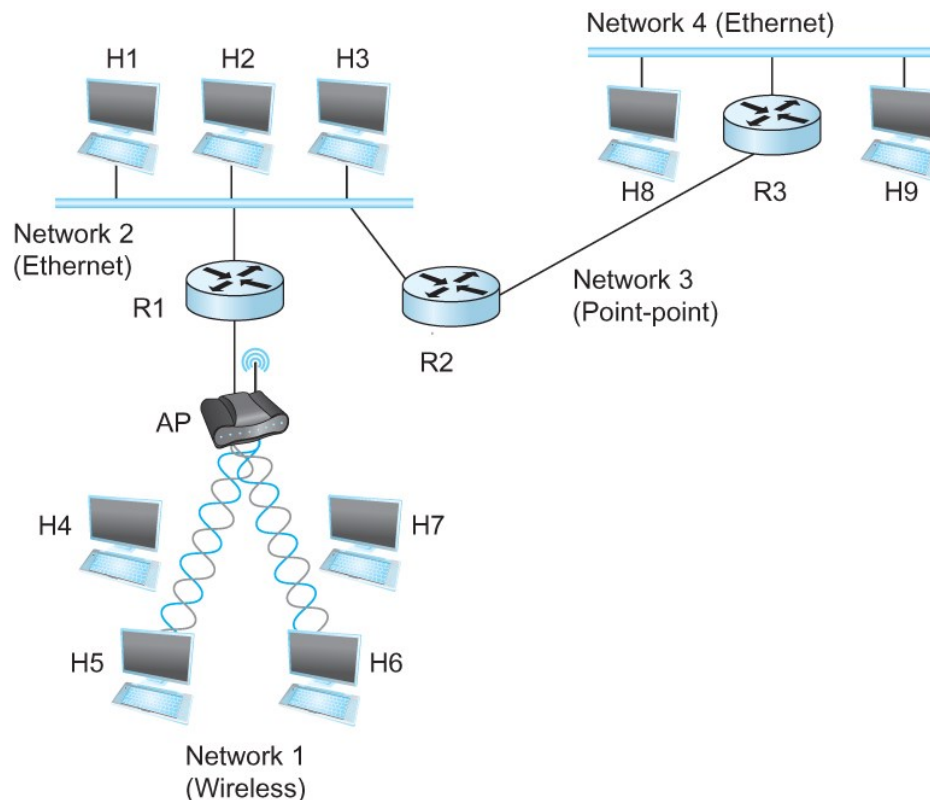
- ❑ Topic: internetworking
 - Case study: Internet Protocol (IP) Suite
- ❑ Simple interworking
 - **i**nternet and the **I**nternet
 - Global addressing scheme
 - Packet fragmentation and assembly
 - Best effort service model and datagram forwarding
 - Address translation
 - Host configuration
 - Error reporting

Heterogeneity and Scalability

- ❑ LAN: small in size
- ❑ How to extend LAN?
 - Bridges and switches
 - Good for global networks?
 - ❑ Spanning tree algorithms → very long path and huge forwarding tables
 - ❑ Bridges and switches: link level/layer 2 devices → networks must be using the same type of links
- ❑ Problems to deal with
 - Scalability: global networks are huge in size
 - Heterogeneity: networks of different types of links are in use

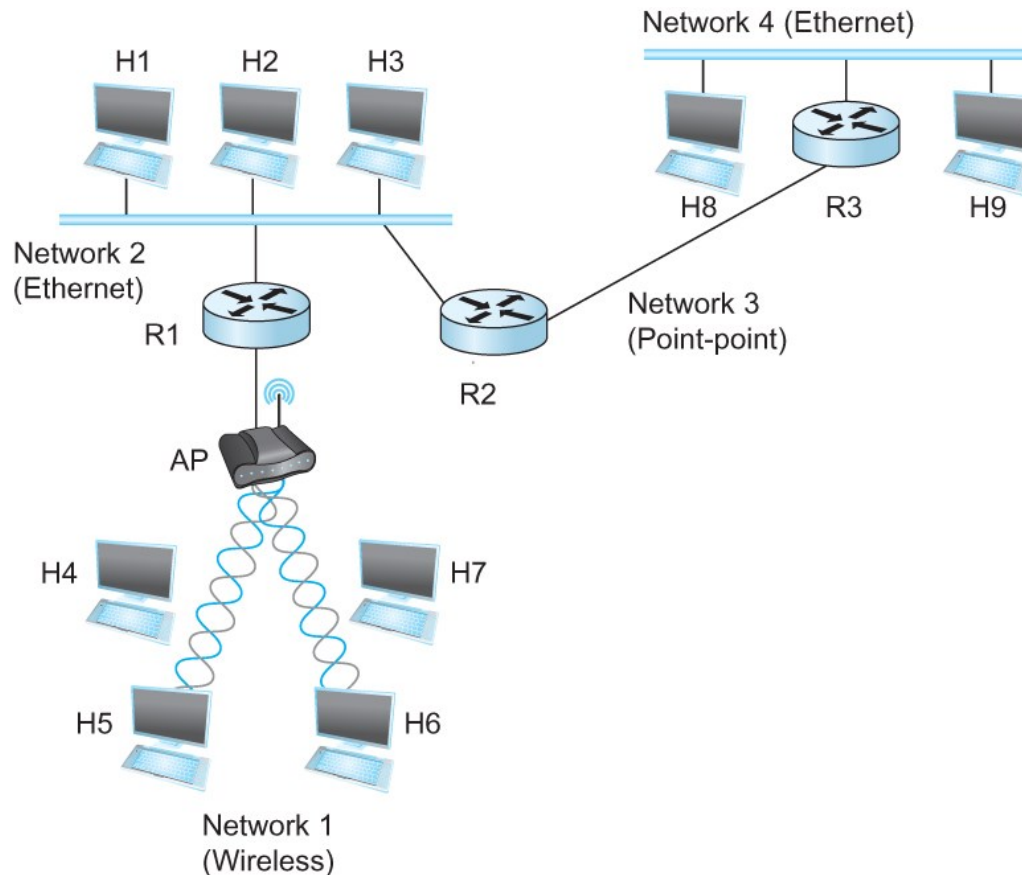
Network of Networks: internetworking

- ❑ An arbitrary collection of networks interconnected to provide some sort of host-host to packet delivery service



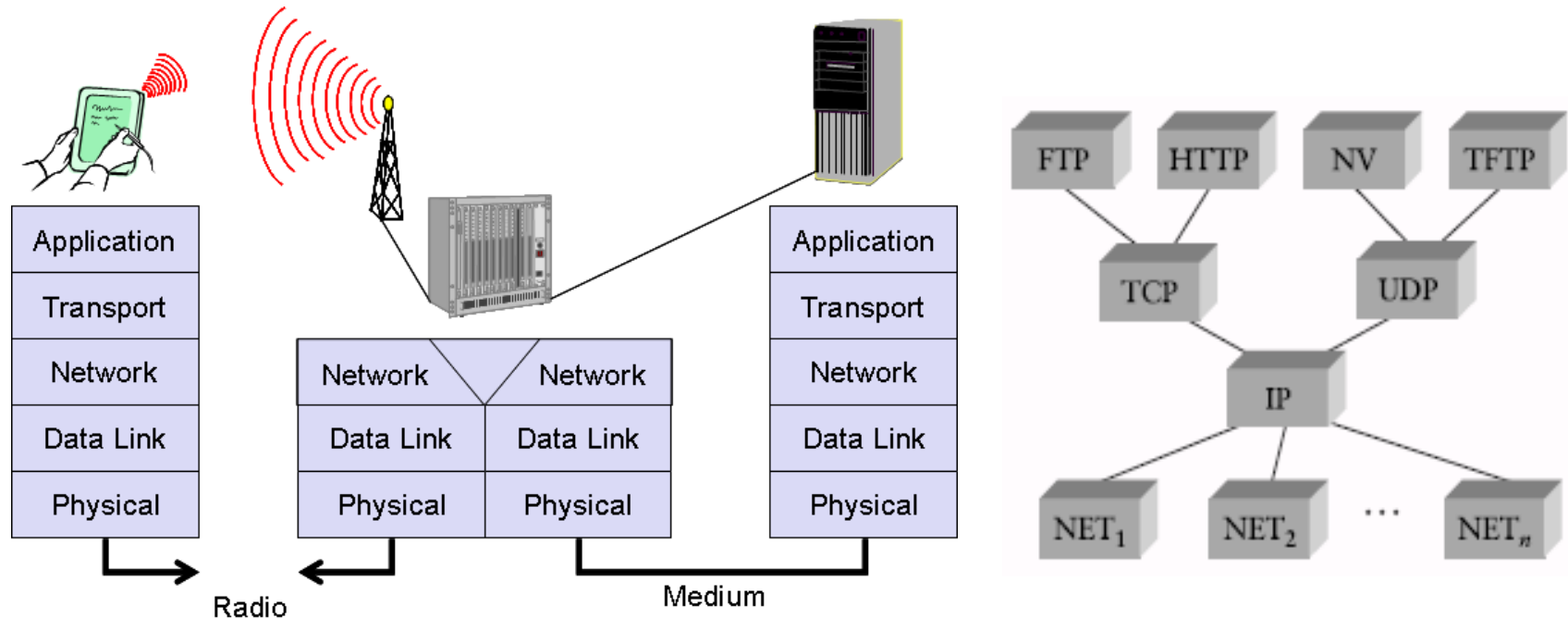
Solution to Scalability: Network of Networks

- ❑ Forwarding packets to networks from networks: smaller forwarding tables

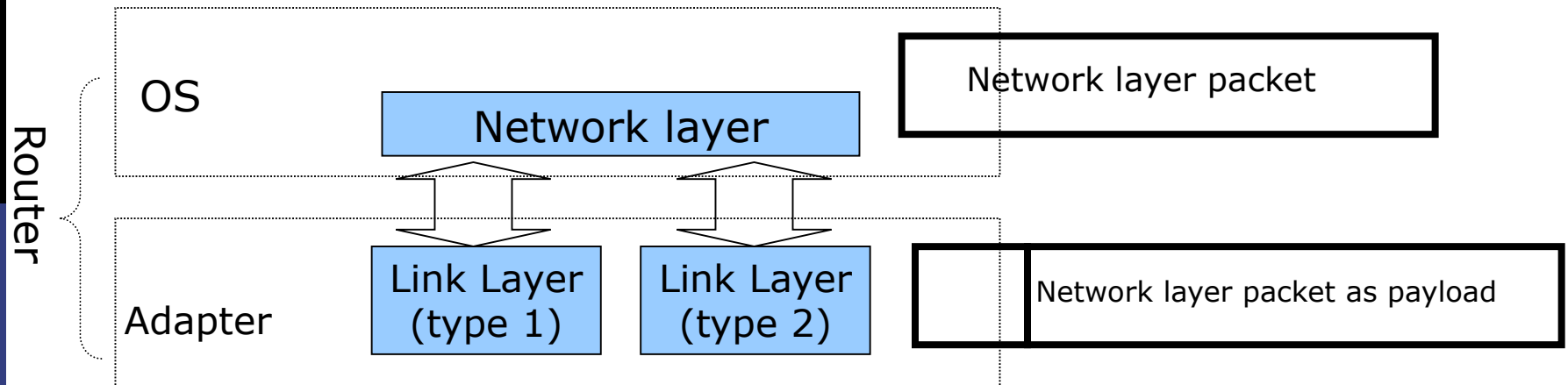


Solution to Heterogeneity: layered architecture and hourglass design

- ❑ Network layer encapsulates the heterogeneity and the complexity of data link and physical layers (or the host-to-network layer)

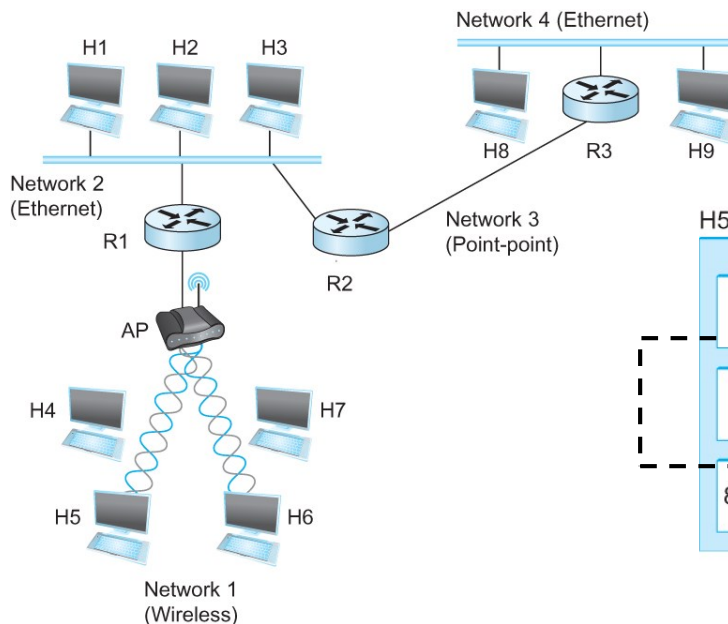


Network Layer and Lower Layers

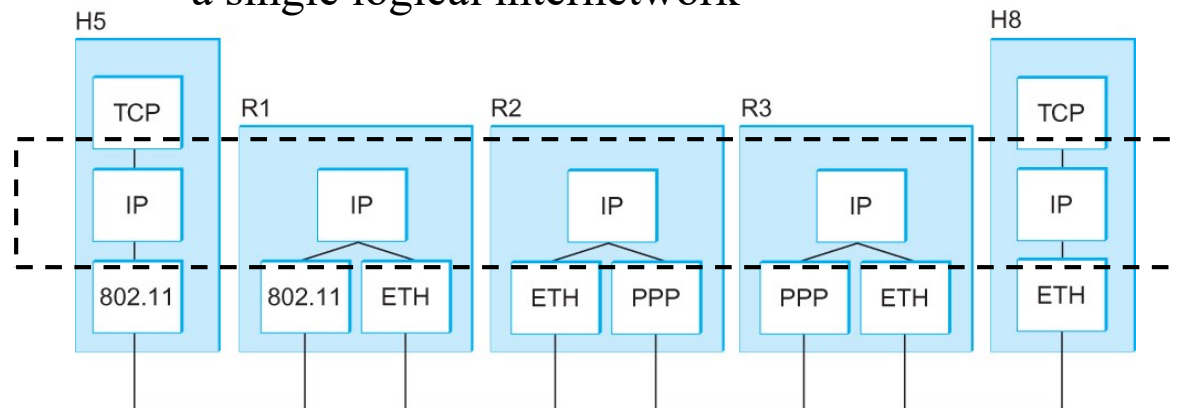


Internet Protocol

- ❑ IP = Internet Protocol
- ❑ Key tool used today to build scalable, heterogeneous internetworks
 - Routers forward packets to “networks”: forwarding tables can be smaller
 - Above link layer: can cope with different link layer technology



- Runs on all the nodes in a collection of networks
- Defines the infrastructure
- Allows these nodes and networks to function as a single logical internetwork



Case Study: The Internet

- ❑ Global internetworks built on IP → The Internet ≠ internet
- ❑ Using Internet Protocol (IP) as a case study
 - Datagram forwarding and service model
 - IP packet format and global IP addressing scheme
 - Deal with Link layer and network layer interfacing
 - ❑ Packet fragmentation and assembly
 - ❑ Address translation
 - Other important issues
 - ❑ Host configuration
 - ❑ Error reporting

IP Service Model

❑ Packet Delivery Model

- Connectionless model for data delivery
- Best-effort delivery (unreliable service)
 - ❑ packets may be lost
 - ❑ packets may be delivered out of order
 - ❑ duplicate copies of a packet may be delivered
 - ❑ packets may be delayed for a long time

❑ Global Addressing Scheme

- Provides a way to identify all hosts in the network

Basic Data Structure: IP Packet

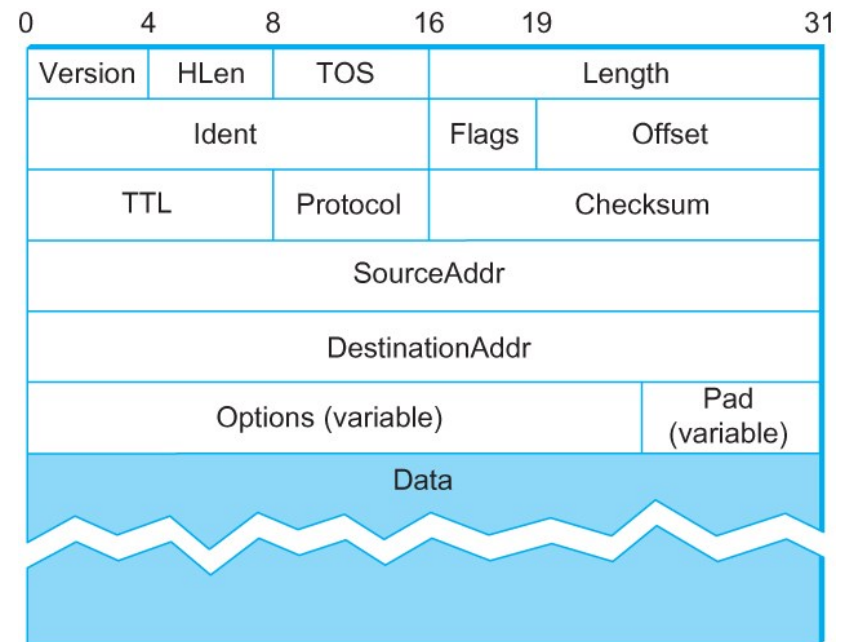
□ Design Goals

■ Attributes and purposes

- Support error detection and handling
- Support networks as a forwarding source and destinations
- Support different networking technologies
- Support multiplexing
- Support extensibility

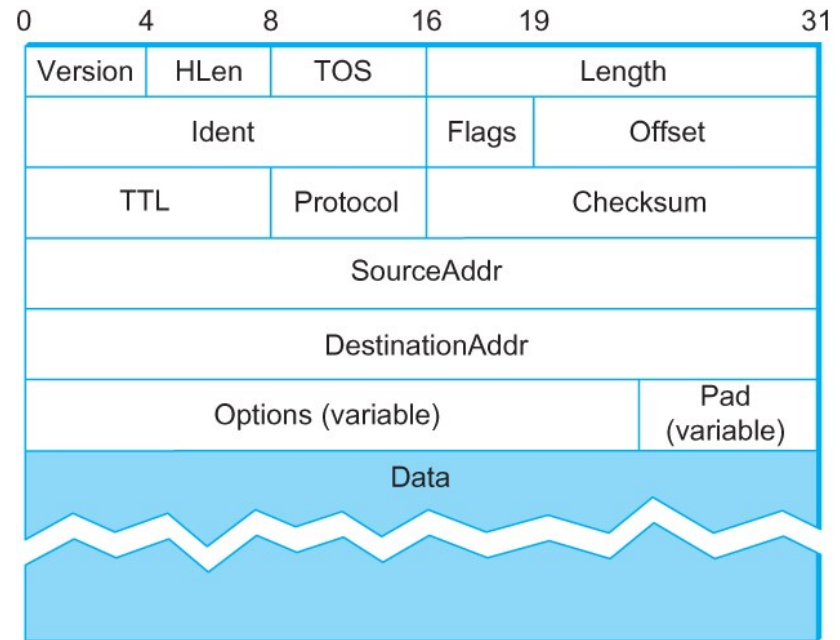
IP Packet Format (1)

- ❑ Discussing IP version 4 (IPv4). Discussing IP version 6 in later lessons
 - Convention used to illustrate IP packet
 - ❑ 32 bit words
 - ❑ Top word transmit first
 - ❑ Left-most byte transmit first



Packet Format

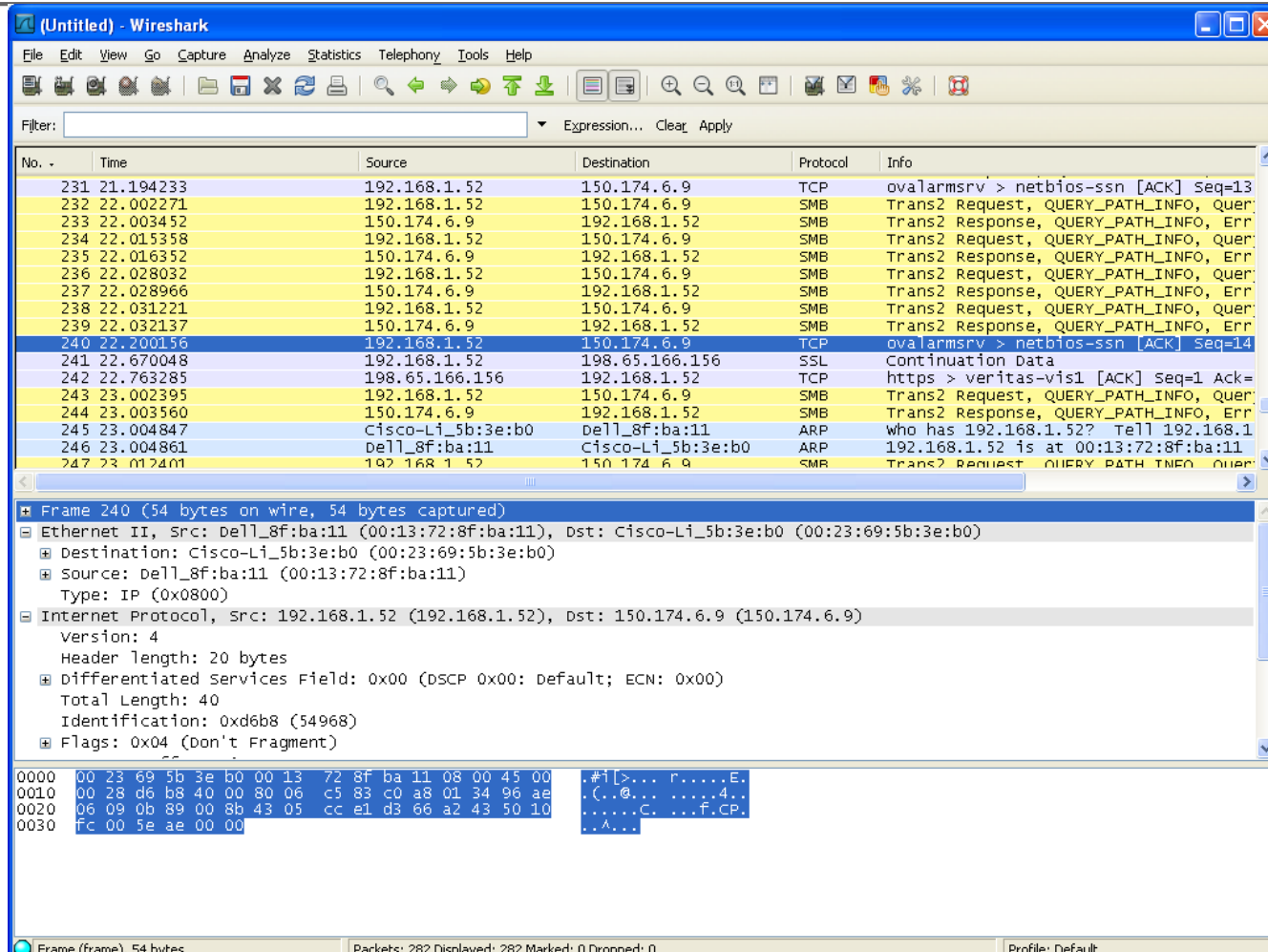
- ❑ Version (4): 4 or 6. The rest is for version 4. Discussing 6 in later lessons
- ❑ Hlen (4): number of 32-bit words in header
- ❑ TOS (8): type of service (not widely used)
- ❑ Length (16): number of bytes in this datagram
- ❑ Ident (16): used by fragmentation
- ❑ Flags/Offset (16): used by fragmentation
- ❑ TTL (8): number of hops this datagram has traveled
- ❑ Protocol (8): demux key (TCP=6, UDP=17)
- ❑ Checksum (16): of the header only
- ❑ DestAddr & SrcAddr (32)



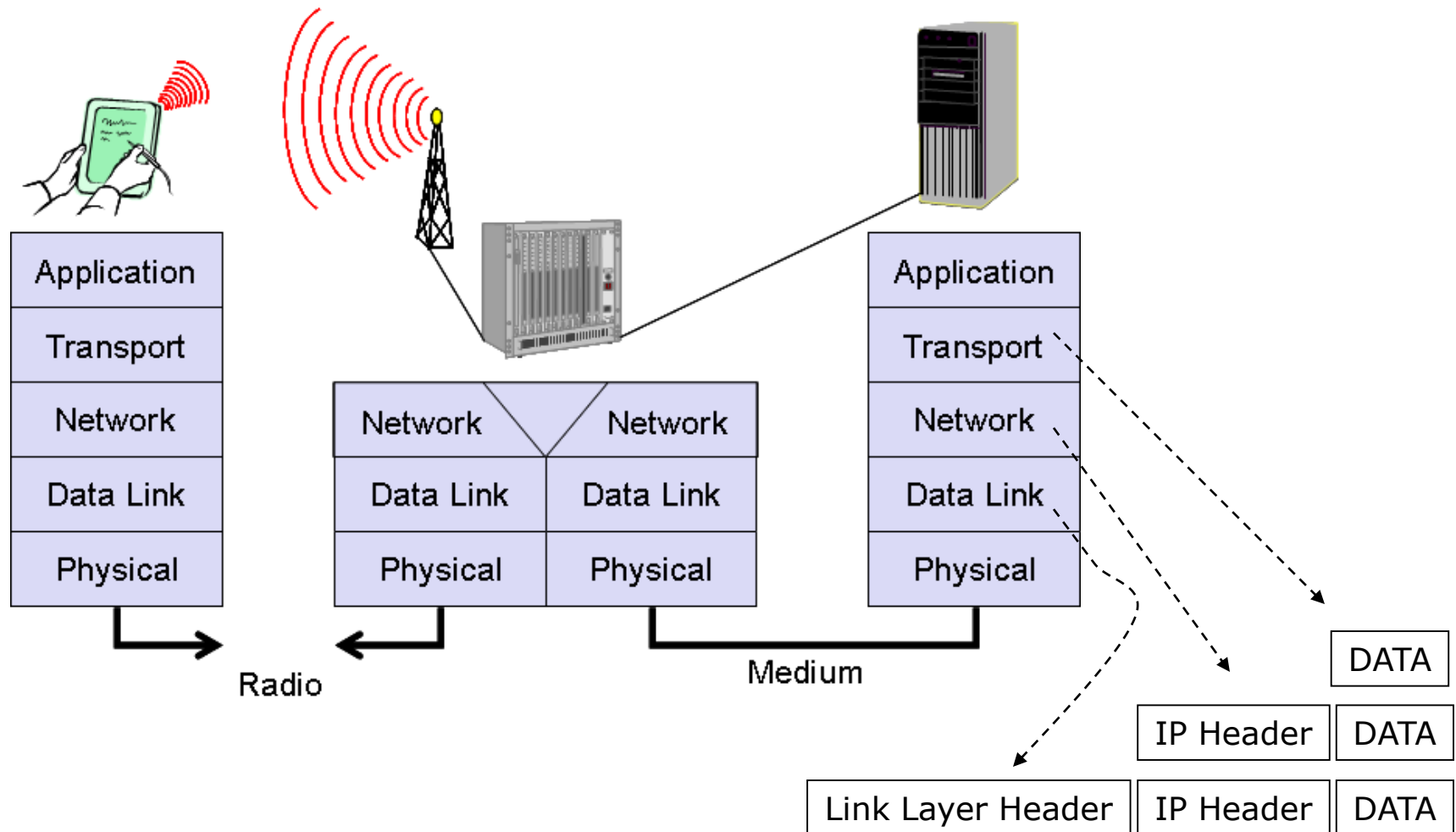
Capturing an IP Packet

- ❑ And examining it ...
- ❑ Use the *ethercap* application (a part of homework 1)
- ❑ Use Wireshark
- ❑ Use Microsoft Network Monitor ([Message Analyzer](#))
- ❑ Use tcpdump
- ❑ Use *libpcap* in your own application
- ❑

Using Wireshark



IP Packet



A Captured IP Packet

```
hchen@hecuba: ~/Project/csed/csed/networks/ethernet
Interface: eth0
0000  00 23 ae 7b 49 11 00 13 72 8f ba 11 08 00 45 00  .#.{I...r....E.
0010  00 28 78 41 40 00 80 06 fe d4 c0 a8 01 34 c0 a8  .(xM@.....4..
0020  01 35 07 e3 00 16 b6 c0 0a da b6 1e 1a b7 50 10  .5.....P.
0030  f1 80 a0 30 00 00 00 00 00 00 00 00 00 00 00  ...0.....
Interface: eth0
0000  00 23 ae 7b 49 11 00 13 72 8f ba 11 08 00 45 00  .#.{I...r....E.
0010  00 28 78 42 40 00 80 06 fe d3 c0 a8 01 34 c0 a8  .(xB@.....4..
0020  01 35 07 e3 00 16 b6 c0 0a da b6 1e 1c 17 50 10  .5.....P.
0030  fc 00 94 50 00 00 00 00 00 00 00 00 00 00 00  ...P.....
Interface: eth0
0000  00 23 ae 7b 49 11 00 13 72 8f ba 11 08 00 45 00  .#.{I...r....E.
0010  00 5c 78 43 40 00 80 06 fe 9e c0 a8 01 34 c0 a8  .\xC@.....4..
0020  01 35 07 e3 00 16 b6 c0 0a da b6 1e 1c 17 50 18  .5.....P.
0030  fc 00 7f 6a 00 00 3e ad c9 12 24 58 f0 fd e7 96  ...j..>...$X....
0040  79 09 a2 b5 73 4a 88 a1 20 1d c4 87 44 72 e0 8c  y...sJ...Dr..
0050  67 02 37 a3 de f4 c8 cc ec 18 dc ca d1 3a 2a 33  g.7.....:*3
0060  a6 75 c4 14 4d 57 1f 1a 0c f9  .u..MW....
Interface: eth0
0000  00 13 72 8f ba 11 00 23 ae 7b 49 11 08 00 45 10  ..r....#.{I...E.^C
0000  00 13 72 8f ba 11 00 23 ae 7b 49 11 08 00 45 10  ..r....#.{I...E.

User pressed CTRL-C. Exiting ...
[hchen@hecuba ethernet]$
```

Ethernet Protocol Numbers

```
VIM - /usr/include/net/ethernet.h

/* 10Mb/s ethernet header */
struct ether_header
{
    u_int8_t  ether_dhost[ETH_ALEN]; /* destination eth addr */
    u_int8_t  ether_shost[ETH_ALEN]; /* source ether addr */
    u_int16_t ether_type;             /* packet type ID field */
} __attribute__((__packed__));

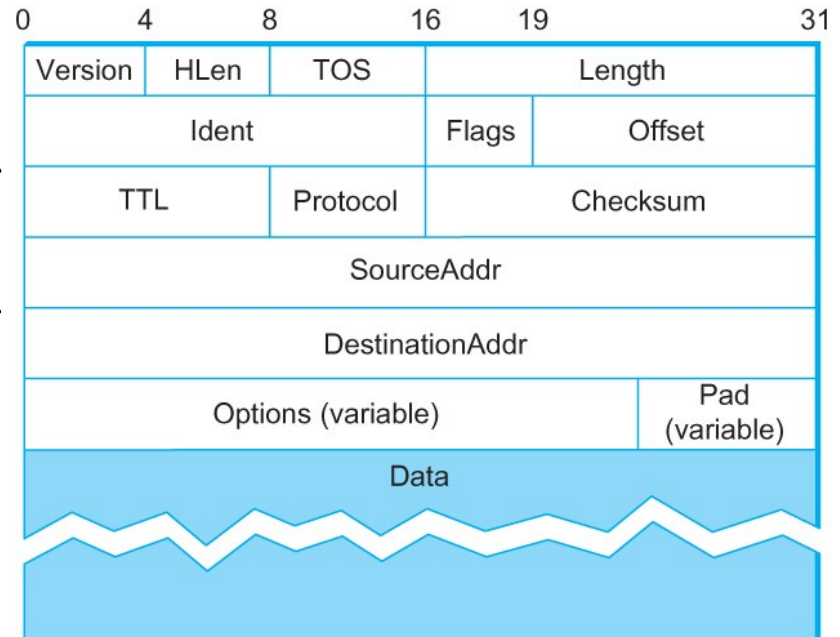
/* Ethernet protocol ID's */
#define ETHERTYPE_PUP      0x0200      /* Xerox PUP */
#define ETHERTYPE_SPRITE  0x0500      /* Sprite */
#define ETHERTYPE_IP      0x0800      /* IP */
#define ETHERTYPE_ARP     0x0806      /* Address resolution */
#define ETHERTYPE_REVARP  0x8035      /* Reverse ARP */
#define ETHERTYPE_AT      0x809B      /* AppleTalk protocol */
#define ETHERTYPE_AARP    0x80F3      /* AppleTalk ARP */
#define ETHERTYPE_VLAN    0x8100      /* IEEE 802.1Q VLAN tagging */
#define ETHERTYPE_IPX     0x8137      /* IPX */
#define ETHERTYPE_IPV6    0x86dd      /* IP protocol version 6 */
#define ETHERTYPE_LOOPBACK 0x9000      /* used to test interfaces */

#define ETHER_ADDR_LEN    ETH_ALEN    /* size of ethernet addr */
"/usr/include/net/ethernet.h" [readonly] 84L, 3221C          49,1          60%
```

Exercise L10-1

□ Below shows a captured Ethernet frame

- Q1: what is the length in bytes of the largest IP packet?
- Q2: what is the length in bytes of the smallest IP packet?
- Q3: what are the values (bits) of each field in the IP packet below (underlined). Which byte is the last byte of this IP packet?



```
0000  00 23 ae 7b 49 11 00 13 72 8f ba 11 08 00 45 00
0010  00 28 78 41 40 00 80 06 fe d4 c0 a8 01 34 c0 a8
0020  01 35 07 e3 00 16 b6 c0 0a da b6 1e 1a b7 50 10
0030  f1 80 a0 30 00 00 00 00 00 00 00 00 00 00 00
```

IP Fragmentation and Reassembly

- ❑ Each network has some MTU (Maximum Transmission Unit)
 - Ethernet (1500 bytes), FDDI (4500 bytes)
- ❑ Strategy
 - Fragmentation occurs in a router when it receives a datagram that it wants to forward over a network which has (MTU < datagram)
 - Reassembly is done at the receiving host
 - All the fragments carry the same identifier in the *Ident* field
 - Fragments are self-contained datagrams
 - IP does not recover from missing fragments

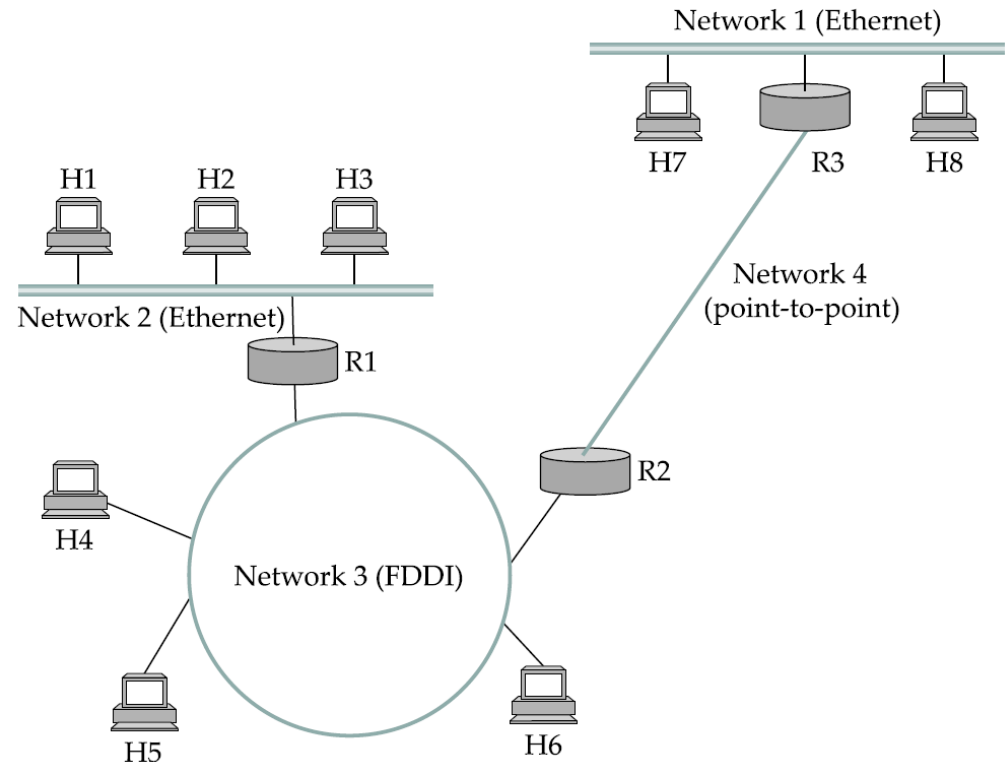
IP Fragmentation and Reassembly: Example

□ IP packet

- Data: 1400 bytes
- IP header: 20 bytes

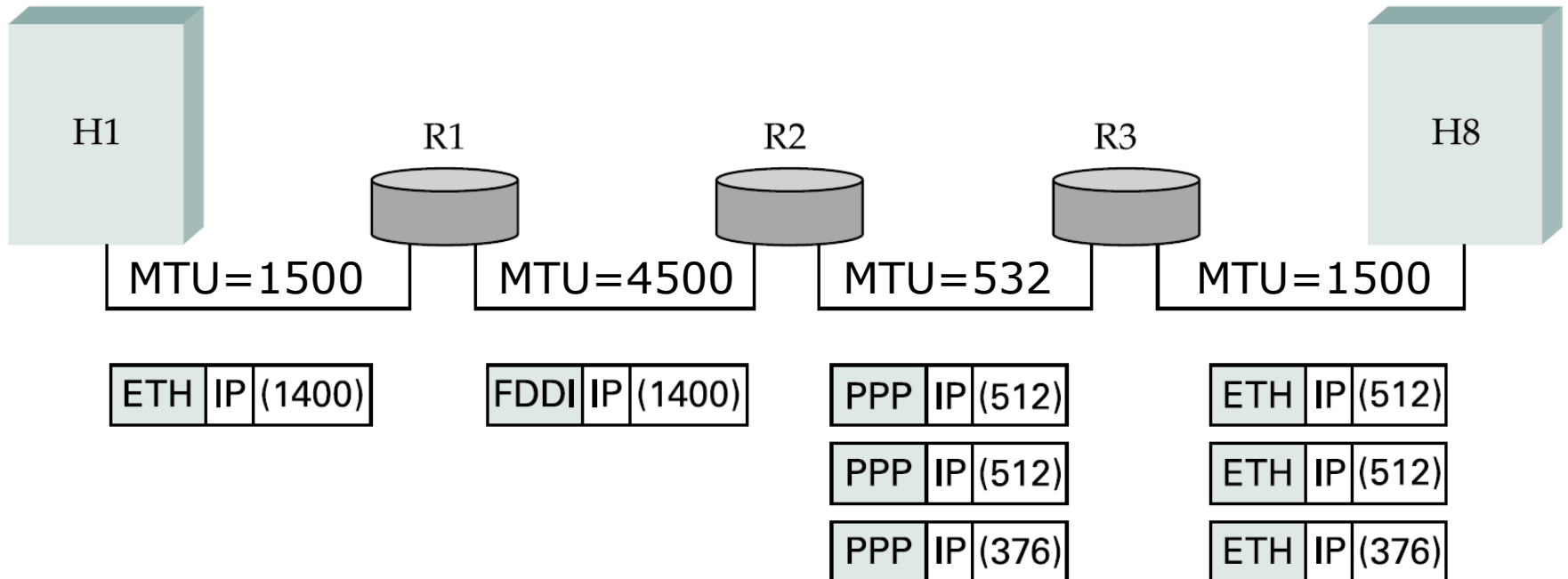
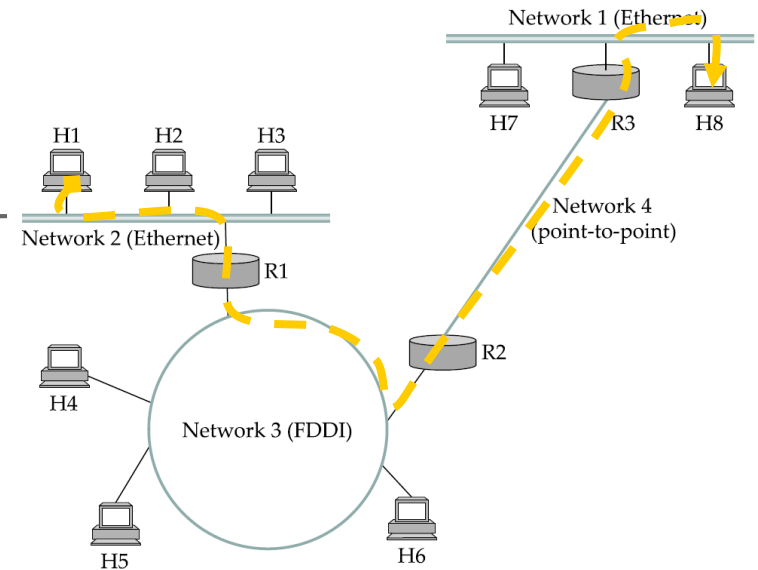
□ MTU

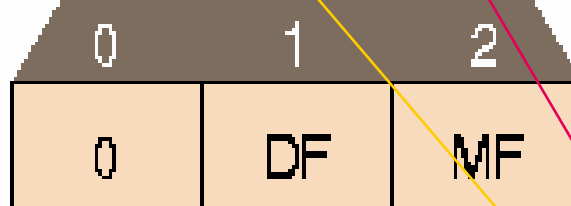
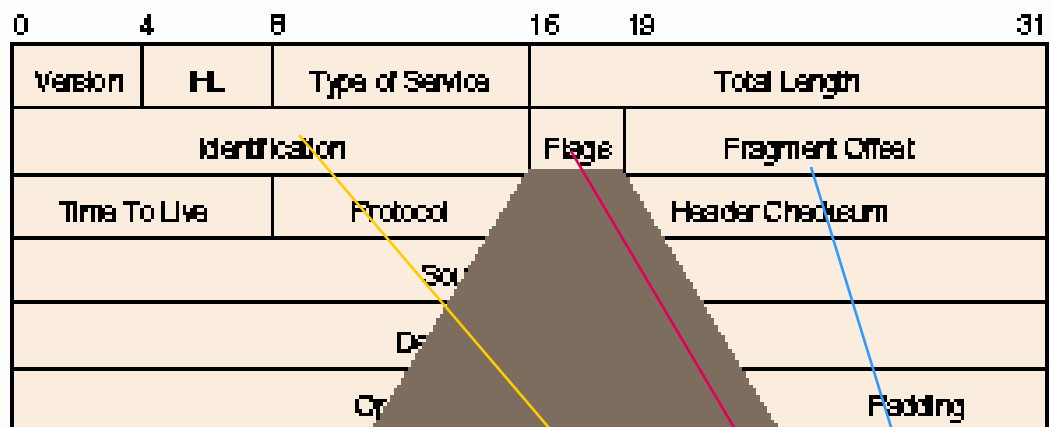
- Ethernet=1500
- FDDI=4500
- PPP=532



Example

IP packet at H1
Data: 1400 bytes
IP header: 20 bytes





```

0000  00 23 ae 7b 49 11 00 13 72 8f ba 11 08 00 45 00
0010  00 28 78 41 40 00 80 06 fe d4 c0 a8 01 34 c0 a8
0020  01 35 07 e3 00 16 b6 c0 0a da b6 1e 1a b7 50 10
0030  f1 80 a0 30 00 00 00 00 00 00 00 00 00 00 00
  
```

IP packet begin

IP packet ends

Bit 0: reserved, must be zero

Bit 1: (DF) 0 = May Fragment, 1 = **Don't Fragment**.

Bit 2: (MF) 0 = Last Fragment, 1 = **More Fragments**.

Source: <http://www.freesoft.org/CIE/Course/Section3/7.htm>

Example

Ident:

Same across all fragments

Unique for each packet

MF ($M_{ore} F_{ragments}$) bit in Flags:

set \rightarrow more fragments to follow

0 \rightarrow last fragment

Offset

in 8-byte chunks

Start of header				
Ident = x			0	Offset = 0
Rest of header				
1400 data bytes				

Fragmented
into three
fragments

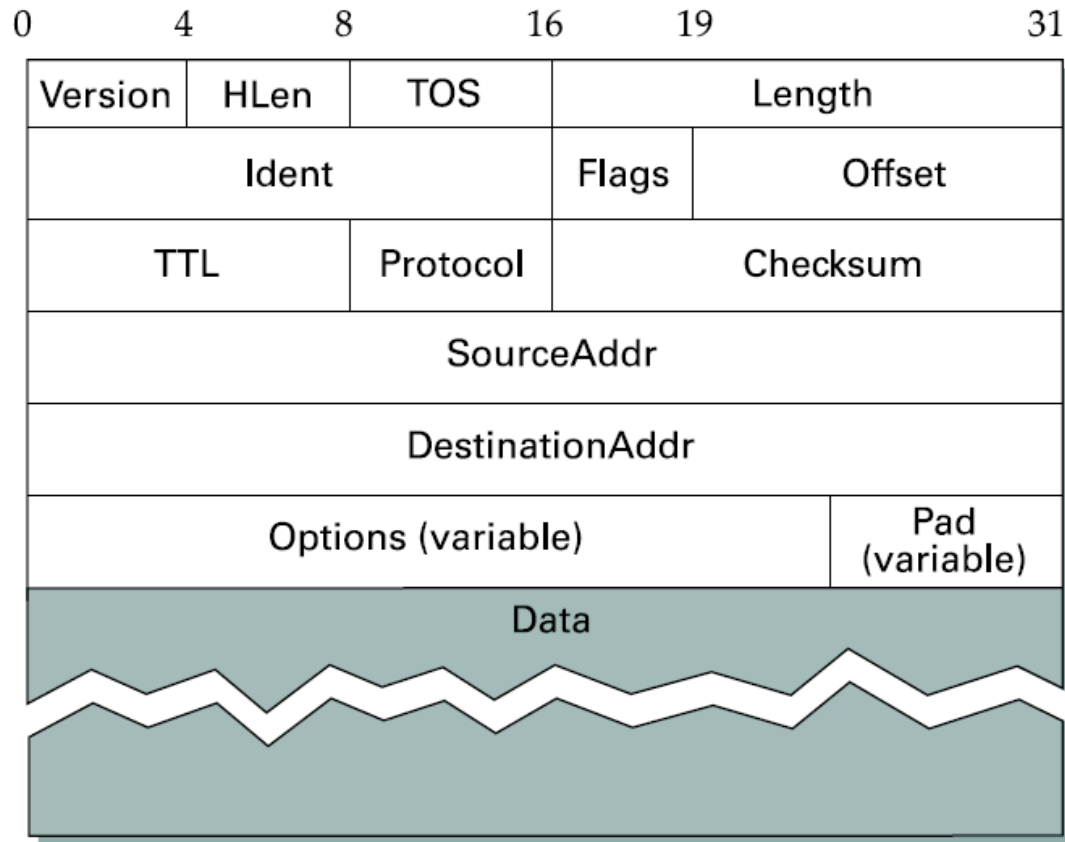
Q: why 8-byte chunks?

Start of header				
Ident = x			1	Offset = 0
Rest of header				
512 data bytes				

Start of header				
Ident = x			1	Offset = 64
Rest of header				
512 data bytes				

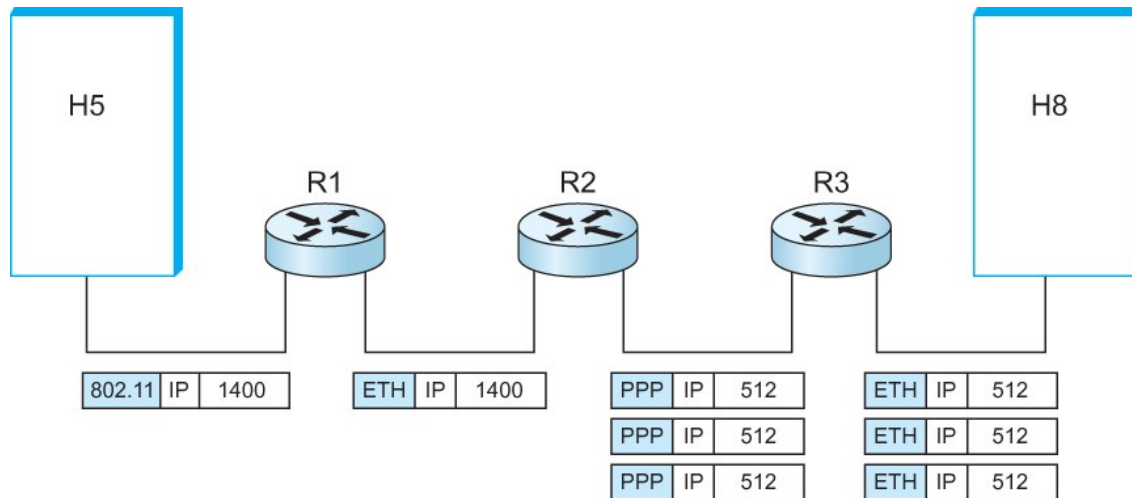
Start of header				
Ident = x			0	Offset = 128
Rest of header				
376 data bytes				

Hint for “*Why 8-byte Chunk?*”



IP Fragmentation and Reassembly

- IP datagrams traversing the sequence of physical networks



(a)

Start of header			
Ident = x		0	Offset = 0
Rest of header			
1400 data bytes			

(b)

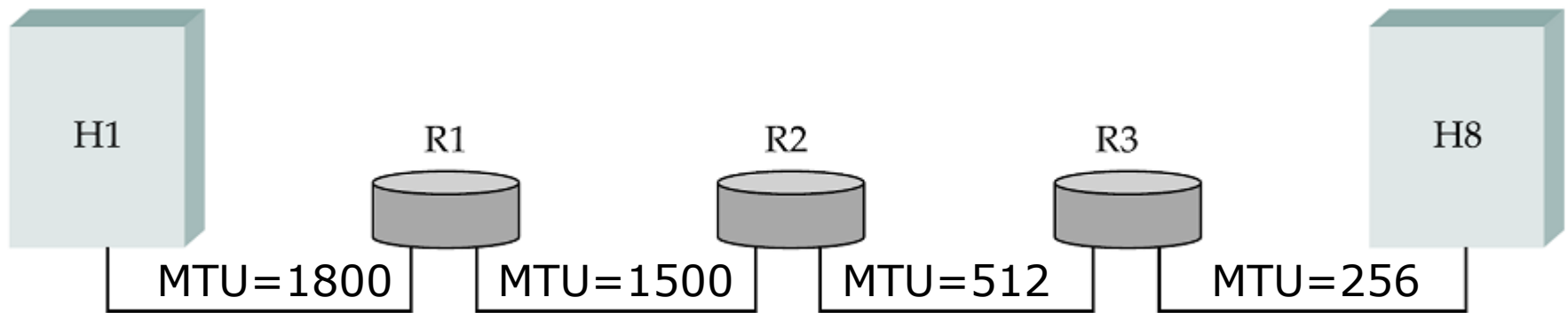
Start of header			
Ident = x		1	Offset = 0
Rest of header			
512 data bytes			

Start of header			
Ident = x		1	Offset = 64
Rest of header			
512 data bytes			

Start of header			
Ident = x		0	Offset = 128
Rest of header			
376 data bytes			

Exercise L10-2

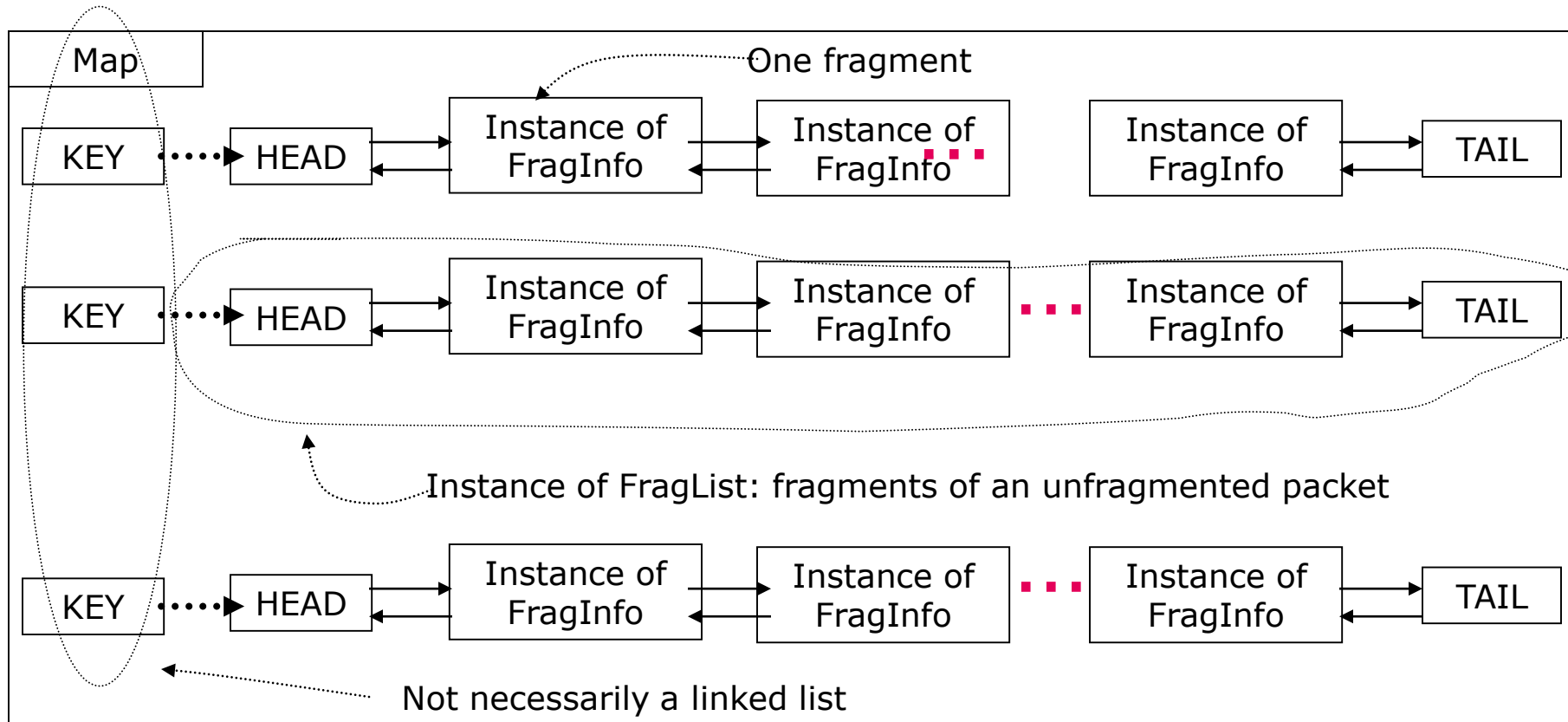
□ For an imaginary network below



- Q: H1 sends an IP packet of 1800 bytes including IP header to H8. Please show
1. IP datagrams traversing the sequence of physical networks graphed above
 2. Header fields of IP datagrams before entering and after leaving each router and hosts

Implementation of Reassembly

- ❑ Hints to understand the program (pp.243-247)



Global Addresses

❑ Internet Protocol (IP) Address

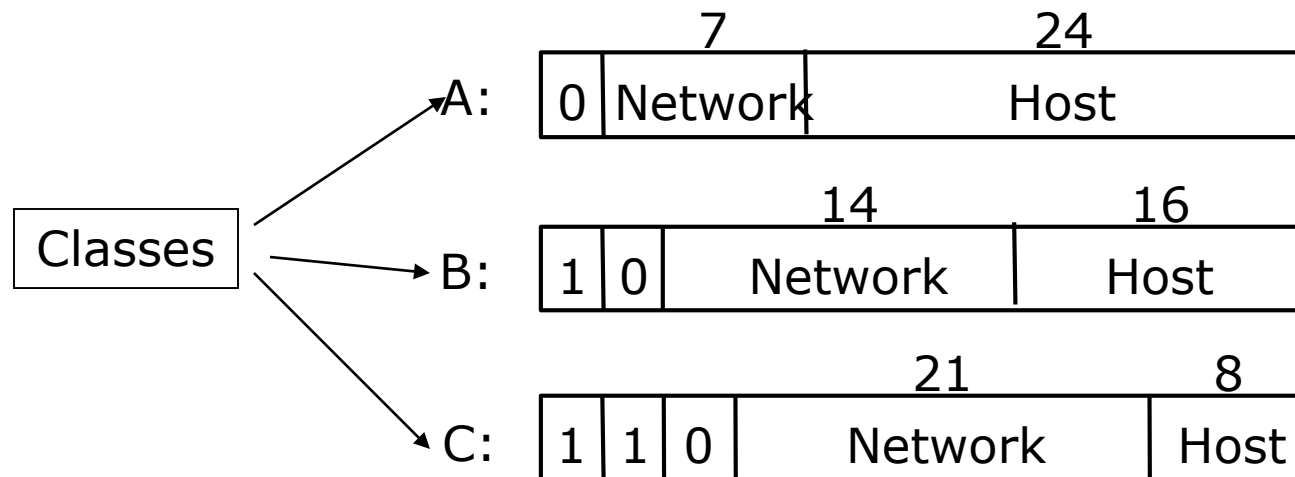
- Globally unique
- hierarchical: network + host

❑ IPv4

- 4 Billion IPv4 addresses, half are A type, $\frac{1}{4}$ is B type, and $\frac{1}{8}$ is C type
- 32 bit integer
- Human-readable form: IPv4 numbers-and-dots notation
 - ❑ 150.174.44.57
- Facing exhaustion of address space, moving to IPv6

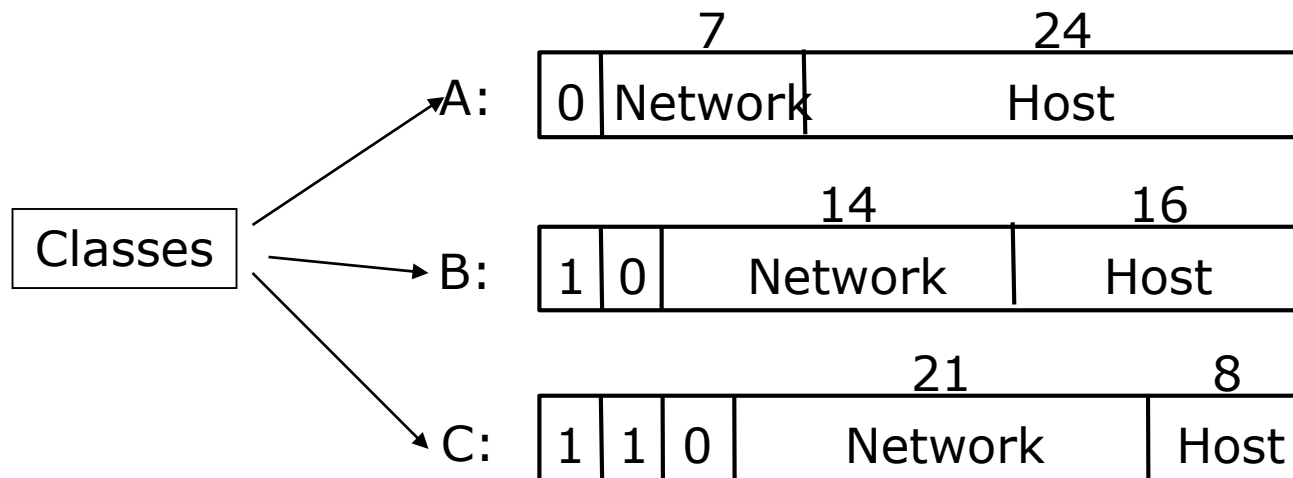
IPv4 Address Classes (Legacy)

- Classes (legacy)
 - To express networks



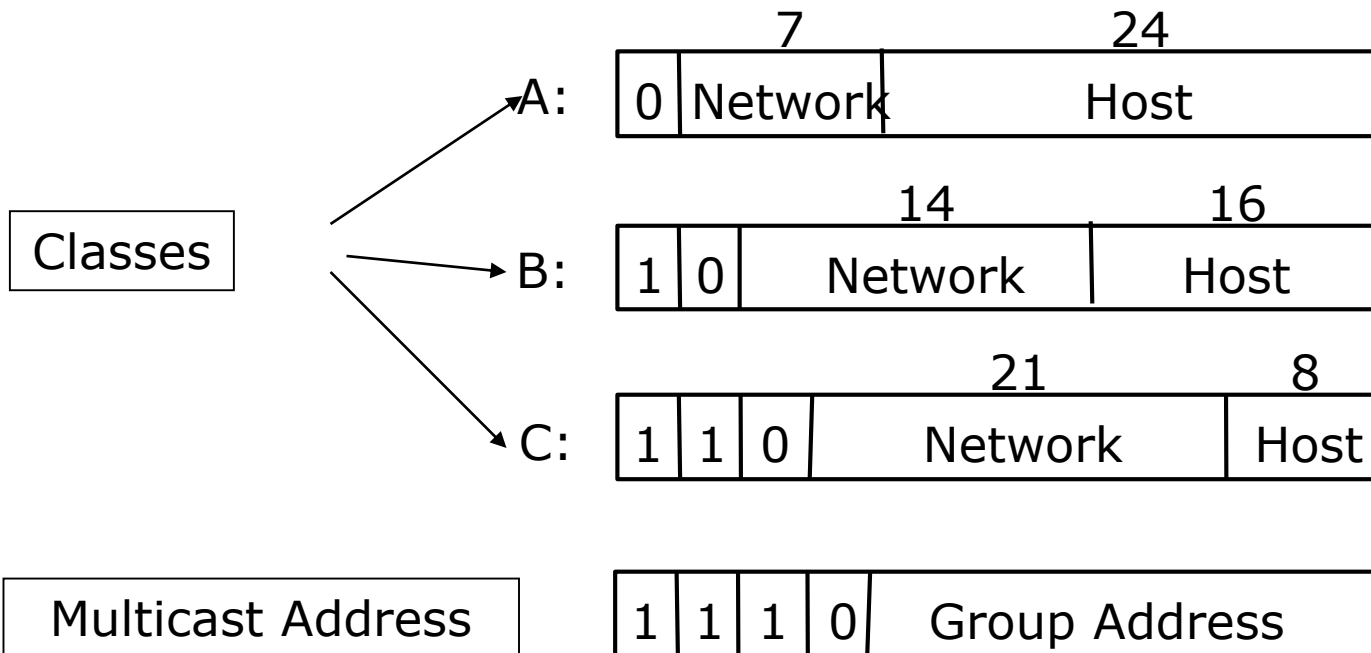
Broadcast and Multicast Addresses

- Do the classes of IP addresses discussed including any IP addresses starting with bits 111?



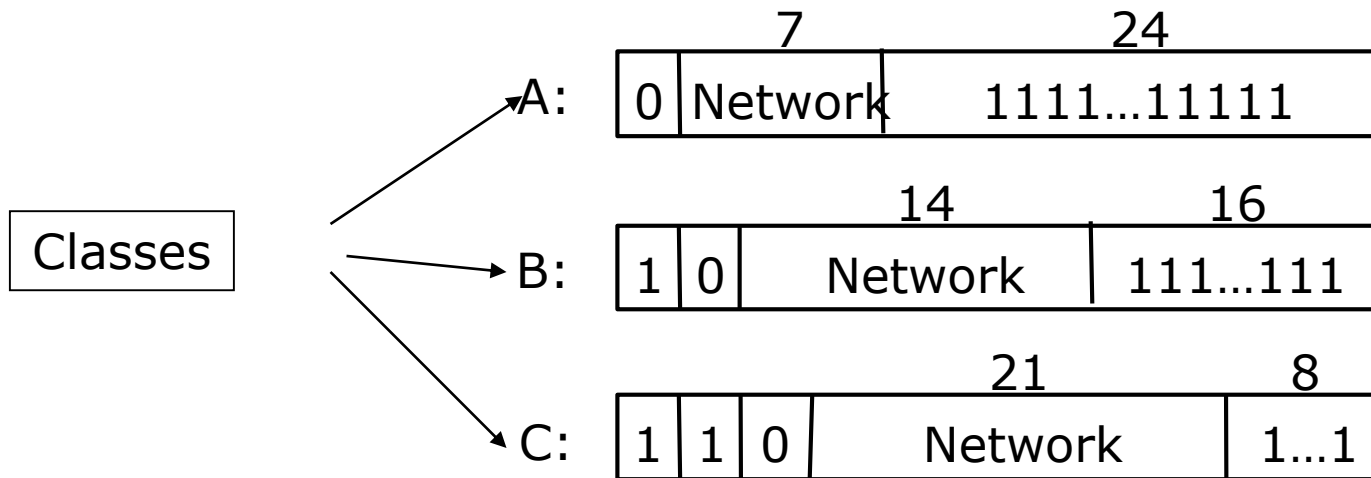
IPv4 Multicast Address

- Addresses starting with 1110



IPv4 Broadcast Address

- setting all the host bits to 1



IPv4 Address Spaces for Private Networks

□ See [RFC 1918](#)

□ Private networks

- 24-bit block 10.0.0.0–10.255.255.255
- 20-bit block 172.16.0.0–172.31.255.255
- 16-bit block 192.168.0.0–192.168.255.255

Link Local IPv4 Address

- ❑ See [RFC 3927](#)
- ❑ Link-Local IPv4 Address
 - ❑ 16-bit block 169.254.0.0–169.254.255.255

Exercise L10-3

- ❑ Find out IPv4 addresses of following hosts and indicate the class to which the IP addresses belong
 - www.vsu.edu
 - www.drsr.sk
 - www.google.com
- ❑ Remark
 - There are many ways to find out the IP address of a host given a domain name
 - ❑ Example: nslookup www.vsu.edu (which works on most platforms including Windows, Unix/Linux, and Mac OS X)
 - Convert the first number (from left) to a binary number, then take a look at the 1st, and/or 2nd, and/or 3rd bit

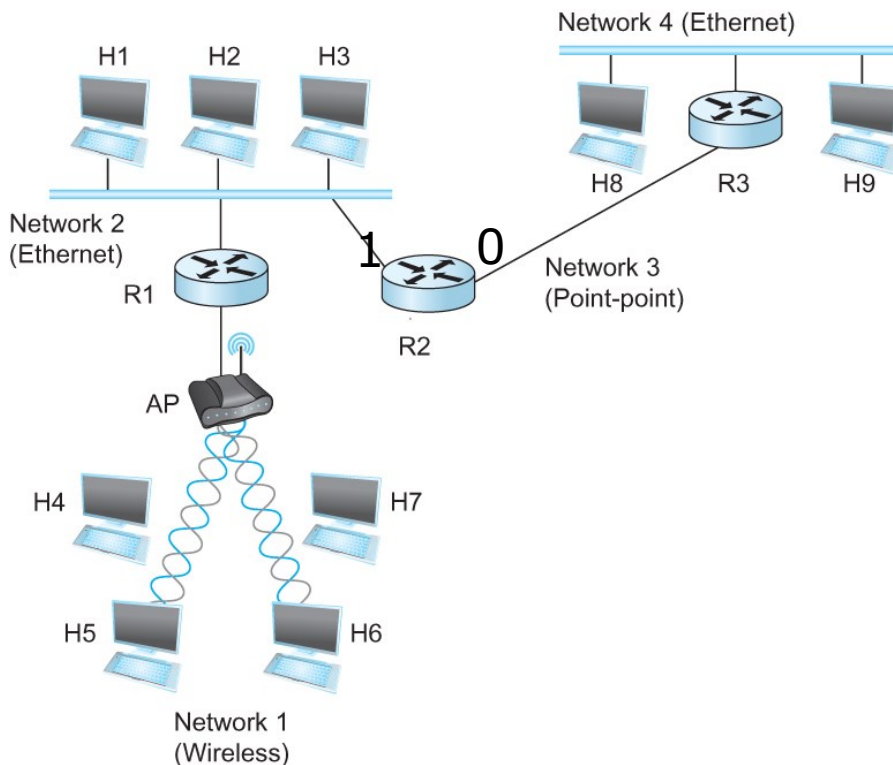
IP Datagram Forwarding

□ Strategy

- every datagram contains destination's address
- if directly connected to destination network, then forward to host
- if not directly connected to destination network, then forward to some router
- forwarding table maps network number into next hop
- each host has a default router
- each router maintains a forwarding table

Forwarding Table: Example

- Forwarding table at router R2 that has two interfaces 0 and 1



NetworkNum	NextHop
1	R1
2	Interface 1
3	Interface 0
4	R3

Forwarding Algorithm

□ Algorithm

```
if (NetworkNum of destination = NetworkNum of one of  
    my interfaces) then  
    deliver packet to destination over that interface  
else  
    if (NetworkNum of destination is in my forwarding  
        table) then  
        deliver packet to NextHop router  
    else  
        deliver packet to default router
```

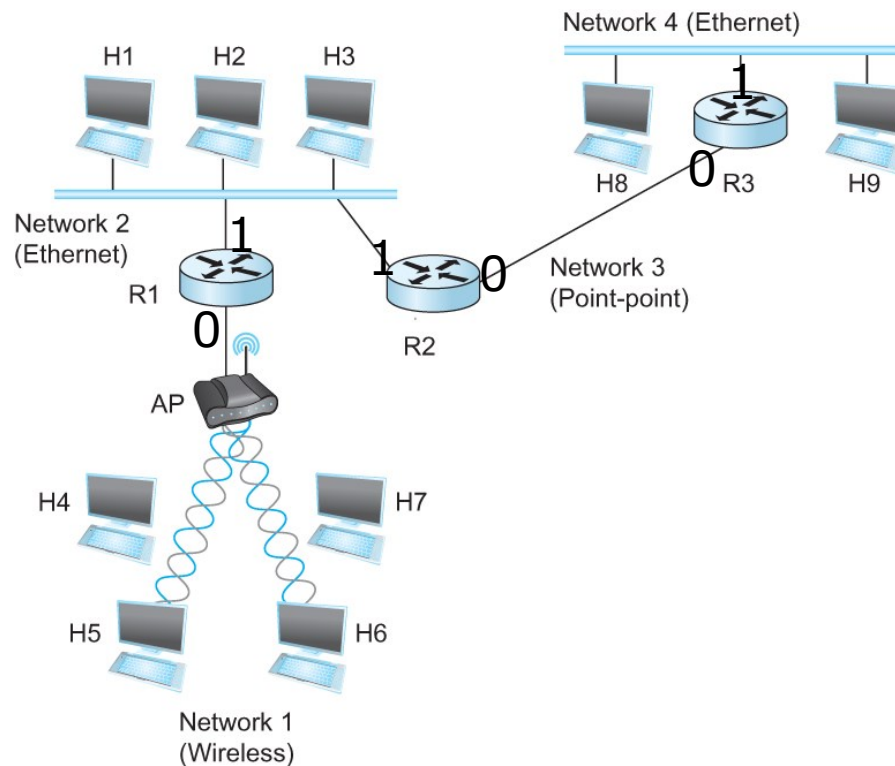

Forwarding Algorithm

□ For a host with only one interface and only a default router in its forwarding table, this simplifies to

```
if (NetworkNum of destination = my NetworkNum) then  
    deliver packet to destination directly  
else  
    deliver packet to default router
```

Exercise L10-4

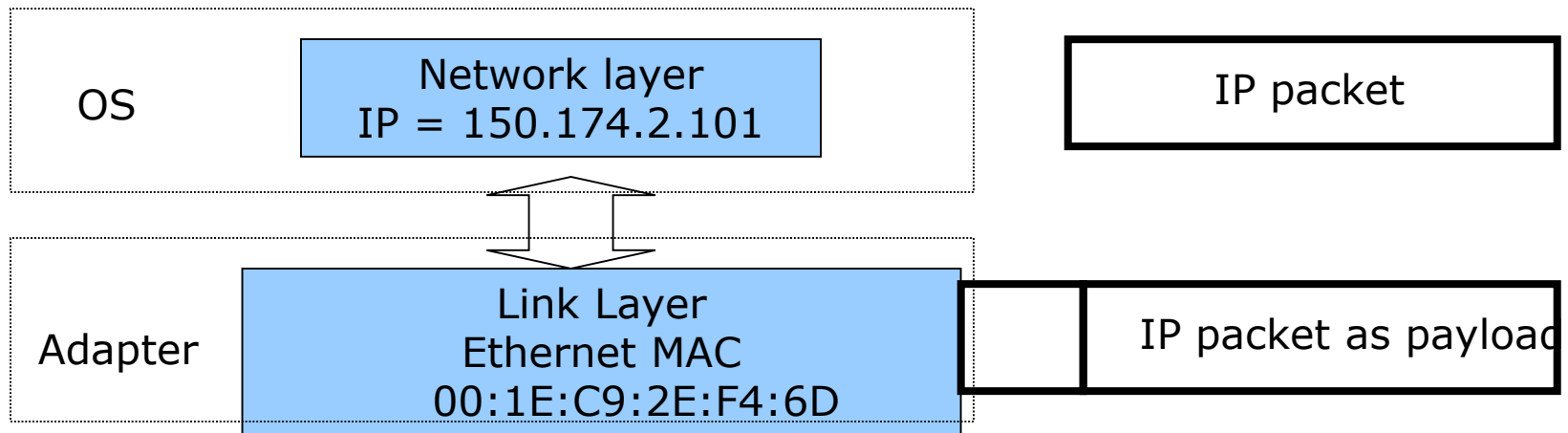
- Construct forwarding tables for routers R1 and R3. Interfaces of routers are marked



IP address to Physical Address Translation

□ Questions:

- In an IP network, an IP packet is the payload of one or more Ethernet frames. Who prepares the Ethernet frame headers which contain destination Ethernet/physical address of the destination node?
- How does the source node know the Ethernet/physical address of the destination node?

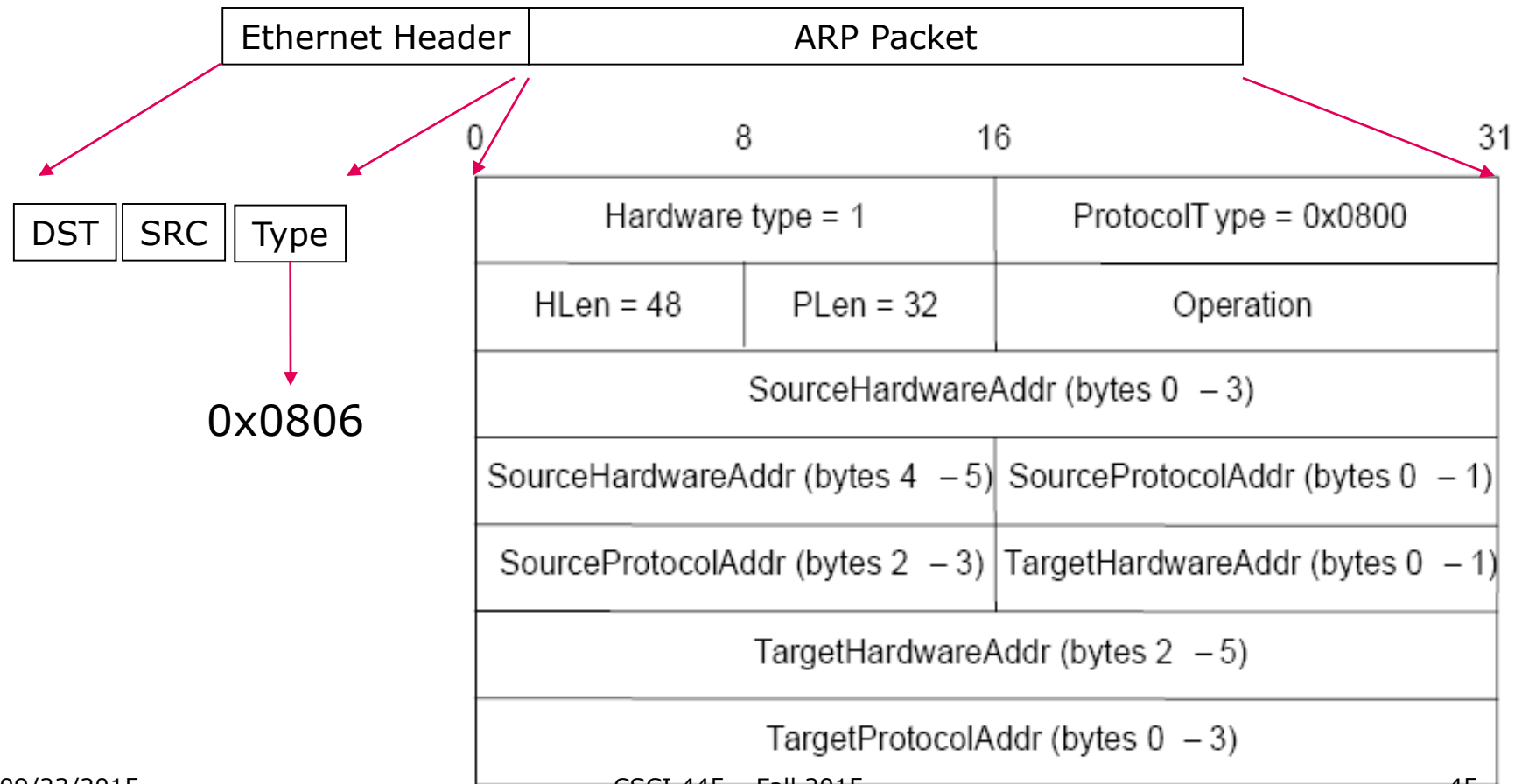


Map IP Addresses into Physical Addresses

- ❑ Map IP addresses into physical addresses
 - destination host
 - next hop router
- ❑ Techniques
 - encode physical address in host part of IP address
 - table-based
- ❑ ARP (Address Resolution Protocol)
 - table of IP to physical address bindings
 - broadcast request if IP address not in table
 - target machine responds with its physical address
 - table entries are discarded if not refreshed

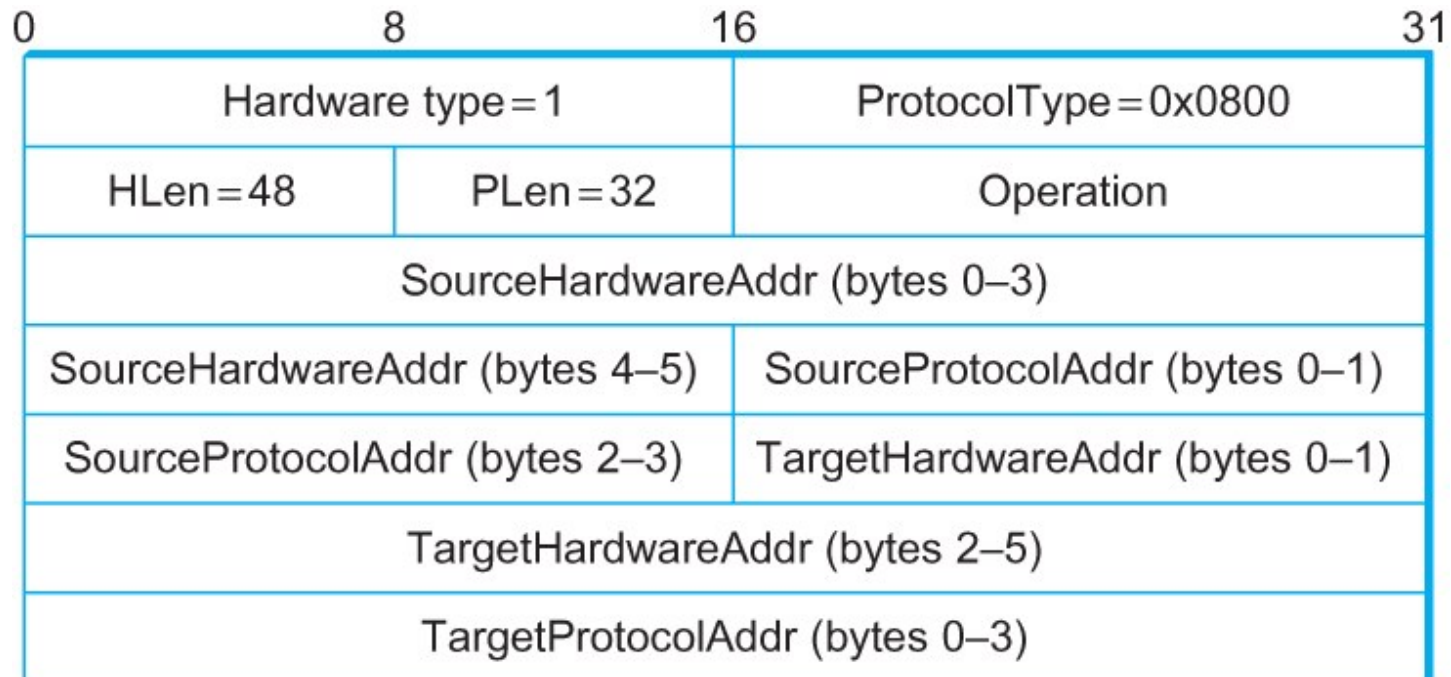
ARP Packet Format

- An ARP packet is the payload of a frame

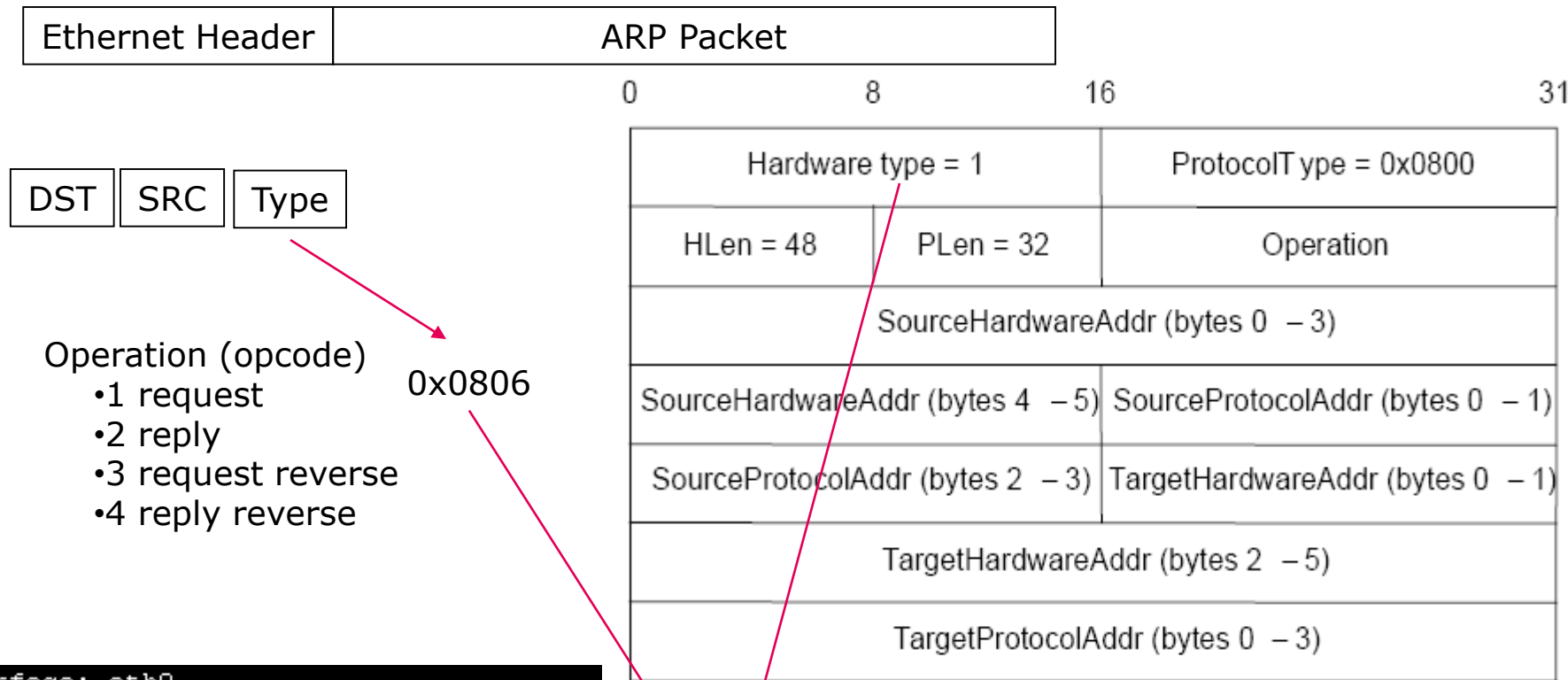


ARP Packet Format

- ❑ HardwareType: type of physical network (e.g., Ethernet)
- ❑ ProtocolType: type of higher layer protocol (e.g., IP)
- ❑ HLEN & PLEN: length of physical and protocol addresses
- ❑ Operation: request or response
- ❑ Source/Target Physical/Protocol addresses



ARP Packet: Examples



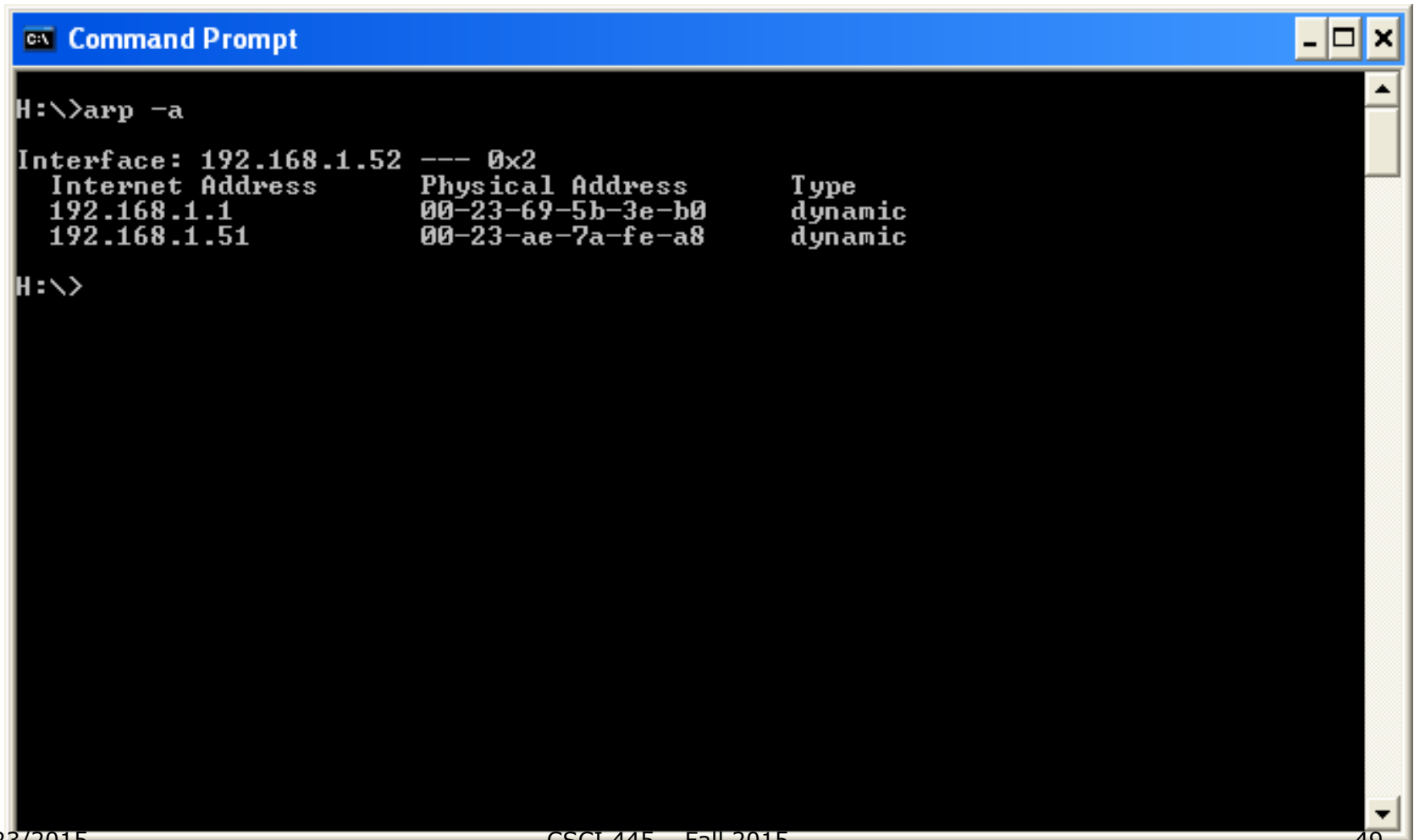
```

Interface: eth0
0000  ff ff ff ff ff ff 00 0c 29 a0 51 6d 08 06 00 01  .....).Qm....
0010  08 00 06 04 00 01 00 0c 29 a0 51 6d c0 a8 01 48  .....).Qm...H
0020  00 00 00 00 00 00 c0 a8 01 33  .....3
Interface: eth0
0000  ff ff ff ff ff ff 00 0c 29 a0 51 6d 08 06 00 01  .....).Qm....
0010  08 00 06 04 00 01 00 0c 29 a0 51 6d c0 a8 01 48  .....).Qm...H
0020  00 00 00 00 00 00 c0 a8 01 33  .....3
  
```

ARP: Discussion

- ❑ Prevent stalled entries
 - Table entries will timeout (~15 minutes)
 - Do not refresh table entries upon reference
- ❑ Fresh entries (reset timer)
 - Update table if already have an entry
- ❑ Reduce ARP messages
 - Update table with source when you are the target in ARP request messages

ARP Table in Practice (1)



The screenshot shows a Windows Command Prompt window with a blue title bar that reads "C:\ Command Prompt". The window contains the following text:

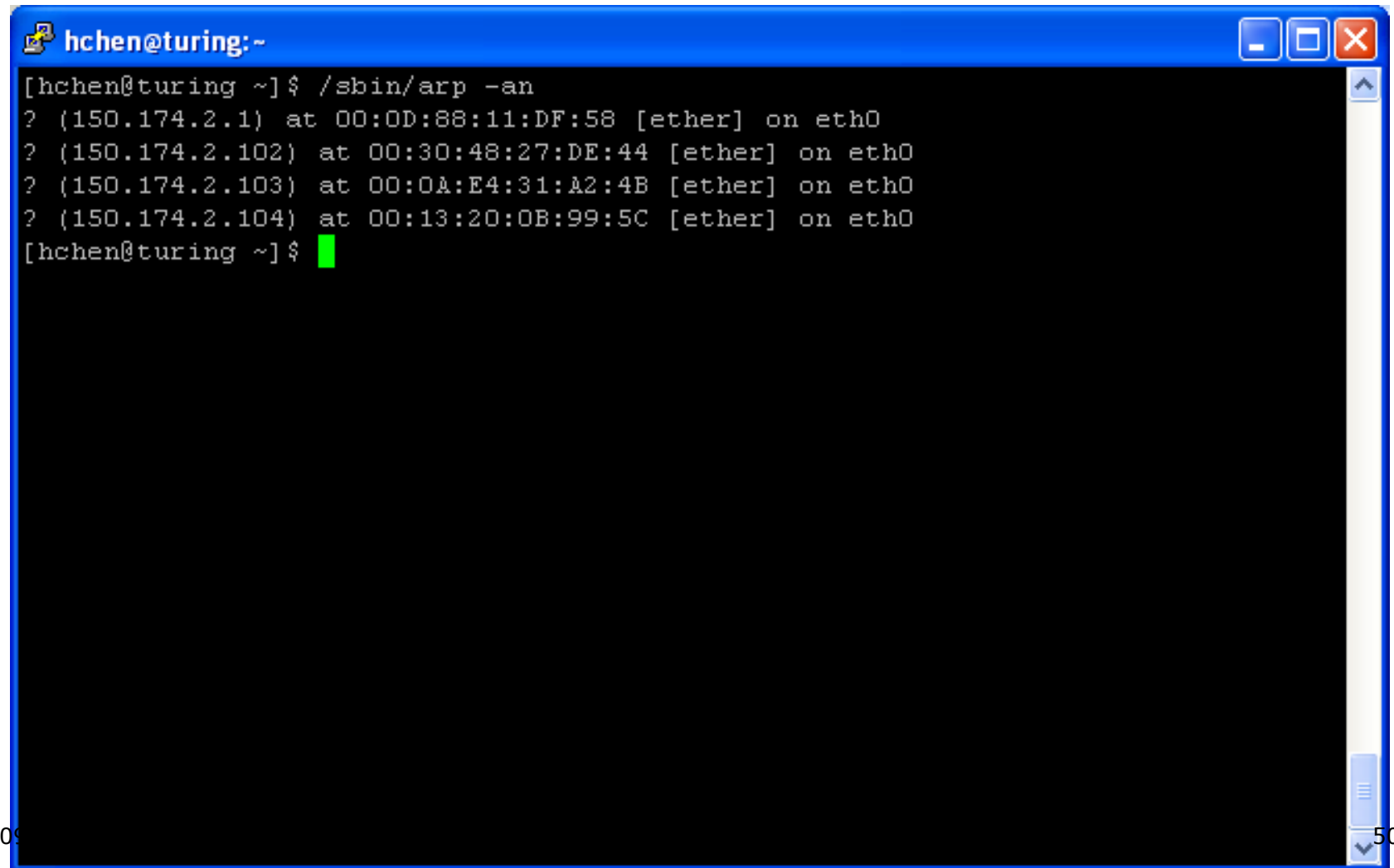
```
H:\>arp -a

Interface: 192.168.1.52 --- 0x2
    Internet Address      Physical Address      Type
    192.168.1.1           00-23-69-5b-3e-b0     dynamic
    192.168.1.51          00-23-ae-7a-fe-a8     dynamic

H:\>
```

The output displays the ARP table for the interface 192.168.1.52. It lists two entries: one for 192.168.1.1 with physical address 00-23-69-5b-3e-b0, and another for 192.168.1.51 with physical address 00-23-ae-7a-fe-a8. Both entries are marked as "dynamic".

ARP Table in Practice (2)

A terminal window with a blue title bar containing the text 'hchen@turing:~' and standard window control buttons. The terminal output shows the command '/sbin/arp -an' and its results, which list four IP addresses and their corresponding MAC addresses and interface names. A green cursor is visible on the line following the command.

```
[hchen@turing ~]$ /sbin/arp -an
? (150.174.2.1) at 00:0D:88:11:DF:58 [ether] on eth0
? (150.174.2.102) at 00:30:48:27:DE:44 [ether] on eth0
? (150.174.2.103) at 00:0A:E4:31:A2:4B [ether] on eth0
? (150.174.2.104) at 00:13:20:0B:99:5C [ether] on eth0
[hchen@turing ~]$
```

Host Configuration

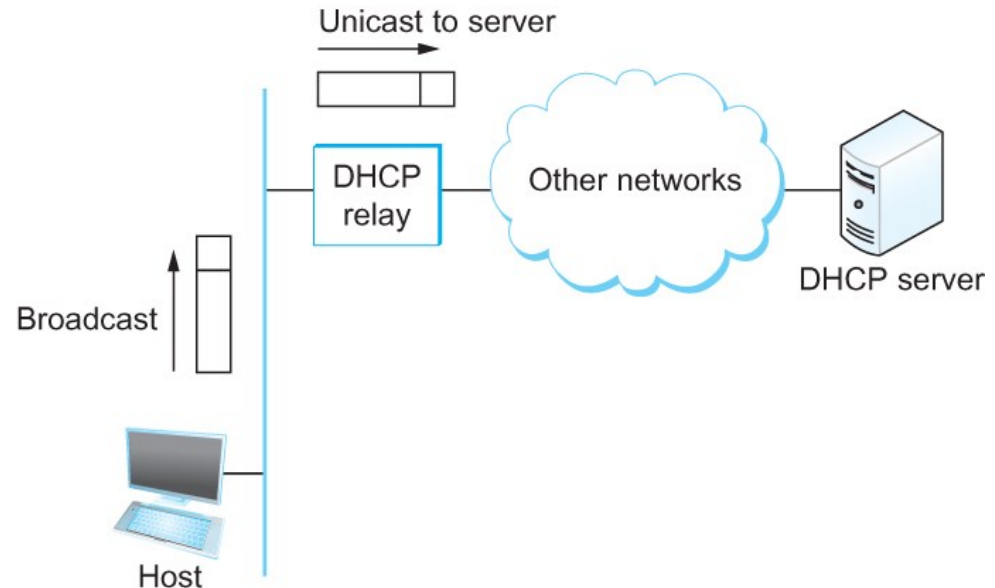
- ❑ Ethernet addresses are configured into network by manufacturer and they are unique
- ❑ IP addresses must be unique on a given internetwork but also must reflect the structure of the internetwork
- ❑ Most host Operating Systems provide a way to manually configure the IP information for the host
- ❑ Drawbacks of manual configuration
 - A lot of work to configure all the hosts in a large network
 - Configuration process is error-prone
- ❑ Automated Configuration Process is required

Dynamic Host Configuration Protocol (DHCP)

- ❑ DHCP server is responsible for providing configuration information to hosts
- ❑ There is at least one DHCP server for an administrative domain
- ❑ DHCP server maintains a pool of available addresses

DHCP

- ❑ Newly booted or attached host sends DHCPDISCOVER message to a special IP address (255.255.255.255)
- ❑ DHCP relay agent unicasts the message to DHCP server and waits for the response



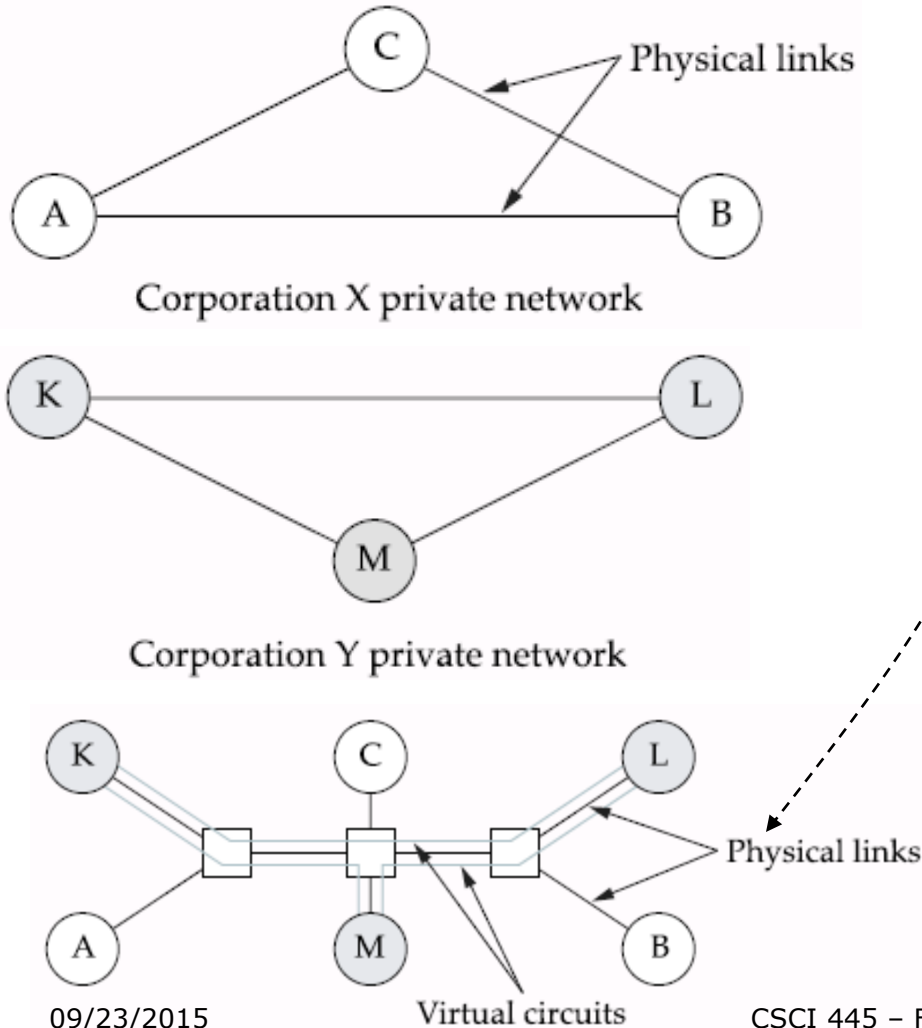
Internet Control Message Protocol (ICMP)

- ❑ Defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully
 - Destination host unreachable due to link /node failure
 - Reassembly process failed
 - TTL had reached 0 (so datagrams don't cycle forever)
 - IP header checksum failed
- ❑ ICMP-Redirect
 - From router to a source host
 - With a better route information

Virtual Networks and Tunnels

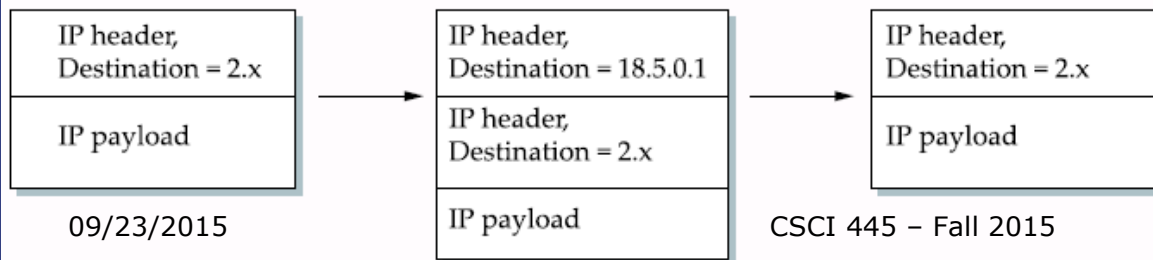
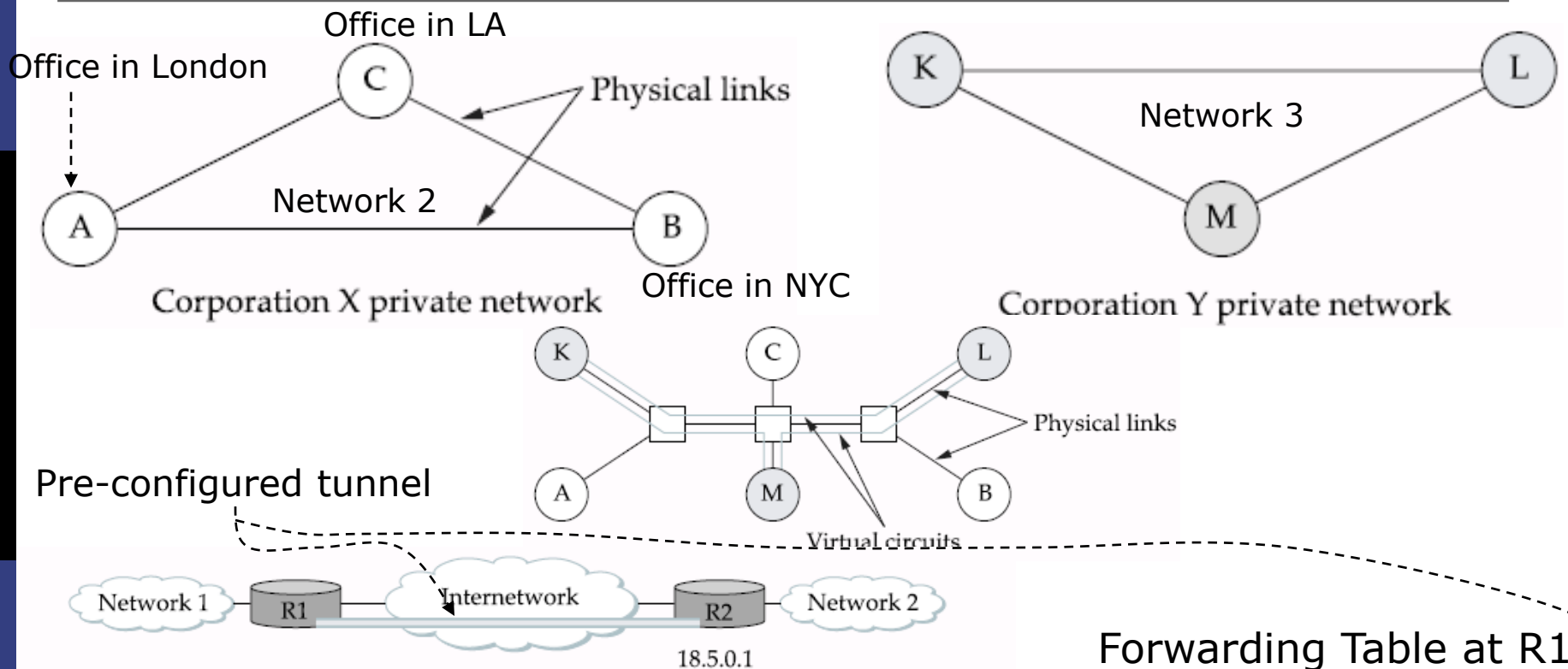
- ❑ Internetworks often have shared infrastructure networks
- ❑ Data packets may not be forwarded without restriction
- ❑ Virtual Private Networks (VPN)
 - VPN is a heavily overused and definitions vary
 - An “private” network utilizing an shared network infrastructure

Virtual Private Networks: Example



- ❑ Corporations X and Y want their own networks via “leased lines” belonging to other networks
- ❑ X wants to keep their data private
- ❑ So does Y
- ❑ X and Y have “virtual” private networks
- ❑ “virtualization” can be done on different layers
 - Layer 2 VPN
 - Layer 3 VPN

Virtual Private Networks via IP Tunneling



Forwarding Table at R1

NetworkNum	NextHop
1	Interface 0
2	Virtual interface 0
Default	Interface 1

Summary

- ❑ **i**nternet and the **I**nternet
- ❑ Global addressing scheme
- ❑ Packet fragmentation and assembly
- ❑ Best effort service model and datagram forwarding
- ❑ Address translation
- ❑ Host configuration
- ❑ Error reporting