TCP/IP Protocols

TCP/IP and the Internet

- In 1969
  - ARPA funded and created the "ARPA*net*" network
    - Robust, reliable, vendor-independent data communications
- In 1975
  - Convert from experimental to operational network
  - TCP/IP begun to be developed
- In 1983
  - The TCP/IP is adopted as Military Standards
  - ARPnet → MILNET + ARPnet = Internet
- In 1985
  - The NSF created the NSFnet to connect to Internet
- In 1990
  - ARPA passed out of existence, and in 1995, the NSFnet became the primary Internet backbone network

ARPA = Advanced Research Project Agency
NSF = National Science Foundation

Introduction (1)

> TCP/IP
  - **Used to provide data communication between hosts**
    - How to delivery data reliably
    - How to address remote host on the network
    - How to handle different type of hardware device
  - **4 layers architecture**
    - Each layer perform certain tasks
    - Each layer only need to know how to pass data to adjacent layers

Application	Telnet, FTP, e-mail, etc.
Transport	TCP, UDP
Network	IP, ICMP, IGMP
Link	device driver and interface card

Introduction (2)

> Four layer architecture

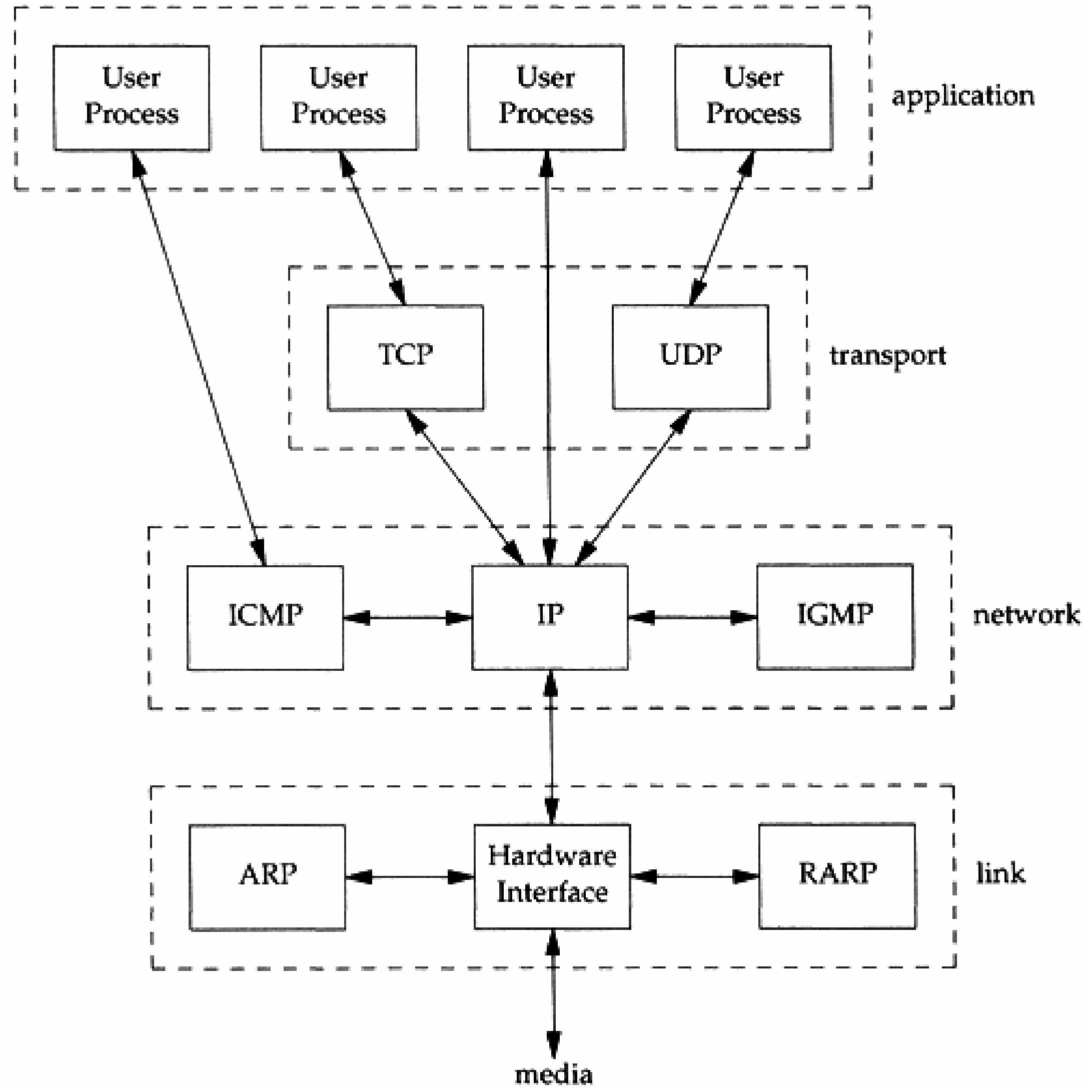
- **Link Layer (Data Link Layer)**
  - Network Interface Card + Driver
  - Handle all the hardware detail of whatever type of media
- **Network Layer (Internet Layer)**
  - Handle the movement of packets on the network
- **Transport Layer**
  - Provide end-to-end data delivery services
- **Application Layer**
  - Handle details of the particular application

Introduction (3)

- Various Address
  - **MAC Address**
    - Media Access Control Address
    - 48-bit Network Interface Card Hardware Address
      - 24bit manufacture ID
      - 24bit serial number
    - Ex:
      - 00:07:e9:10:e6:6b
  - **IP Address**
    - 32-bit Internet Address
    - Ex:
      - 140.113.209.64
  - **Port**
    - 16-bit uniquely identify application
    - Ex:
      - FTP port 21, ssh port 22, telnet port 23

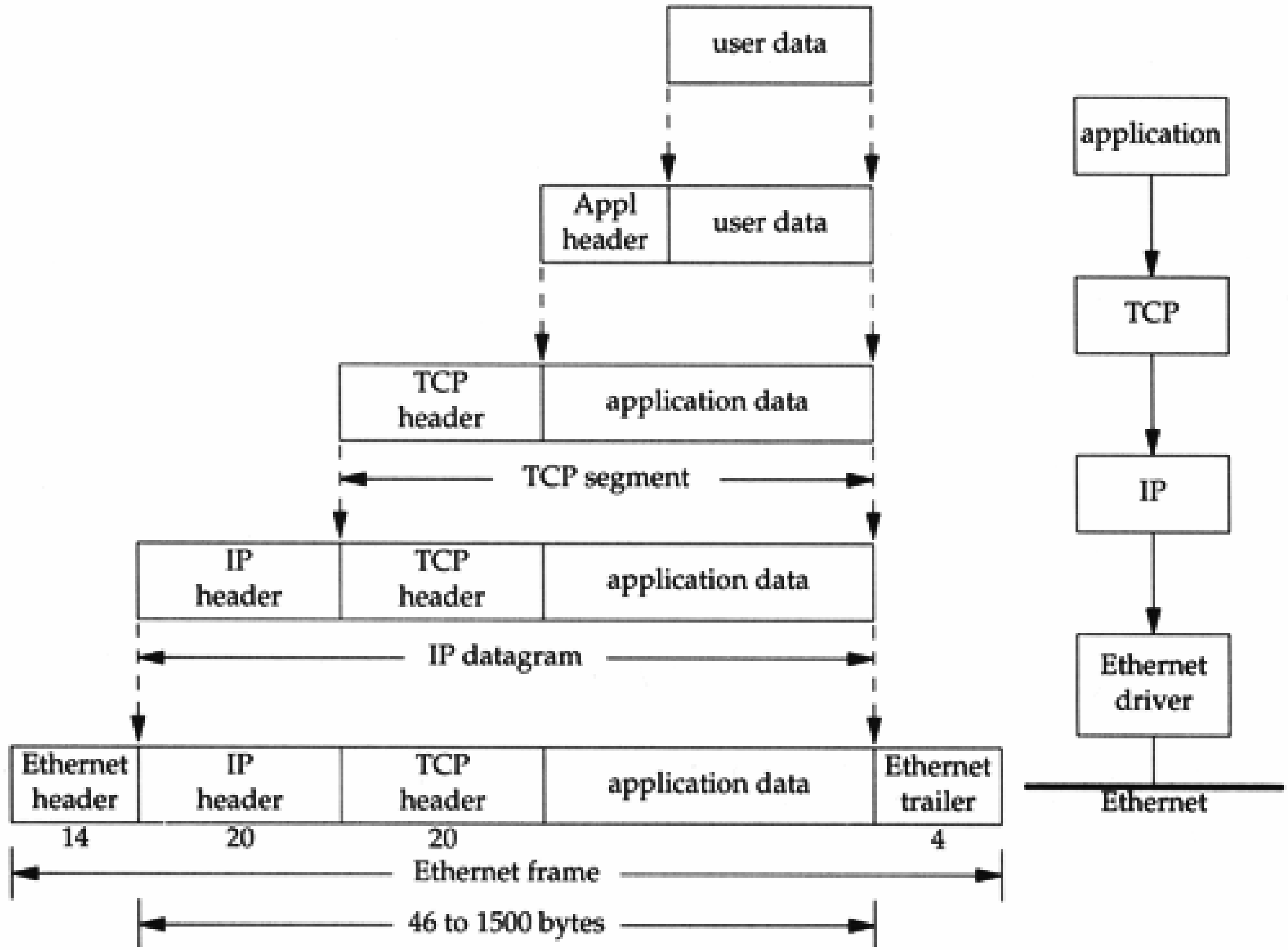
Introduction (4)

- Each layer has several protocols
  - A layer define a data communication function that may be performed by certain protocols
  - A protocol provides a service suitable to the function of that layer



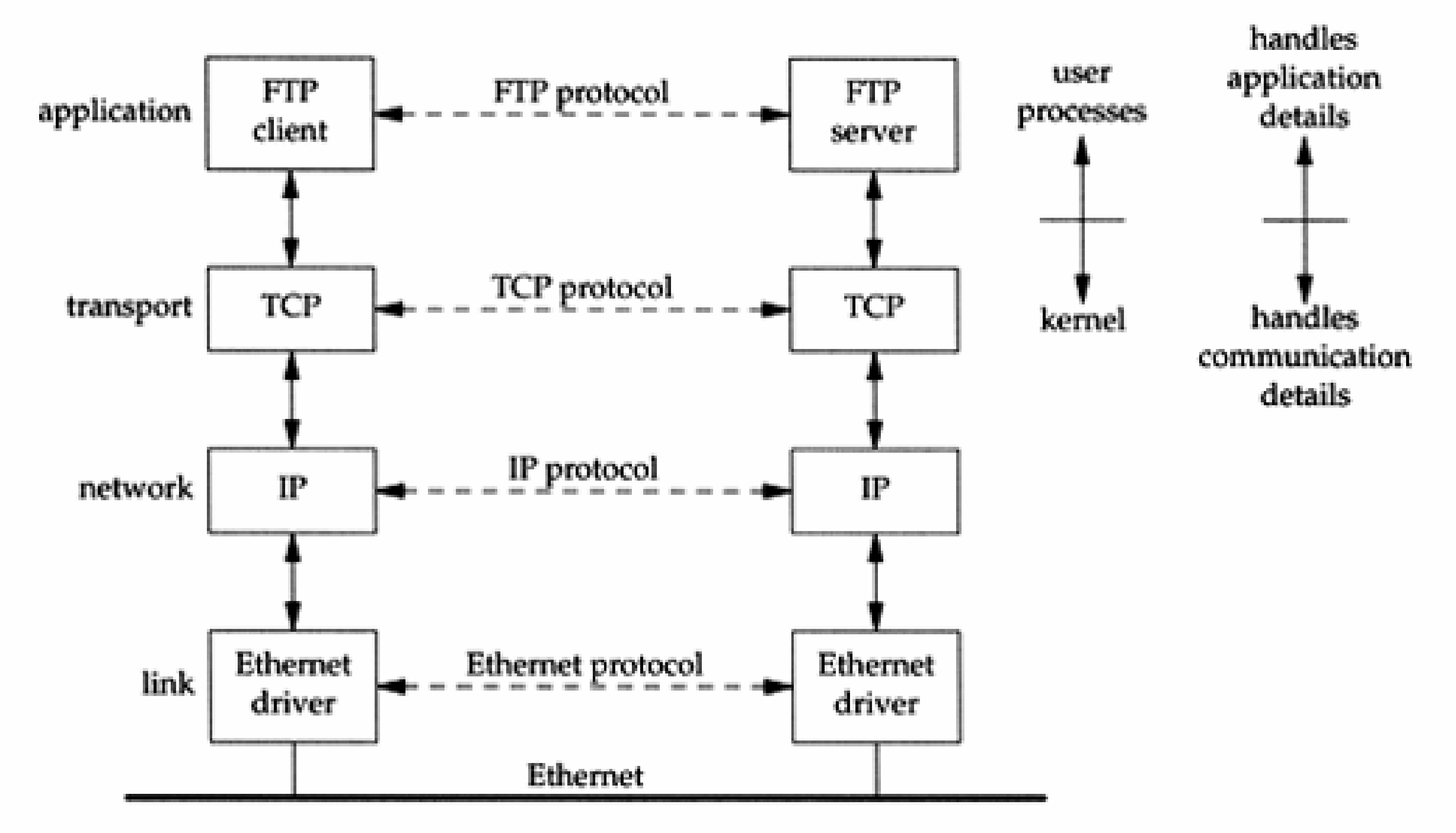
Introduction (5)

> Send data
  – **encapsulation**



Introduction (6)

> Addressing
  - **Nearby**
    (same network)

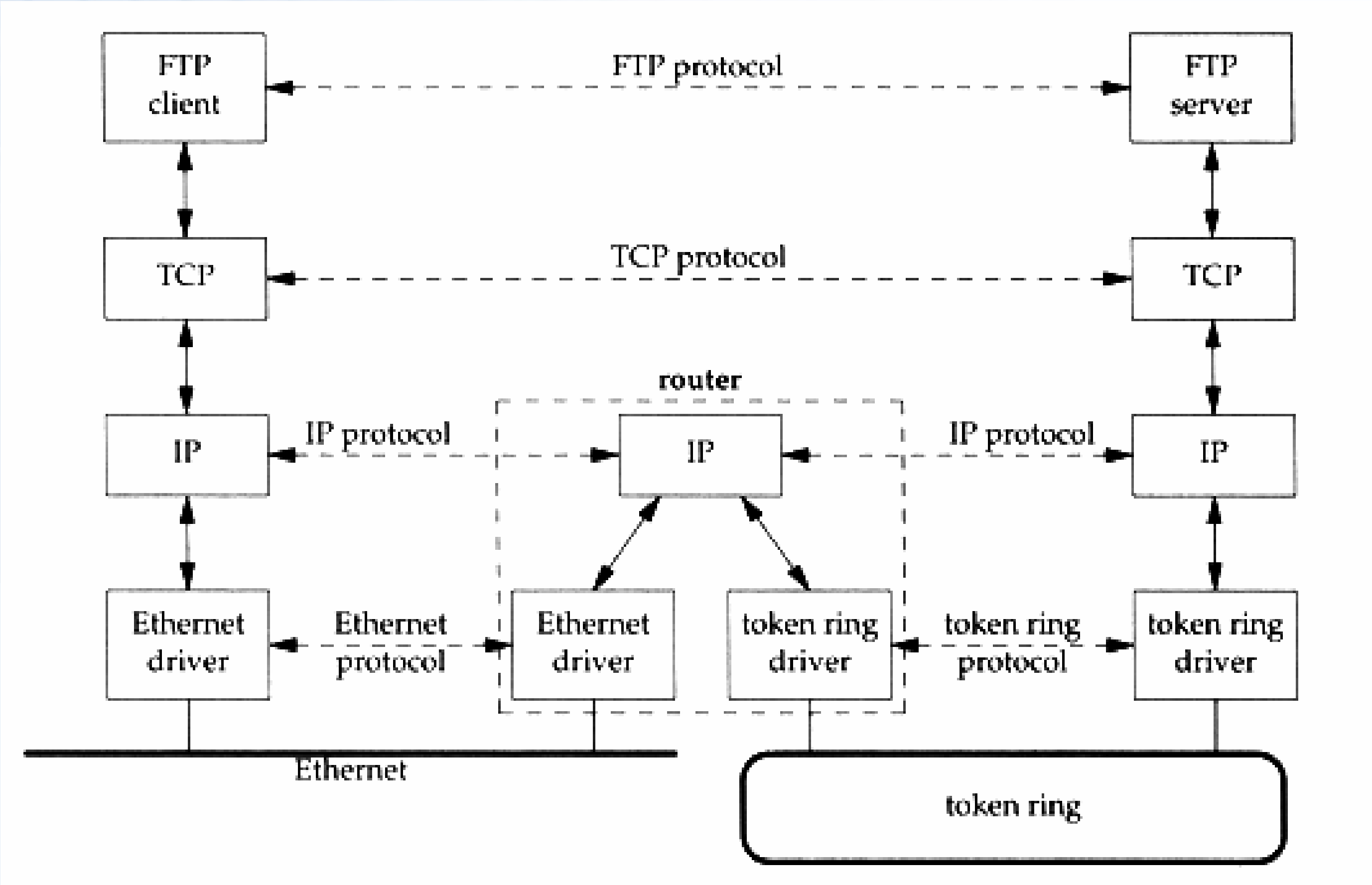


Introduction (7)

> Addressing

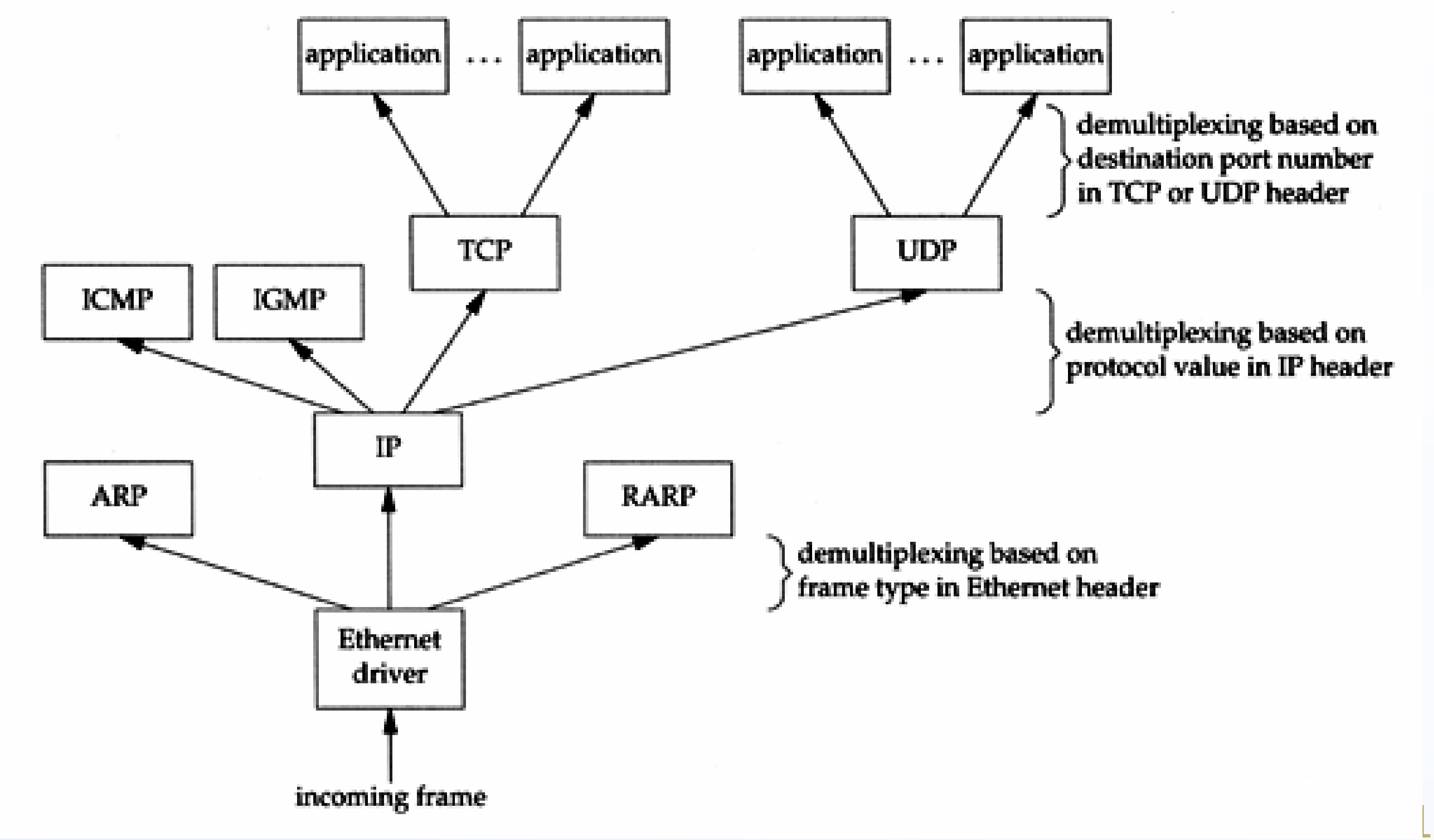
– **Faraway**

(across network)



Introduction (8)

> Receive Data
  – **Demultiplexing**



Link Layer

Introduction of Link Layer

> Purpose of the link layer
  - **Send and receive IP datagram for IP module**
  - **ARP request and reply**
  - **RARP request and reply**

> TCP/IP support various link layers, depending on the type of hardware used:
  - **Ethernet**
    - Teach in this class
  - **Token Ring**
  - **FDDI (Fiber Distributed Data Interface)**
  - **Serial Line**

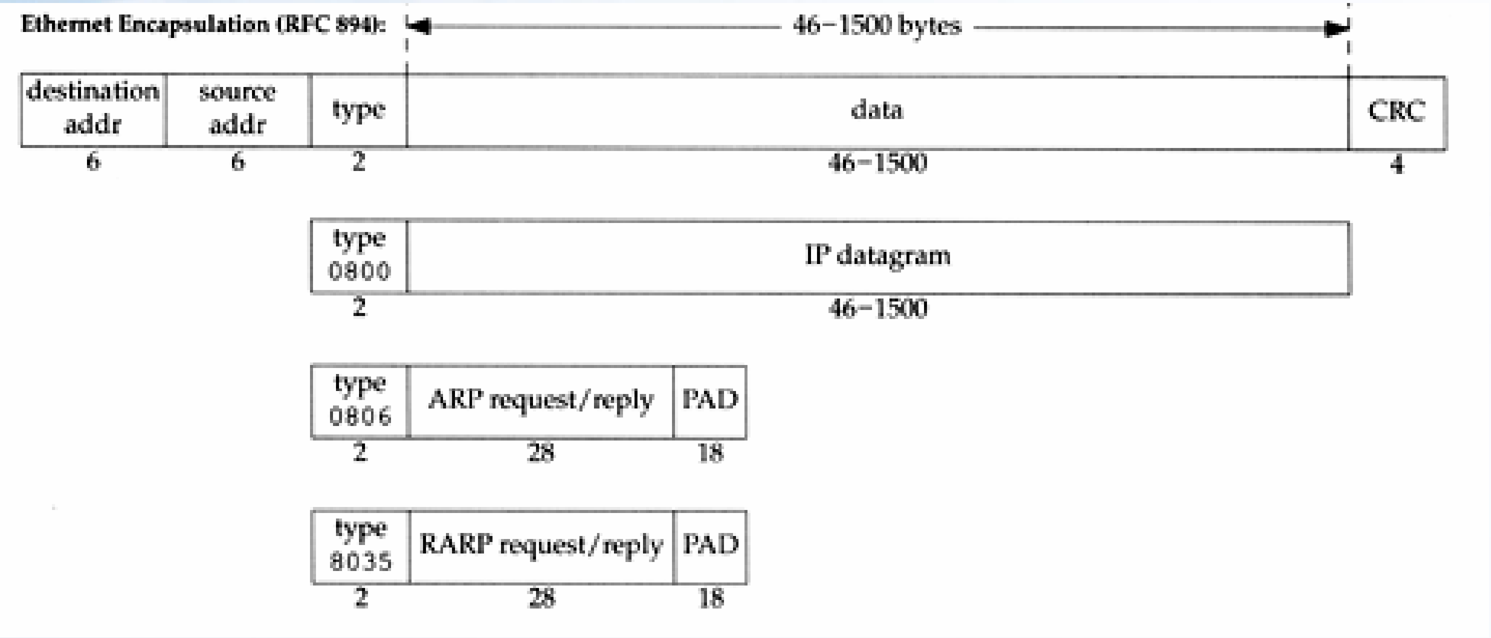
Ethernet

> Features

- Predominant form of local LAN technology used today
- Use CSMA/CD
  - Carrier Sense, Multiple Access with Collision Detection
- Use 48bit MAC address
- Operate at 10 Mbps
  - Fast Ethernet at 100 Mbps
- Ethernet frame format is defined in RFC894
  - This is the actually used format in reality

Ethernet Frame Format

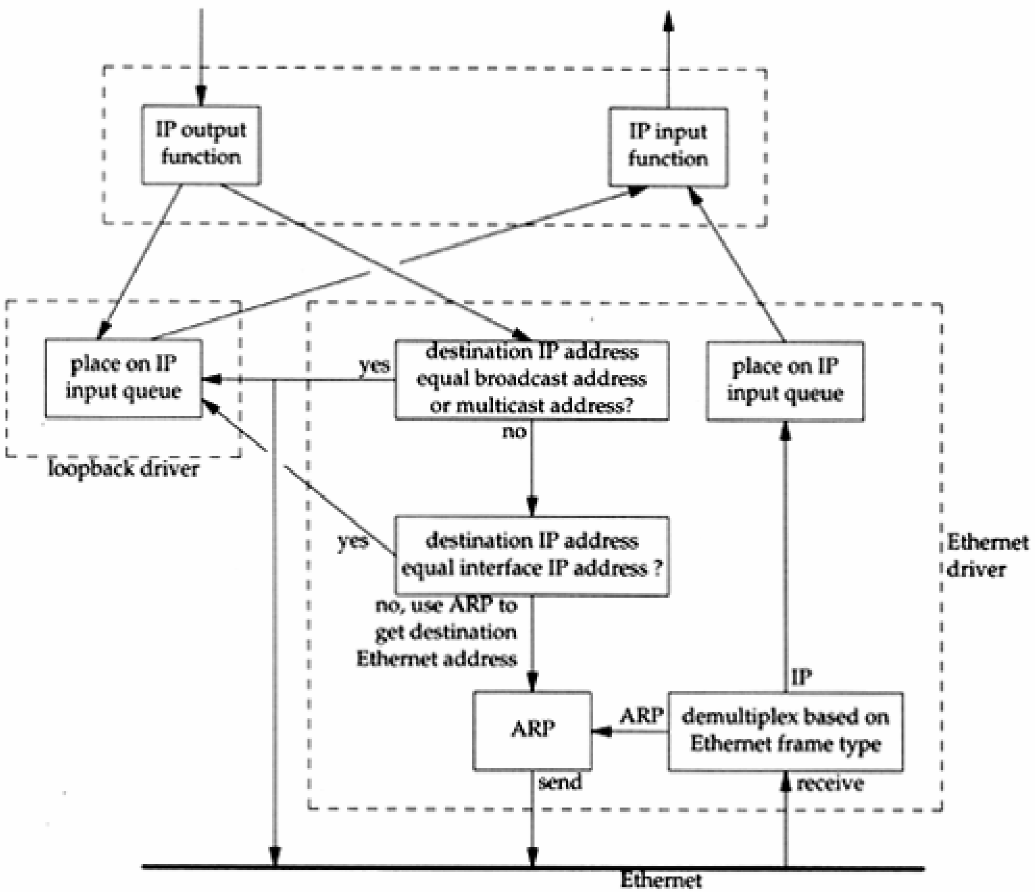
- 48bit hardware address
  - For both destination and source address
  - 16bit type is used to specify the type of following data
    - 0800 → IP datagram
    - 0806 → ARP, 8035 → RARP



Loopback Interface

> Pseudo NIC

- Allow client and server on the same host to communicate with each other using TCP/IP
- IP
  - 127.0.0.1
- Hostname
  - localhost



MTU

> Maximum Transmission Unit
  - **Limit size of payload part of Ethernet frame**
    - 1500 bytes
  - **If the IP datagram is larger than MTU,**
    - IP performs “fragmentation”

> MTU of various physical device

> Path MTU
  - **Smallest MTU of any data link MTU between the two hosts**
  - **Depend on route**

Network	MTU (bytes)
Hyperchannel	65535
16 Mbits/sec token ring (IBM)	17914
4 Mbits/sec token ring (IEEE 802.5)	4464
FDDI	4352
Ethernet	1500
IEEE 802.3/802.2	1492
X.25	576
Point-to-point (low delay)	296

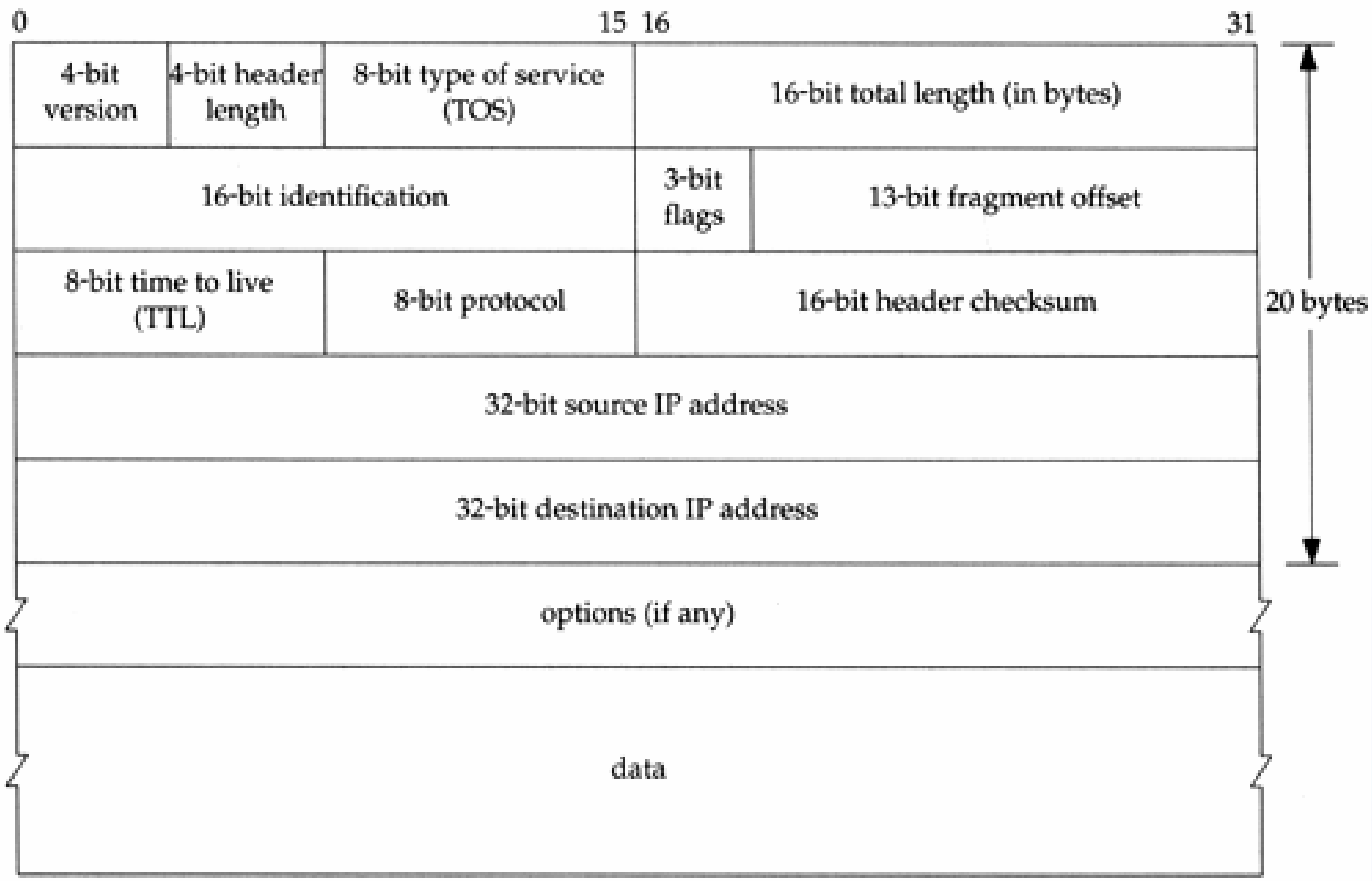
Network Layer

Introduction to Network Layer

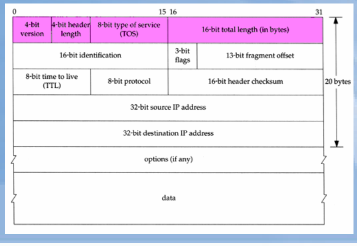
- Unreliable, connectionless datagram delivery service
  - **IP Routing**
  - **IP provides best effort service (unreliable)**
  - **IP datagram can be delivered out of order (connectionless)**
- Protocols using IP
  - **TCP, UDP, ICMP, IGMP**

IP Header (1)

> 20 bytes in total length, excepts options



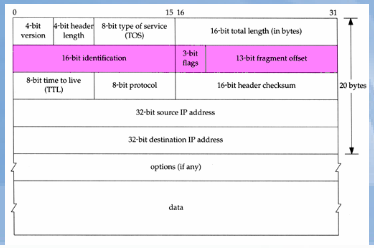
IP Header (2)



- Version (4bit)
  - 4 for IPv4 and 6 for IPv6
- Header length (4bit)
  - The number of 32bit words in the header (15*4=60bytes)
  - Normally, the value is 5 (no option)
- TOS-Type of Service (8bit)
  - 3bit precedence + 4bit TOS + 1bit unused
- Total length (16bit)
  - Total length of the IP datagram in bytes

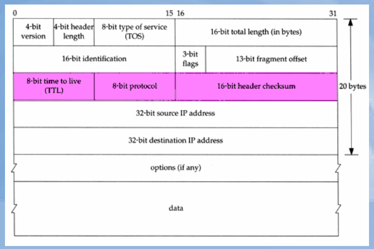
Application	Minimize delay	Maximize throughput	Maximize reliability	Minimize monetary cost	Hex value
Telnet/Rlogin	1	0	0	0	0x10
FTP					
control	1	0	0	0	0x10
data	0	1	0	0	0x08
any bulk data	0	1	0	0	0x08
TFTP	1	0	0	0	0x10
SMTP					
command phase	1	0	0	0	0x10
data phase	0	1	0	0	0x08

IP Header (3)



- Identification (16bit)
- Fragmentation offset (13bit)
- Flags (3bit)
  - **All these three fields are used for fragmentation**

IP Header (4)



> TTL (8bit)
  - Limit of next hop count of routers

> Protocol (8bit)
  - Used to demultiplex to other protocols
  - TCP, UDP, ICMP, IGMP

> Header checksum (16bit)
  - Calculated over the IP header only
  - If checksum error, IP discards the datagram and no error message is generated

IP Routing (1)

> Difference between Host and Router

- Router forwards datagram from one of its interface to another, while host does not
- Almost every Unix system can be configured to act as a router or both

> Router

- IP layer has a routing table, which is used to store the information for forwarding datagram
- When router receiving a datagram
  - If Dst. IP = my IP, demultiplex to other protocol
  - Other, forward the IP based on routing table

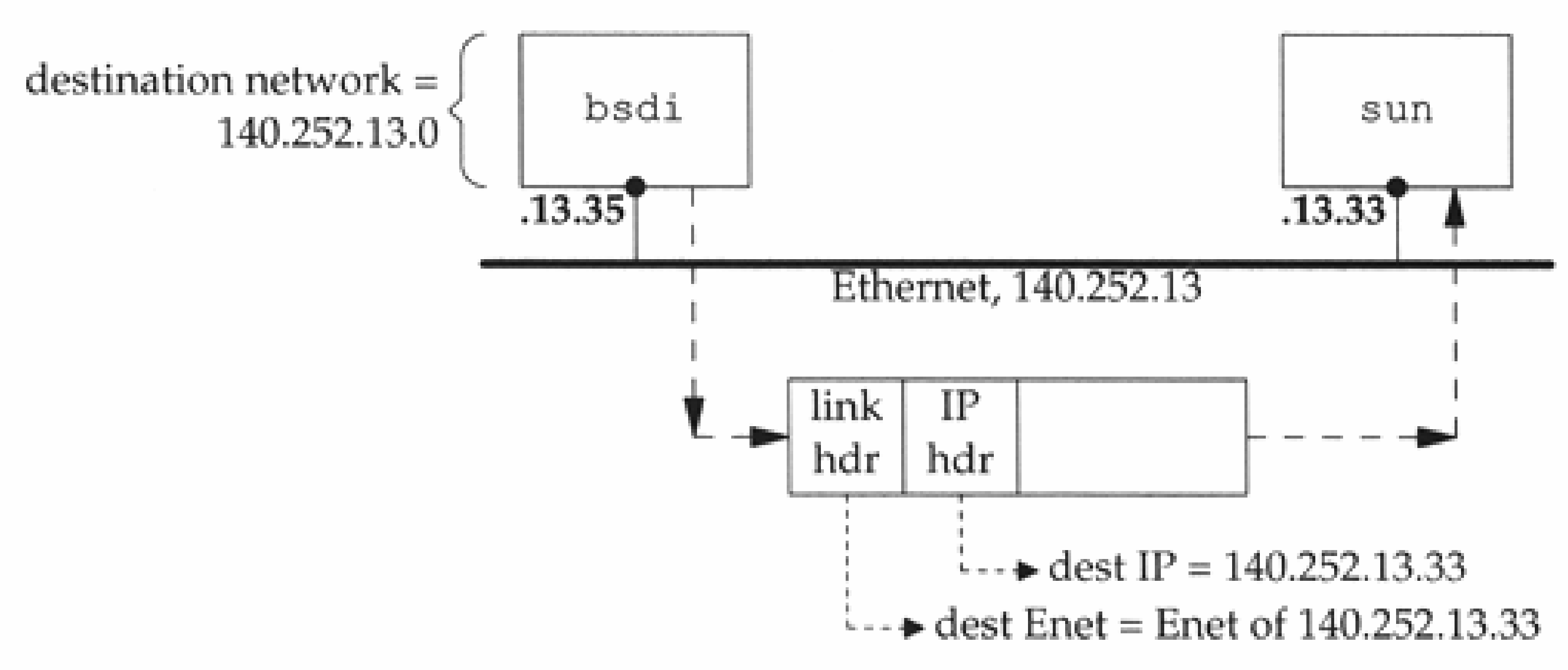
IP Routing (2)

- Routing table information
  - **Destination IP**
  - **IP address of next-hop router or IP address of a directly connected network**
  - **Flags**
  - **Next interface**
- IP routing
  - **Done on a hop-by-hop basis**
  - **It assumes that the next-hop router is closer to the destination**
  - **Steps:**
    - Search routing table for complete matched IP address
      - Send to next-hop router or to the directly connected NIC
    - Search routing table for matched network ID
      - Send to next-hop router or to the directly connected NIC
    - Search routing table for default route
      - Send to this default next-hop router
    - host or network unreachable

IP Routing (3)

> Ex1: routing in the same network

– **bsdi: 140.252.13.35**
– **sun: 140.252.13.33**

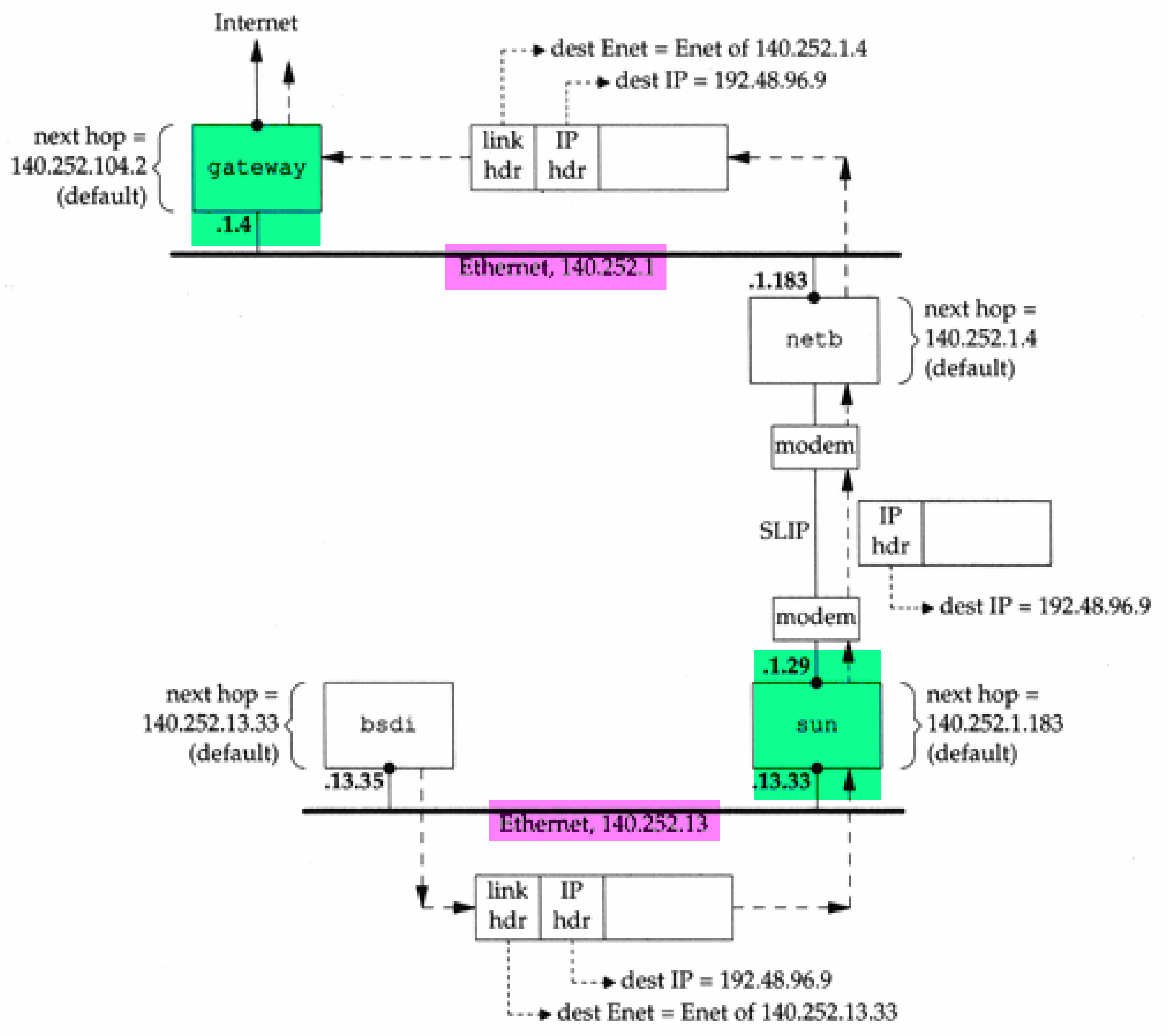


Ex Routing table:

140.252.13.33 00:d0:59:83:d9:16 UHLW fxp1

IP Routing (4)

> Ex2:
  - routing across multi-network



IP Address (1)

> 32-bit long
  - **Network part**
    - Identify a logical network
  - **Host part**
    - Identify a machine on certain network
> IP address category

Class	1st byte[a]	Format	Comments
A	1-126	N.H.H.H	Very early networks, or reserved for DOD
B	128-191	N.N.H.H	Large sites, usually subnetted, were hard to get
C	192-223	N.N.N.H	Easy to get, often obtained in sets
D	224-239	–	Multicast addresses, not permanently assigned
E	240-254	–	Experimental addresses

a. The values 0 and 255 are special and are not used as the first byte of regular IP addresses. 127 is reserved for the loopback address.

IP Address (2)

> Ex:
  - **NCTU**
    - Class B address: 140.113.0.0
    - Network ID: 140.113
    - Number of hosts: 255*255 = 65535

> Problems of Class A or B network
  - **Number of hosts is enormous**
  - **Hard to maintain and management**
  - **Solution → subnetting**

subnetting and netmask (1)

- Subnetting
  - Borrow some bits from network ID to extends hosts ID
  - Ex:
    - ClassB address : 140.113.0.0
      = 256 ClassC-like IP addresses in N.N.N.H subnetting method
    - 140.113.209.0 subnet
- netmask
  - Specify how many bits of network-ID are used for network-ID
  - Continuous 1 bits form the network part
  - Ex:
    - 255.255.255.0 in NCTU-CSIE example
      - 256 hosts available
    - 255.255.255.248 in ADSL example
      - Only 8 hosts available
  - Shorthand notation
    - Address/prefix-length
      - Ex: 140.113.209.8/24

subnetting and netmask (2)

> How to determine your network ID?

- Bit-wise-and IP and netmask
- Ex:
  - 140.113.214.37 & 255.255.255.0 → 140.113.214.0
  - 140.113.209.37 & 255.255.255.0 → 140.113.209.0

  - 140.113.214.37 & 255.255.0.0 → 140.113.0.0
  - 140.113.209.37 & 255.255.0.0 → 140.113.0.0

  - 211.23.188.78 & 255.255.255.248 → 211.23.188.76
    - 78 = 01001110
    - 78 & 248= 01001110 & 11111000 =72

subnetting and netmask (3)

> In a subnet, not all IP are usable
  - **The first one IP → network ID**
  - **The last one IP → broadcast address**
  - **Ex:**
    - Netmask 255.255.255.0
    - 140.113.209.32/24
    - 140.113.209.0 → network ID
    - 140.113.209.255 → broadcast address
    - 1 ~ 254, total 254 IPs are usable
  - **Ex:**
    - Netmask 255.255.255.252
    - 211.23.188.78/29
    - 211.23.188.72 → network ID
    - 211.23.188.79 → broadcast address
    - 73 ~ 78, total 6 IPs are usable

subnetting and netmask (4)

- > The smallest subnetting
  - Network portion : 30 bits
  - Host portion : 2 bits
  - ➔ **4 hosts, but only 2 IPs are available**
- > ipcalc.pl

```
[shrang@r21607 ~]$ ./ipcalc 211.23.188.78/29
IP address        211    .    23    .    188    .    78     / 29   211.23.188.78/29
Netmask bits     11111111 11111111 11111111 11111000
Netmask bytes     255    .   255    .   255    .   248            255.255.255.248
Address bits     11010011 00010111 10111100 01001110
Network           211    .    23    .    188    .    72            211.23.188.72
Broadcast         211    .    23    .    188    .    79            211.23.188.79
First Host        211    .    23    .    188    .    73            211.23.188.73
Last Host         211    .    23    .    188    .    78            211.23.188.78
Total Hosts      6
PTR              78.188.23.211.in-addr.arpa
IP Address (hex) D317BC4E
[shrang@r21607 ~]$
```

subnetting and netmask (5)

> Benefits of subnet

- **Reduce the routing table size of Internet’s routers**
- **Ex:**
  - All external routers have only one entry for 140.113 Class B network

> Problems of Class C network

- **255*255*255 number of Class C network make the size of Internet routes huge**
- **Solution**
  - Classless Inter-Domain Routing

CIDR

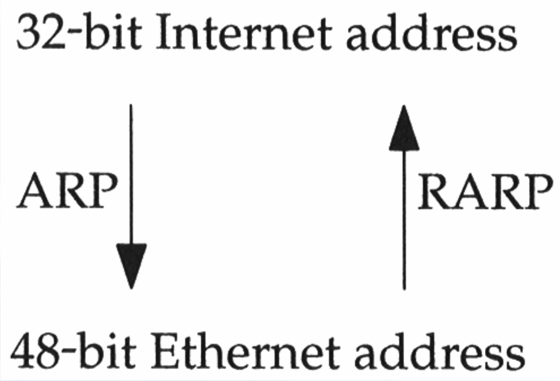
> Classless Inter-Domain Routing
  - **Use address mask instead of old address classes to determine the destination network**
  - **CIDR requires modifications to routers and routing protocols**
    - Need to transmit both destination address and mask

> Benefit of CIDR
  - **We can allocate continuous ClassC network to organization**
    - Reflect physical network topology
    - Reduce the size of routing table

ARP – Address Resolution Protocol and RARP – Reverse ARP

ARP and RARP

- Mapping between IP and Ethernet address



- When an Ethernet frame is sent on LAN from one host to another,
  - **It is the 48bit Ethernet address that determines for which interface the frame is destined**

ARP Example

% ftp ccbsd5

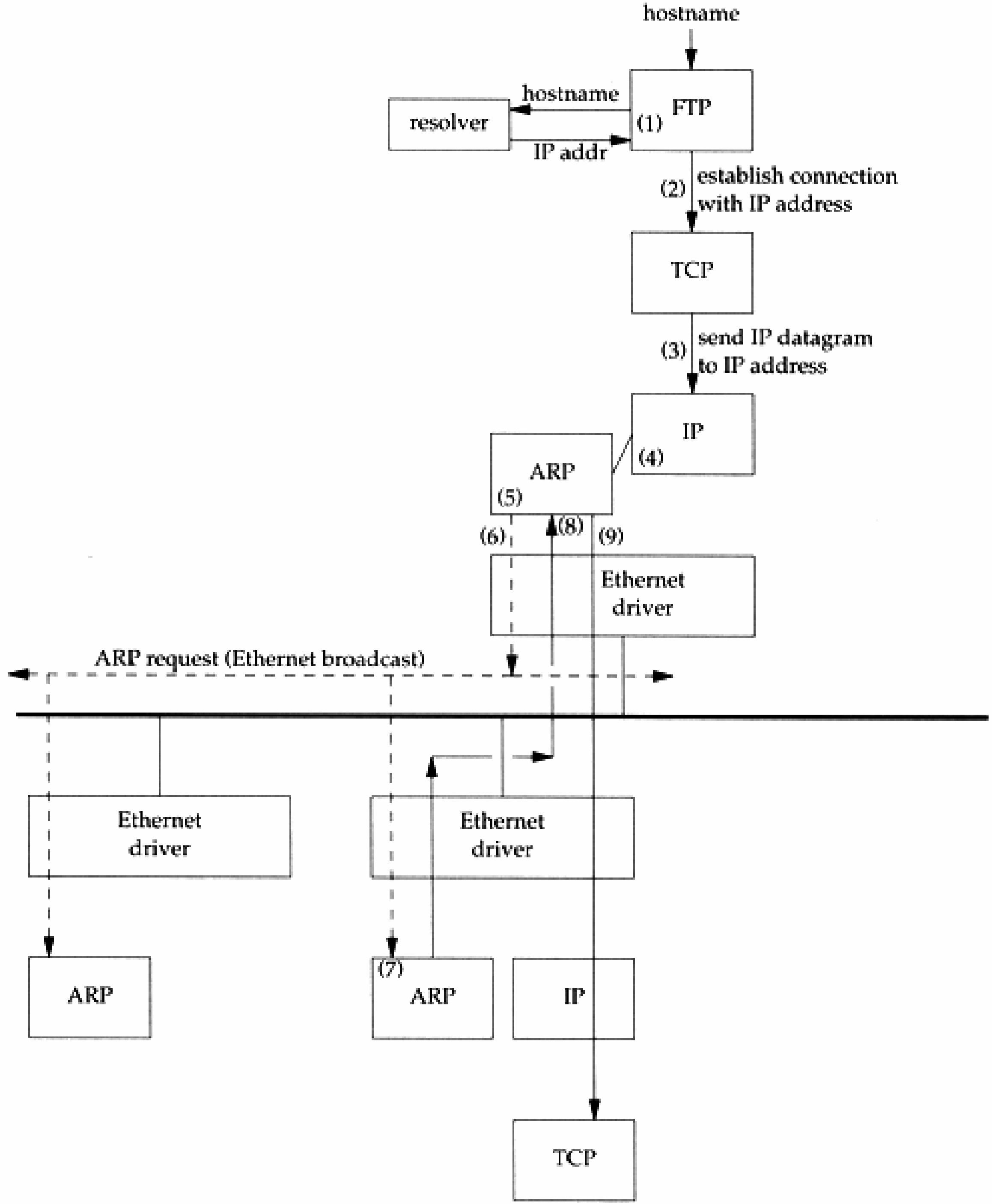
(4) next-hop or direct host

(5) Search ARP cache

(6) Broadcast ARP request

(7) ccbsd5 response ARP reply

(9) Send original IP datagram

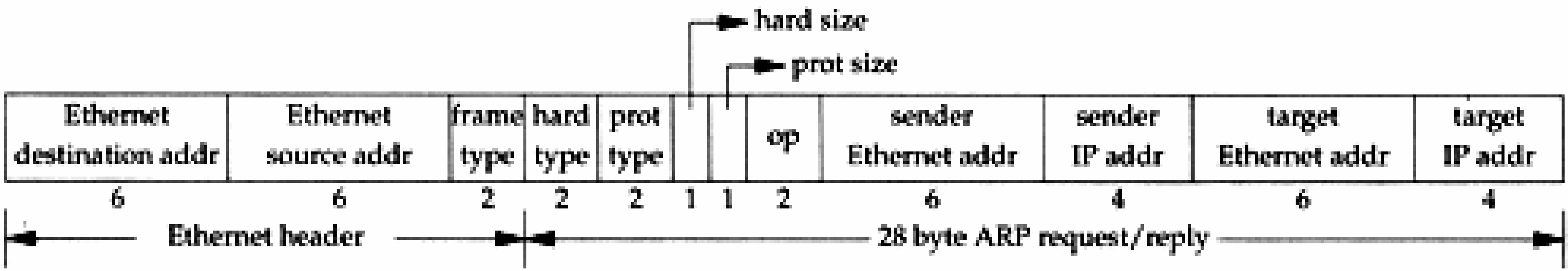


ARP Cache

- Maintain recent ARP results
  - come from both ARP request and reply
  - expiration time
    - Complete entry = 20 minutes
    - Incomplete entry = 3 minutes
  - Use arp command to see the cache
  - Ex:
    - % arp –a
    - % arp –da
    - % arp –S ccbsd5.csie.nctu.edu.tw 00:07:e9:39:66:77

```
tytsai@tybsd:~> arp -a
ccamd.csie.nctu.edu.tw (140.113.235.1) at 00:0f:ea:48:92:85 on fxp0 [ethernet]
tybsd.csie.nctu.edu.tw (140.113.235.4) at 00:09:6b:7a:25:f7 on fxp0 permanent [ethernet]
e3rtn-235.csie.nctu.edu.tw (140.113.235.254) at 00:0e:38:a4:c2:00 on fxp0 [ethernet]
? (192.168.1.30) at (incomplete) on fxp1 [ethernet]
```

ARP/RARP Packet Format



> Ethernet destination addr: all 1's (broadcast)

> Known value for IP <-> Ethernet

- **Frame type: 0x0806 for ARP, 0x8035 for RARP**
- **Hardware type: type of hardware address (1 for Ethernet)**
- **Protocol type: type of upper layer address (0x0800 for IP)**
- **Hard size: size in bytes of hardware address (6 for Ethernet)**
- **Protocol size: size in bytes of upper layer address (4 for IP)**
- **Op: 1, 2, 3, 4 for ARP request, reply, RARP request, reply**

Ex: Use tcpdump to see ARP

> Host 140.113.214.22 → 140.113.214.49
  - Clear ARP cache of 140.113.214.22 (0:20:ed:6d:eb:c)
  - Run tcpdump on 140.113.214.49 (0:2:a5:6e:8d:4)
    - % sudo tcpdump –i fxp0 –e arp
    - % sudo tcpdump –i fxp0 –n –e arp
    - % sudo tcpdump –i fxp0 –n –t –e arp
  - On 140.113.214.22, ssh to 140.113.214.49

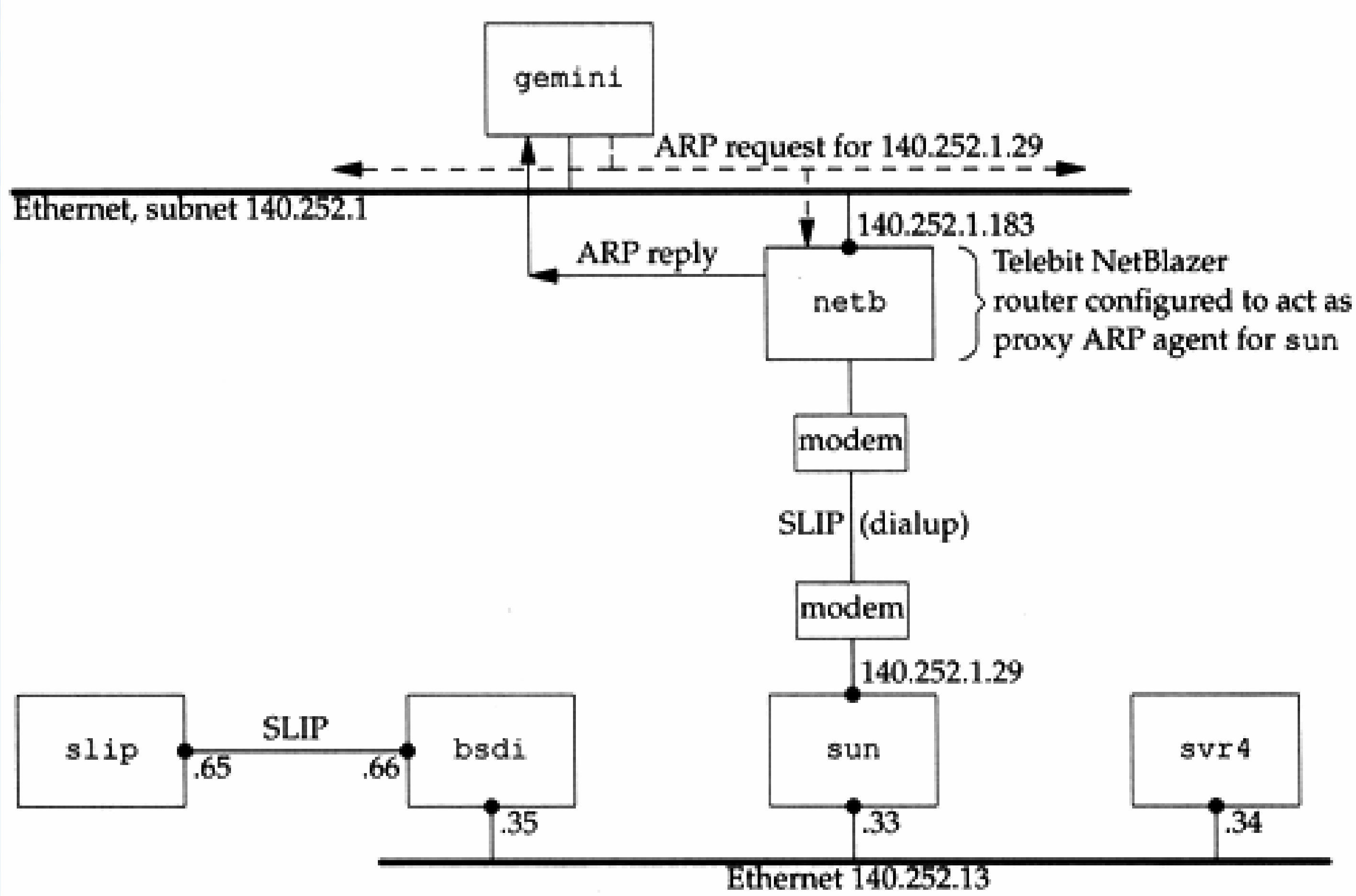
```
23:35:04.971913 0:20:ed:6d:eb:c Broadcast arp 60: arp who-has r21619.csie.nctu.edu.tw tel
l u214.csie.nctu.edu.tw
23:35:04.971921 0:2:a5:6e:8d:42 0:20:ed:6d:eb:c arp 60: arp reply r21619.csie.nctu.edu.tw
 is-at 0:2:a5:6e:8d:42
```

```
23:44:12.407720 0:20:ed:6d:eb:c ff:ff:ff:ff:ff:ff 0806 60: arp who-has 140.113.214.49 tel
l 140.113.214.22
23:44:12.407730 0:2:a5:6e:8d:42 0:20:ed:6d:eb:c 0806 60: arp reply 140.113.214.49 is-at 0
:2:a5:6e:8d:42
```

```
0:20:ed:6d:eb:c ff:ff:ff:ff:ff:ff 0806 60: arp who-has 140.113.214.49 tell 140.113.214.22
0:2:a5:6e:8d:42 0:20:ed:6d:eb:c 0806 60: arp reply 140.113.214.49 is-at 0:2:a5:6e:8d:42
```

Proxy ARP

- Let router answer ARP request on one of its networks for a host on another of its network



Gratuitous ARP

> Gratuitous ARP
  - **The host sends an ARP request looking for its own IP**
  - **Provide two features**
    - Used to determine whether there is another host configured with the same IP
    - Used to cause any other host to update ARP cache when changing hardware address

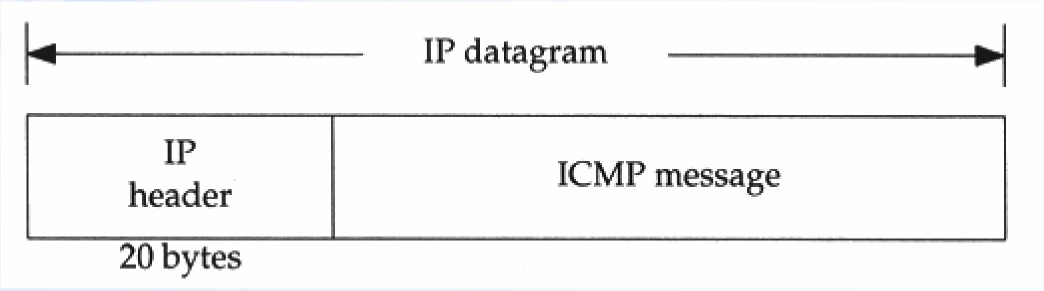
RARP

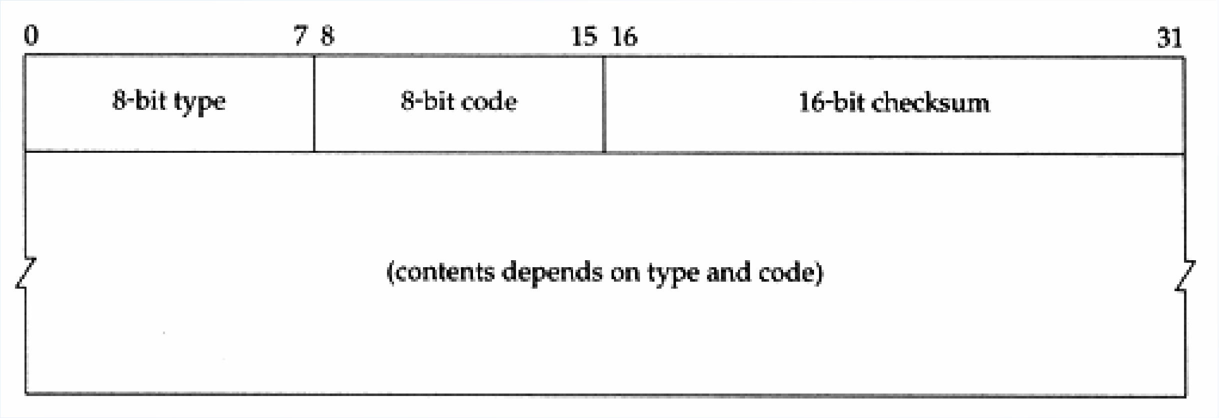
- Principle
  - Used for the diskless system to read its hardware address from the NIC and send an RARP request to gain its IP
- RARP Server Design
  - RARP server must maintain the map from hardware address to an IP address for many host
  - Link-layer broadcast
    - This prevent most routers from forwarding an RARP request

ICMP –
Internet Control Message Protocol

ICMP Introduction

- Part of the IP layer
  - ICMP messages are transmitted within IP datagram
  - ICMP communicates error messages and other conditions that require attention for other protocols
- ICMP message format





ICMP Message Type (1)

type	*code*	Description	Query	Error
0	0	echo reply (Ping reply, Chapter 7)	•	
3		destination unreachable:		
	0	network unreachable (Section 9.3)		•
	1	host unreachable (Section 9.3)		•
	2	protocol unreachable		•
	3	port unreachable (Section 6.5)		•
	4	fragmentation needed but don't-fragment bit set (Section 11.6)		•
	5	source route failed (Section 8.5)		•
	6	destination network unknown		•
	7	destination host unknown		•
	8	source host isolated (obsolete)		•
	9	destination network administratively prohibited		•
	10	destination host administratively prohibited		•
	11	network unreachable for TOS (Section 9.3)		•
	12	host unreachable for TOS (Section 9.3)		•
	13	communication administratively prohibited by filtering		•
	14	host precedence violation		•
	15	precedence cutoff in effect		•
4	0	source quench (elementary flow control, Section 11.11)		•

ICMP Message Type (2)

5		redirect (Section 9.5):		
	0	redirect for network		•
	1	redirect for host		•
	2	redirect for type-of-service and network		•
	3	redirect for type-of-service and host		•
8	0	echo request (Ping request, Chapter 7)	•	
9	0	router advertisement (Section 9.6)	•	
10	0	router solicitation (Section 9.6)	•	
11		time exceeded:		
	0	time-to-live equals 0 during transit (Traceroute, Chapter 8)		•
	1	time-to-live equals 0 during reassembly (Section 11.5)		•
12		parameter problem:		
	0	IP header bad (catchall error)		•
	1	required option missing		•
13	0	timestamp request (Section 6.4)	•	
14	0	timestamp reply (Section 6.4)	•	
15	0	information request (obsolete)	•	
16	0	information reply (obsolete)	•	
17	0	address mask request (Section 6.3)	•	
18	0	address mask reply (Section 6.3)	•	

ICMP Query Message - Address Mask Request/Reply (1)

> Address Mask Request and Reply

- **Used for diskless system to obtain its subnet mask**
- **Identifier and sequence number**
  - Can be set to anything for sender to match reply with request
- **The receiver will response an ICMP reply with the subnet mask of the receiving NIC**

0	8	16	31
TYPE (17 or 18)	CODE (0)	CHECKSUM	
IDENTIFIER		SEQUENCE NUMBER	
ADDRESS MASK			

ICMP Query Message - Address Mask Request/Reply (2)

> Ex:

```
tytsai@tybsd:~/<1>tcpipi/icmpaddrmask> ping -M mask ccsun1
ICMP_MASKREQ
PING ccsun1.csie.nctu.edu.tw (140.113.209.101): 56 data bytes
68 bytes from 140.113.209.101: icmp_seq=0 ttl=254 time=0.389 ms
mask=not-a-legal-address (255.255.255.0)
68 bytes from 140.113.209.101: icmp_seq=1 ttl=254 time=0.363 ms
mask=not-a-legal-address (255.255.255.0)
^C
--- ccsun1.csie.nctu.edu.tw ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.363/0.376/0.389/0.013 ms
tytsai@tybsd:~/<1>
tytsai@tybsd:~/<1>tcpipi/icmpaddrmask> sudo ./icmpaddrmask ccsun1.csie.nctu.edu.tw
received mask = ffffff00, from 140.113.209.101
```

ICMP Query Message - Timestamp Request/Reply (1)

> Timestamp request and reply

- **Allow a system to query another for the current time**
- **Milliseconds resolution, since midnight UTC**
- **Requestor**
  - Fill in the originate timestamp and send
- **Reply system**
  - Fill in the receive timestamp when it receives the request and the transmit time when it sends the reply

0	8	16	31
TYPE (13 or 14)	CODE (0)	CHECKSUM	
IDENTIFIER		SEQUENCE NUMBER	
ORIGINATE TIMESTAMP			
RECEIVE TIMESTAMP			
TRANSMIT TIMESTAMP			

ICMP Query Message - Timestamp Request/Reply (2)

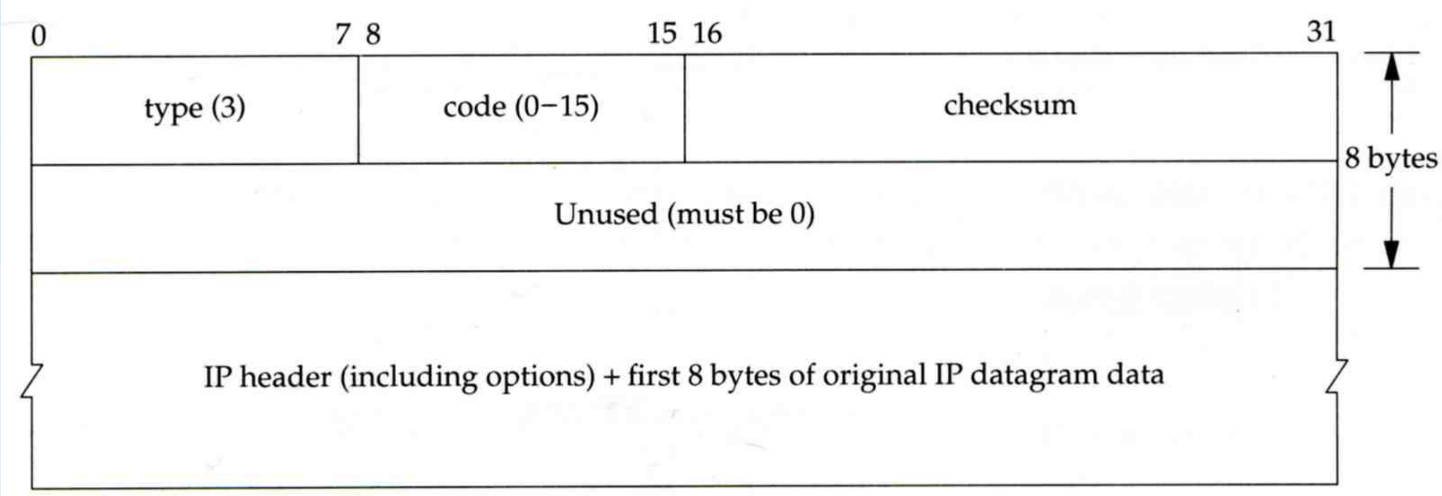
> Ex:

```
tytsai@tybsd:~/<1>tcpipi/icmptime> sudo ./icmptime tybsd.csie.nctu.edu.tw
orig = 38167109, recv = 38167102
adjustment = -7 ms
correction = 0 sec, -7000 usec
tytsai@tybsd:~> sudo tcpdump -i lo0 -e icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo0, link-type NULL (BSD loopback), capture size 96 bytes
10:38:43.547811 ip 40: IP tybsd.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 20:
time stamp query id 56635 seq 14640
10:38:43.547842 ip 40: IP tybsd.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 20:
time stamp reply id 56635 seq 14640 : org 0x248c55b recv 0x248c556 xmit 0x248c556
10:38:55.961538 ip 96: IP tybsd.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 76:
time stamp query id 57147 seq 0
10:38:55.961569 ip 96: IP tybsd.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 76:
time stamp reply id 57147 seq 0 : org 0x248f5d9 recv 0x248f5d5 xmit 0x248f5d5
```

```
tytsai@tybsd:~/<1>tcpipi/icmptime> ping -M time tybsd.csie.nctu.edu.tw
ICMP_TSTAMP
PING tybsd.csie.nctu.edu.tw (140.113.235.4): 56 data bytes
76 bytes from 140.113.235.4: icmp_seq=0 ttl=64 time=0.086 ms
tso=10:38:55 tsr=10:38:55 tst=10:38:55
76 bytes from 140.113.235.4: icmp_seq=1 ttl=64 time=0.066 ms
tso=10:38:56 tsr=10:38:56 tst=10:38:56
```

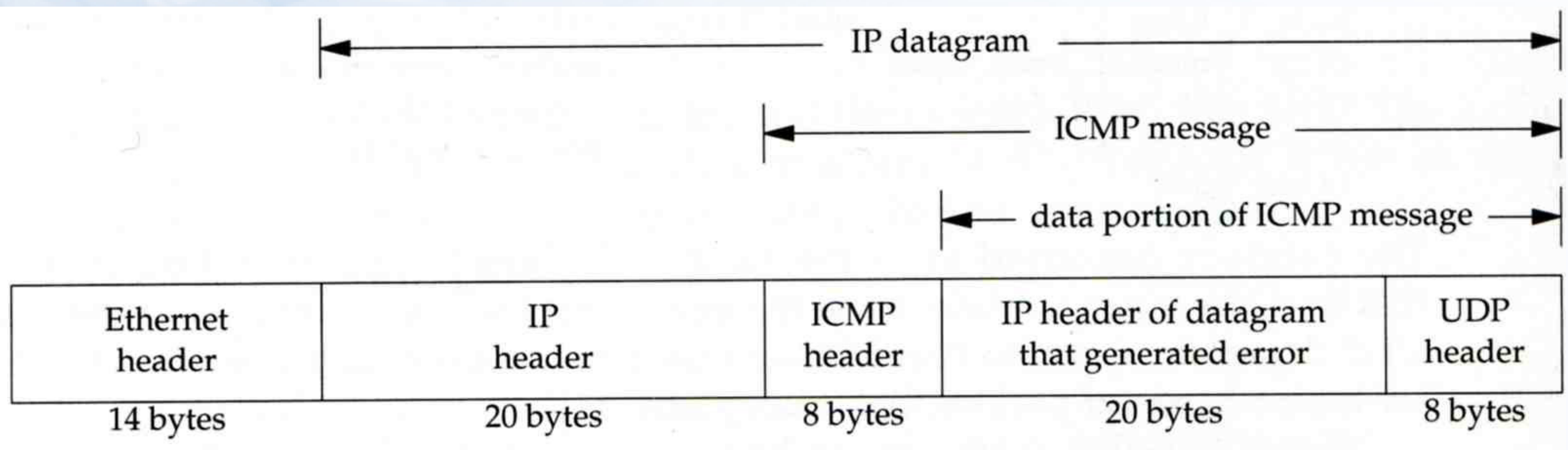
ICMP Unreachable Error Message

> Format
  - **8bytes ICMP Header**
  - **Application-depend data portion**
    - IP header
      > Let ICMP know how to interpret the 8 bytes that follow
    - first 8bytes that followed this IP header
      > Information about who generates the error



ICMP Error Message – Port Unreachable (1)

> ICMP port unreachable
  - **Type = 3 , code = 3**
  - **Host receives a UDP datagram but the destination port does not correspond to a port that some process has in use**



ICMP Error Message – Port Unreachable (2)

> Ex:

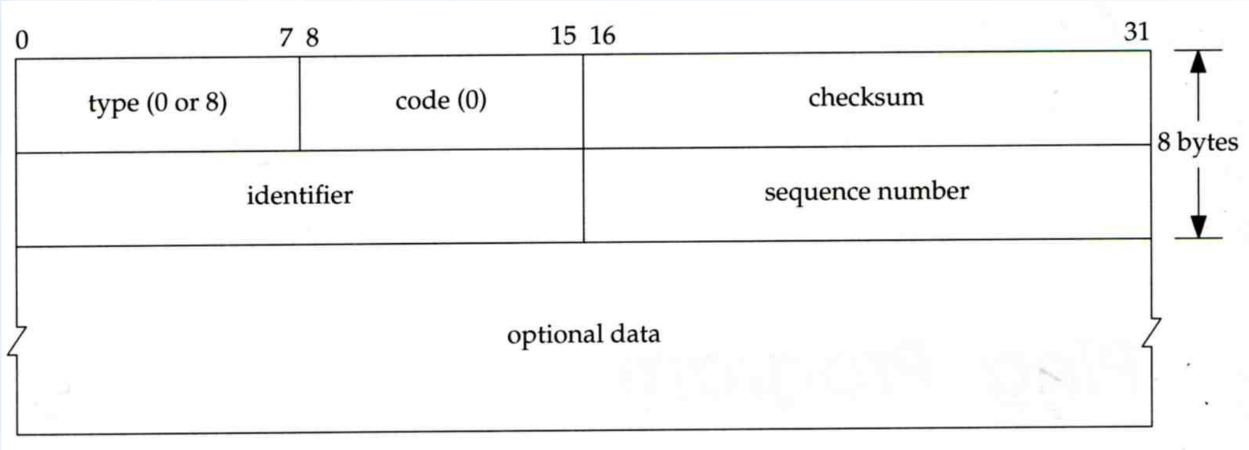
– **Using TFTP (Trivial File Transfer Protocol)**
  - Original port: 69

```
tytsai@tybsd:~> tftp
tftp> connect localhost 8888
tftp> get temp.foo
Transfer timed out.

tftp>
```

```
tytsai@tybsd:~> sudo tcpdump -i lo0 -e
Password:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo0, link-type NULL (BSD loopback), capture size 96 bytes

11:55:38.482768 ip 48: IP localhost.53850 > localhost.8888: UDP, length: 20
11:55:38.482790 ip 56: IP localhost > localhost: icmp 36: localhost udp port 8888 unreachable
11:55:43.488100 ip 48: IP localhost.53850 > localhost.8888: UDP, length: 20
11:55:43.488123 ip 56: IP localhost > localhost: icmp 36: localhost udp port 8888 unreachable
```

Ping Program (1)

- Use ICMP to test whether another host is reachable
  - Type 8, ICMP echo request
  - Type 0, ICMP echo reply
- ICMP echo request/reply format
  - Identifier: process ID of the sending process
  - Sequence number: start with 0
  - Optional data: any optional data sent must be echoed



Ping Program (2)

> Ex:

- 由 tybsd.csie.nctu.edu.tw 對自己 ping
- % tcpdump –i lo0 -x

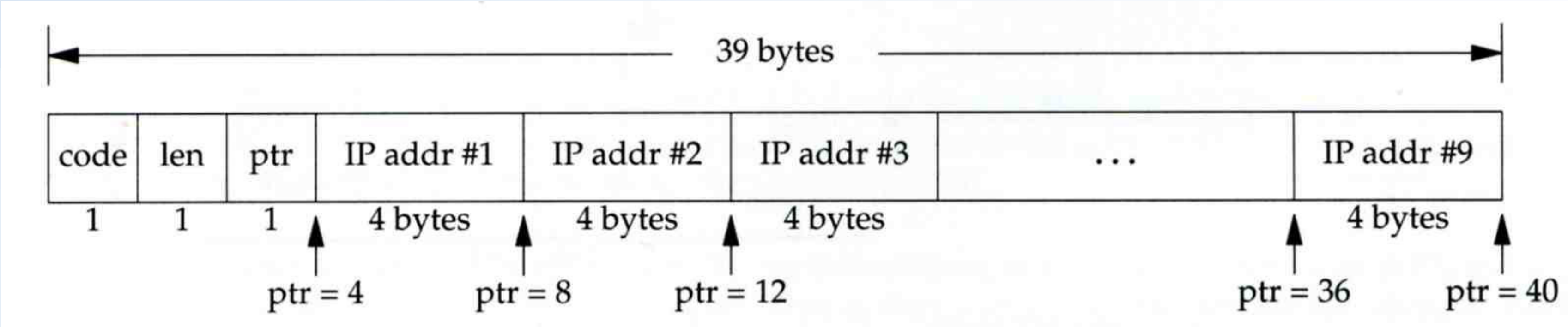
```
tytsai@tybsd:~> ping tybsd.csie.nctu.edu.tw
PING tybsd.csie.nctu.edu.tw (140.113.235.4): 56 data bytes
64 bytes from 140.113.235.4: icmp_seq=0 ttl=64 time=0.086 ms
64 bytes from 140.113.235.4: icmp_seq=1 ttl=64 time=0.053 ms
```

```
12:48:00.153614 IP tybsd.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 64: echo request seq 0
        0x0000:  0200 0000 4500 0054 d19b 0000 4001 ba21  ....E..T....@..!
        0x0010:  8c71 eb04 8c71 eb04 0800 48c1 3b3d 0000  .q...q....H.;=..
        0x0020:  8064 2442 e257 0200 0809 0a0b 0c0d 0e0f  .d$B.W..........
        0x0030:  1011 1213 1415 1617 1819 1a1b 1c1d 1e1f  ................
        0x0040:  2021 2223 2425 2627 2829 2a2b 2c2d 2e2f  .!"#$%&'()*+,-./
        0x0050:  3031 3233 3435 3637                      01234567
12:48:00.153644 IP tybsd.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 64: echo reply seq 0
        0x0000:  0200 0000 4500 0054 d19c 0000 4001 ba20  ....E..T....@...
        0x0010:  8c71 eb04 8c71 eb04 0000 50c1 3b3d 0000  .q...q....P.;=..
        0x0020:  8064 2442 e257 0200 0809 0a0b 0c0d 0e0f  .d$B.W..........
        0x0030:  1011 1213 1415 1617 1819 1a1b 1c1d 1e1f  ................
        0x0040:  2021 2223 2425 2627 2829 2a2b 2c2d 2e2f  .!"#$%&'()*+,-./
        0x0050:  3031 3233 3435 3637                      01234567
```

Ping Program – IP Record Route Option (1)

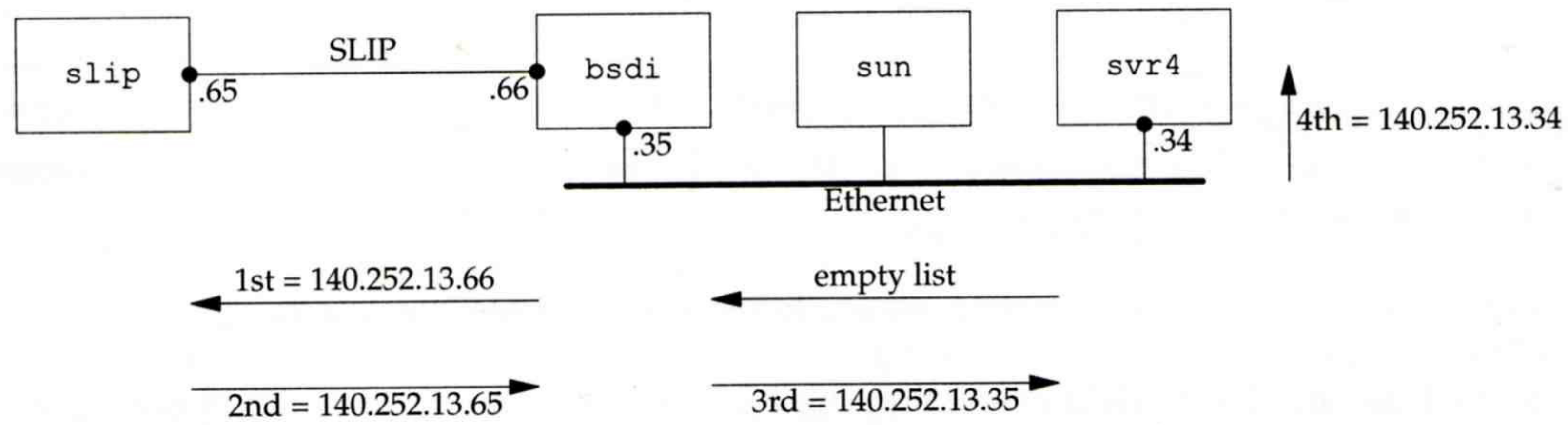
> IP Record Route Option

- ping -R
- Cause every router that handles the datagram to add its IP address to a list in the options field
- Format of Option field for IP RR Option
  - code: type of IP Option (7 for RR)
  - len: total number of bytes of the RR option
  - ptr:4 ~ 40 used to point to the next IP address
- Only 9 IP addresses can be stored
  - Limitation of IP header



Ping Program – IP Record Route Option (2)

> Ex1:




```
svr4 % ping -R slip
PING slip (140.252.13.65): 56 data bytes
64 bytes from 140.252.13.65: icmp_seq=0 ttl=254 time=280 ms
RR:     bsdi (140.252.13.66)
        slip (140.252.13.65)
        bsdi (140.252.13.35)
        svr4 (140.252.13.34)
64 bytes from 140.252.13.65: icmp_seq=1 ttl=254 time=280 ms (same route)
64 bytes from 140.252.13.65: icmp_seq=2 ttl=254 time=270 ms (same route)
^?
--- slip ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 270/276/280 ms
```

Ping Program – IP Record Route Option (3)

> Ex2

```
tytsai@tybsd:~> ping -R www.mren.com.tw
PING webserver.mren.com.tw (210.209.57.65): 56 data bytes
64 bytes from 210.209.57.65: icmp_seq=0 ttl=56 time=17.552 ms
RR:     140.113.0.165
        140.113.0.97
        bb-NCTU-CHT.TANet.edu.tw (192.83.196.113)
        140.111.230.222
        211.73.0.30
        Savecom-Nap-Peering.Twnap.net.tw (210.209.6.45)
        Transit-Peering.Twnap.net.tw (210.209.0.105)
        210.209.0.9
        78.57.209.210-twnap (210.209.57.78)
```

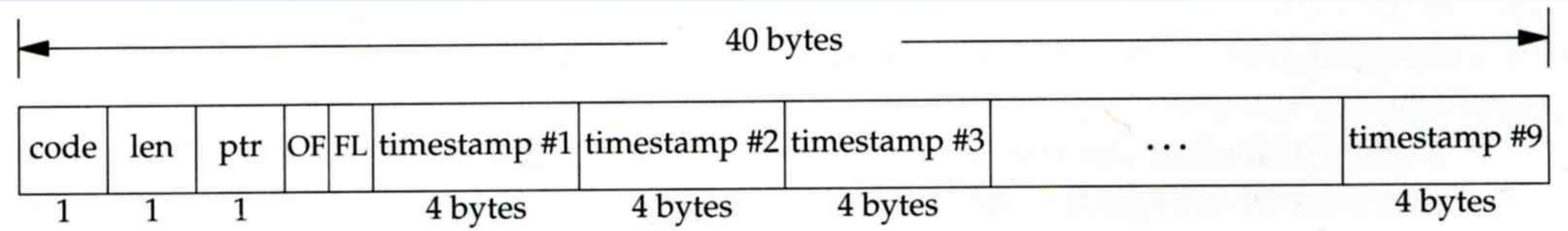
```
tytsai@tybsd:~> less result
17:37:50.804929 00:09:6b:7a:25:f7 > 00:0e:38:a4:c2:00, ethertype IPv4 (0x0800), length 138:
IP (tos 0x0, ttl  64, id 2988, offset 0, flags [none], length: 124, optlength: 40
( RR{#0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.00.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0} EOL ))
140.113.235.4 > 210.209.57.65: icmp 64: echo request seq 0

17:37:50.816999 00:0e:38:a4:c2:00 > 00:09:6b:7a:25:f7, ethertype IPv4 (0x0800), length 138:
IP (tos 0x0, ttl  56, id 27029, offset 0, flags [none], length: 124, optlength: 40
( RR{140.113.0.165 140.113.0.97 192.83.196.113 140.111.230.222 211.73.0.30 210.209.6.45
210.209.0.221 210.209.0.198 210.209.57.78#} EOL ))
210.209.57.65 > 140.113.235.4: icmp 64: echo reply seq 0
```

Ping Program – IP Timestamp Option

> Similar to RR option

– **Record Timestamp in option filed**

- code, len, ptr are the same as IP RR option
- OF
  > Overflow field
  > Router will increment OF if it can't add a timestamp because of no room left
- FL
  > Flags
  > 0: only timestamp
  > 1: both timestamp and IP address
  > 3: the sender initiates the options with up to 4 pairs of IP address and timestamp



Traceroute Program (1)

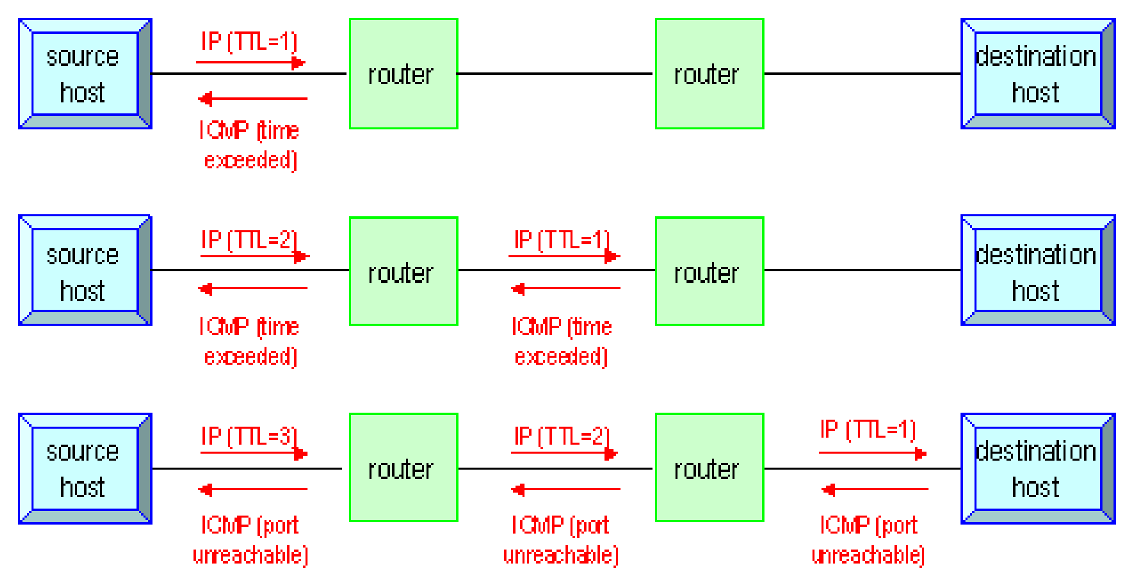
- print the route packets take to network host
- Drawbacks of IP RR options
  - Not all routers have supported the IP RR option
  - Limitation of IP header length

- Background knowledge of traceroute
  - When a router receive a datagram, , it will decrement the TTL by one
  - When a router receive a datagram with TTL = 0 or 1,
    - it will through away the datagram and
    - sends back a "Time exceeded" ICMP message
  - Unused UDP port will generate a "port unreachable" ICMP message

Traceroute Program (2)

> Operation of traceroute

- Send UDP with port > 30000, encapsulated with IP header with TTL = 1, 2, 3, … continuously
- When router receives the datagram and TTL = 1, it returns a "Time exceed" ICMP message
- When destination host receives the datagram and TTL = 1, it returns a "Port unreachable" ICMP message



Traceroute Program (3)

> Time exceed ICMP message

- Type = 11, code = 0 or 1
  - Code = 0 means TTL=0 during transit
  - Code = 1 means TTL=0 during reassembly
- First 8 bytes of datagram
  - UDP header

0	8	16	31
TYPE (11)	CODE (0 or 1)	CHECKSUM	
UNUSED (MUST BE ZERO)			
INTERNET HEADER + FIRST 64 BITS OF DATAGRAM			
. . .			

Traceroute Program (4)

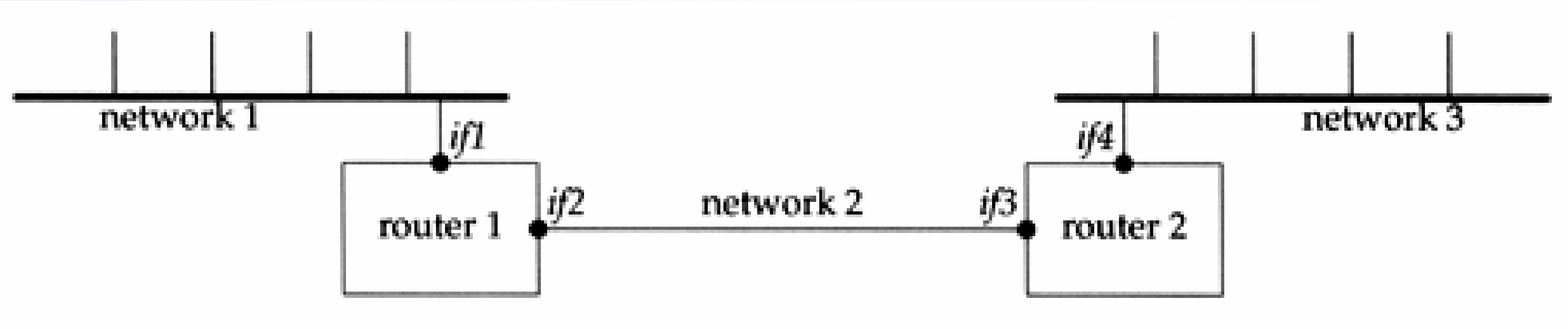
> Ex:

```
tytsai@tybsd:~> traceroute 140.113.1.1
traceroute to 140.113.1.1 (140.113.1.1), 64 hops max, 40 byte packets
 1  e3rtn-235 (140.113.235.254)  0.518 ms  0.414 ms  0.339 ms
 2  140.113.0.166 (140.113.0.166)  0.463 ms  0.359 ms  0.377 ms
 3  140.113.0.149 (140.113.0.149)  0.763 ms  0.548 ms  0.594 ms
 4  ns1.NCTU.edu.tw (140.113.1.1)  0.440 ms  0.359 ms  5.538 ms
```

```
tytsai@tybsd:~> sudo tcpdump -i fxp0 -t icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on fxp0, link-type EN10MB (Ethernet), capture size 96 bytes
IP e3rtn-235.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP e3rtn-235.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP e3rtn-235.csie.nctu.edu.tw > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP 140.113.0.166 > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP 140.113.0.166 > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP 140.113.0.166 > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP 140.113.0.149 > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP 140.113.0.149 > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP 140.113.0.149 > tybsd.csie.nctu.edu.tw: icmp 36: time exceeded in-transit
IP ns1.NCTU.edu.tw > tybsd.csie.nctu.edu.tw: icmp 36: ns1.NCTU.edu.tw udp port 33444 unreachable
IP ns1.NCTU.edu.tw > tybsd.csie.nctu.edu.tw: icmp 36: ns1.NCTU.edu.tw udp port 33445 unreachable
IP ns1.NCTU.edu.tw > tybsd.csie.nctu.edu.tw: icmp 36: ns1.NCTU.edu.tw udp port 33446 unreachable
```

Traceroute Program (5)

- The router IP in traceroute is the interface that receives the datagram
  - Traceroute from left host to right host
    - if1, if3
  - Traceroute from right host to left host
    - if4, if2



Traceroute Program – IP Source Routing Option (1)

> Source Routing

– **Sender specifies the route**

> Two forms of source routing

– **Strict source routing**

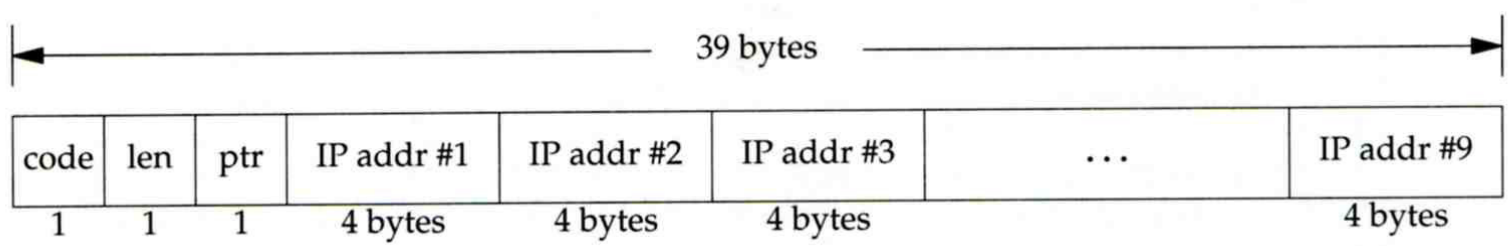
- Sender specifies the exact path that the IP datagram must follow

– **Loose source routing**

- As strict source routing, but the datagram can pass through other routers between any two addresses in the list

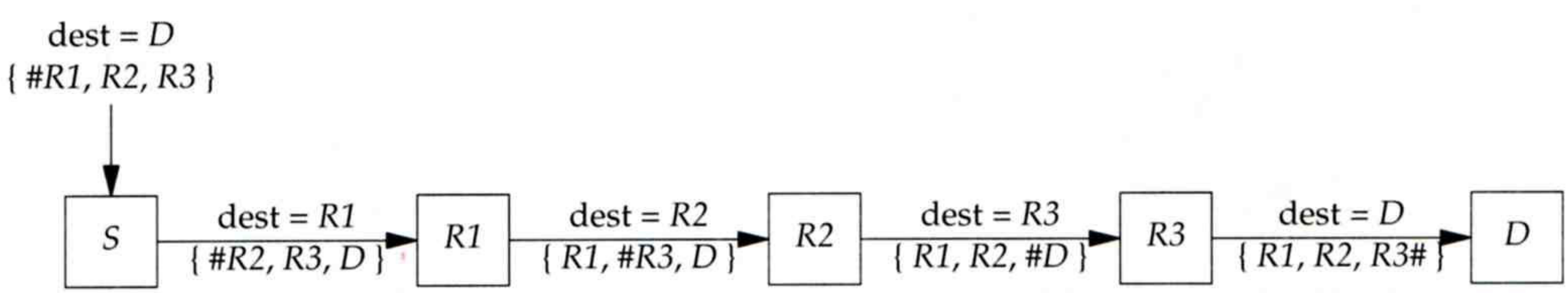
> Format of IP header option field

– **Code = 0x89 for strict and code = 0x83 for loose SR option**



Traceroute Program – IP Source Routing Option (2)

> Scenario of source routing
  - **Sending host**
    - Remove first entry and append destination address in the final entry of the list
  - **Receiving router != destination**
    - Loose source route, forward it as normal
  - **Receiving router = destination**
    - Next address in the list becomes the destination
    - Change source address
    - Increment the pointer



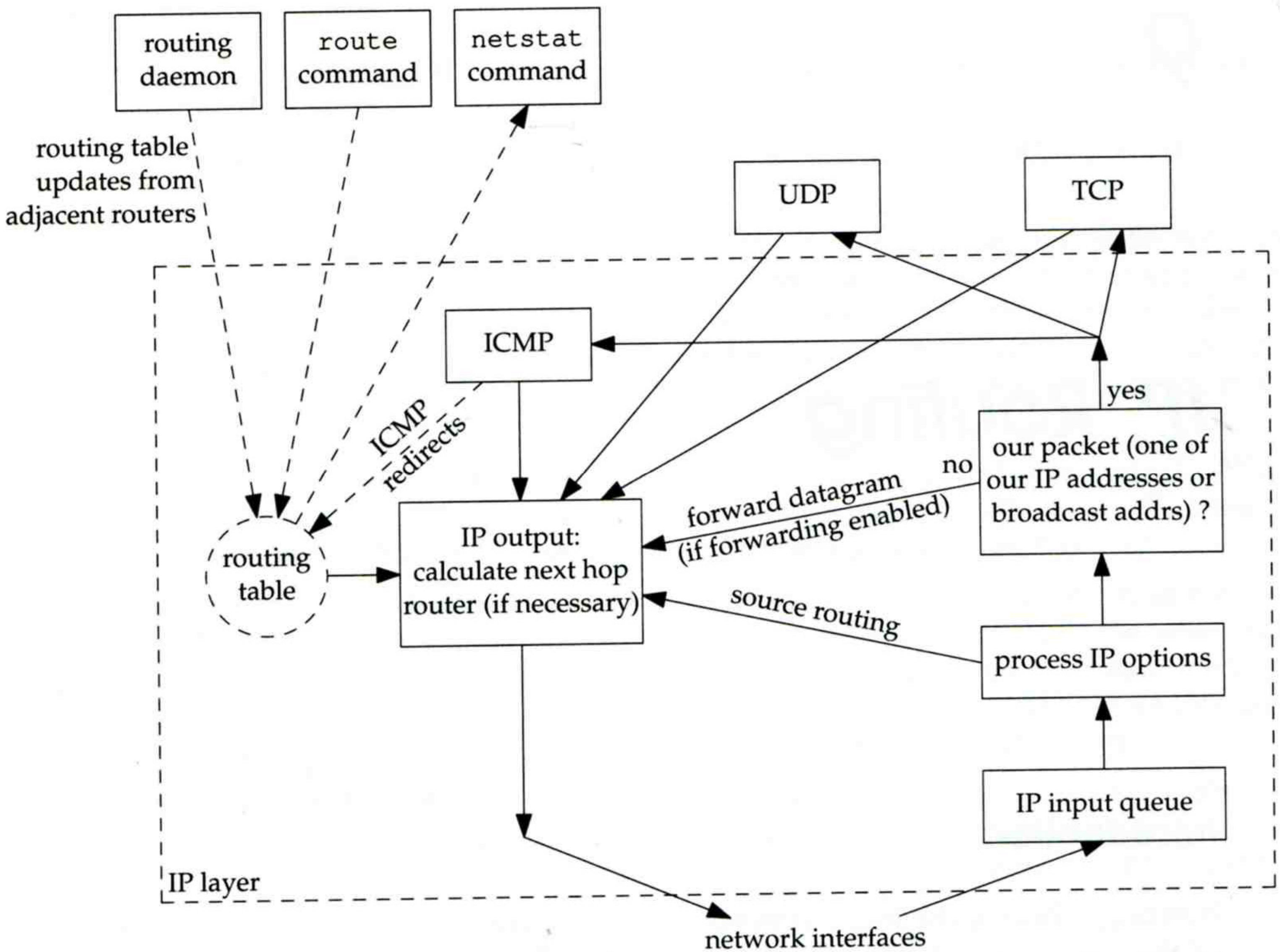
Traceroute Program – IP Source Routing Option (3)

- Traceroute using IP loose SR option
- Ex:

```
tytsai@tybsd:~> traceroute u2.nctu.edu.tw
traceroute to u2.nctu.edu.tw (211.76.240.193), 64 hops max, 40 byte packets
 1  e3rtn-235 (140.113.235.254)  0.549 ms  0.434 ms  0.337 ms
 2  140.113.0.166 (140.113.0.166)  108.726 ms  4.469 ms  0.362 ms
 3  v255-194.NTCU.net (211.76.255.194)  0.529 ms  3.446 ms  5.464 ms
 4  v255-229.NTCU.net (211.76.255.229)  1.406 ms  2.017 ms  0.560 ms
 5  h240-193.NTCU.net (211.76.240.193)  0.520 ms  0.456 ms  0.315 ms
tytsai@tybsd:~> traceroute -g 140.113.0.149 u2.nctu.edu.tw
traceroute to u2.nctu.edu.tw (211.76.240.193), 64 hops max, 48 byte packets
 1  e3rtn-235 (140.113.235.254)  0.543 ms  0.392 ms  0.365 ms
 2  140.113.0.166 (140.113.0.166)  0.562 ms  9.506 ms  0.624 ms
 3  140.113.0.149 (140.113.0.149)  7.002 ms  1.047 ms  1.107 ms
 4  140.113.0.150 (140.113.0.150)  1.497 ms  6.653 ms  1.595 ms
 5  v255-194.NTCU.net (211.76.255.194)  1.639 ms  7.214 ms  1.586 ms
 6  v255-229.NTCU.net (211.76.255.229)  1.831 ms  9.244 ms  1.877 ms
 7  h240-193.NTCU.net (211.76.240.193)  1.440 ms !S  2.249 ms !S  1.737 ms !S
```

IP Routing

Processing in IP Layer



Routing Table (1)

- % netstat –rn
- Flag
  - U: the route is up
  - G: the route is to a router (indirect route)
    - > Indirect route: IP is the dest. IP, MAC is the router's MAC
  - H: the route is to a host (Not to a network)
    - > The dest. filed is either an IP address or network address
- Refs: number of active uses for each route
- Use: number of packets sent through this route

```
tytsai@tybsd:~> netstat -rn
Routing tables

Internet:
Destination        Gateway            Flags    Refs      Use  Netif Expire
default            140.113.235.254    UGS         0   430243   fxp0
127.0.0.1          127.0.0.1          UH          0      435    lo0
140.113.235/24     link#1             UC          0        0   fxp0
140.113.235.4      00:09:6b:7a:25:f7  UHLW        0     6996    lo0
140.113.235.10     00:0d:88:b0:c2:5e  UHLW        0    15092   fxp0   1099
140.113.235.254    00:0e:38:a4:c2:00  UHLW        1        0   fxp0   1199
```

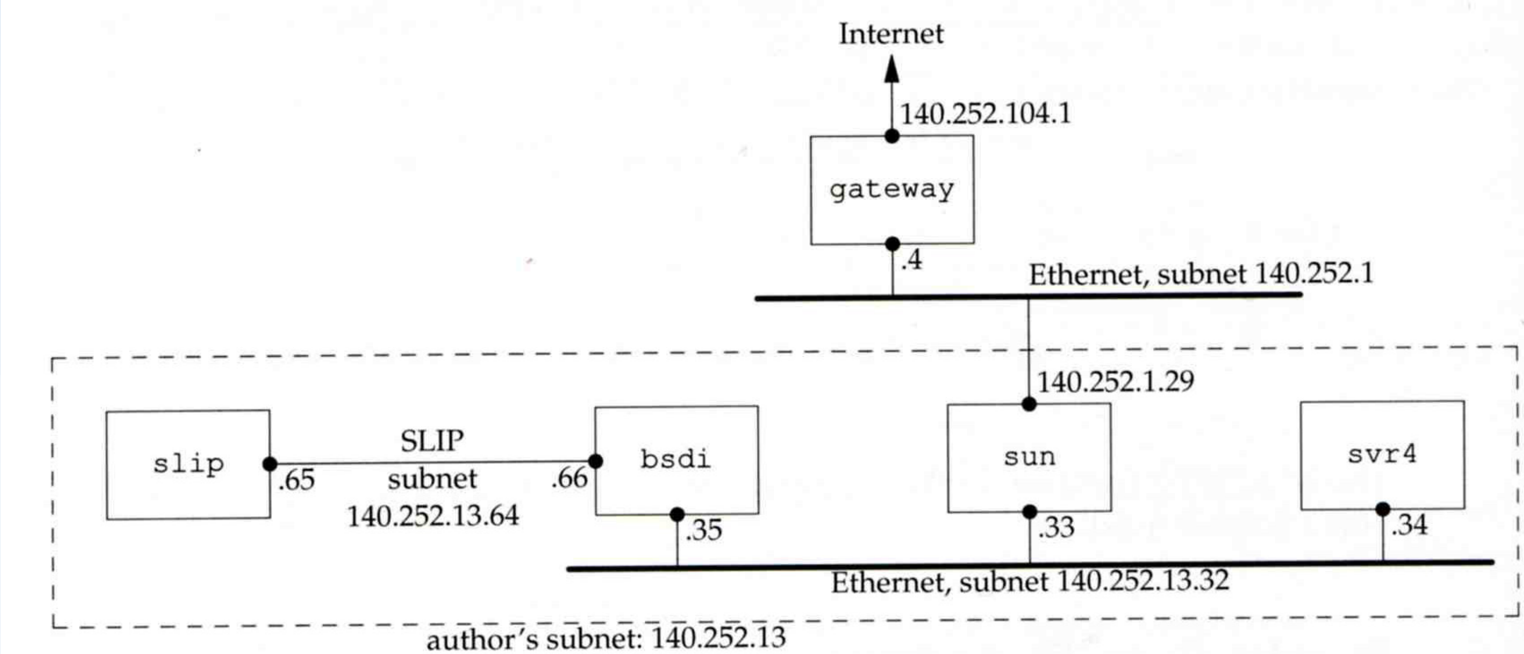
Routing Table (2)

> Ex:

```
svr4 % netstat -rn
Routing tables
Destination        Gateway            Flags     Refcnt  Use       Interface
140.252.13.65      140.252.13.35      UGH       0       0         emd0
127.0.0.1          127.0.0.1          UH        1       0         lo0
default            140.252.13.33      UG        0       0         emd0
140.252.13.32      140.252.13.34      U         4       25043     emd0
```

1. dst. = sun
2. dst. = slip
3. dst. = 192.207.117.2
4. dst. = svr4 or 140.252.13.34
5. dst. = 127.0.0.1

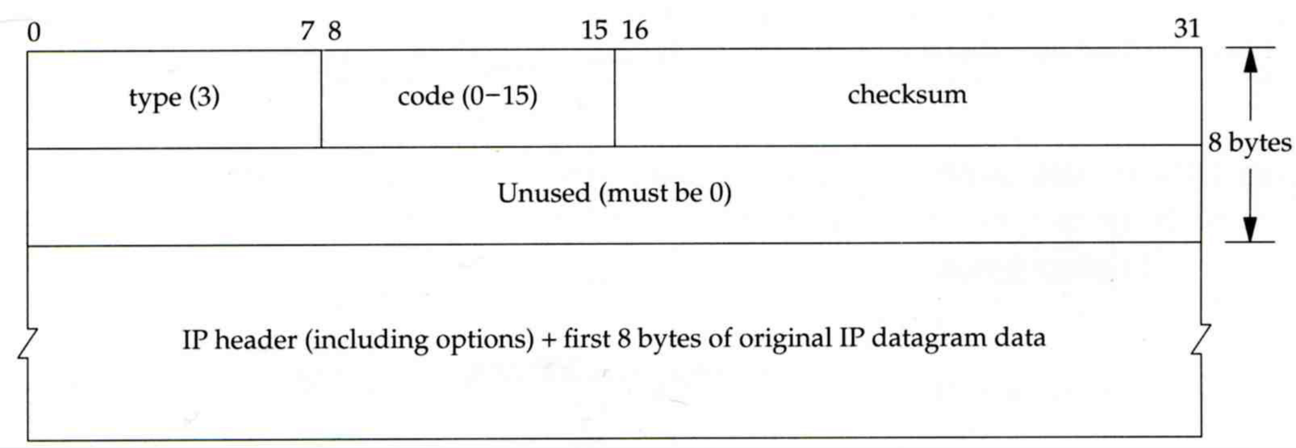
loopback



No Route to Destination

> If there is no match in routing table

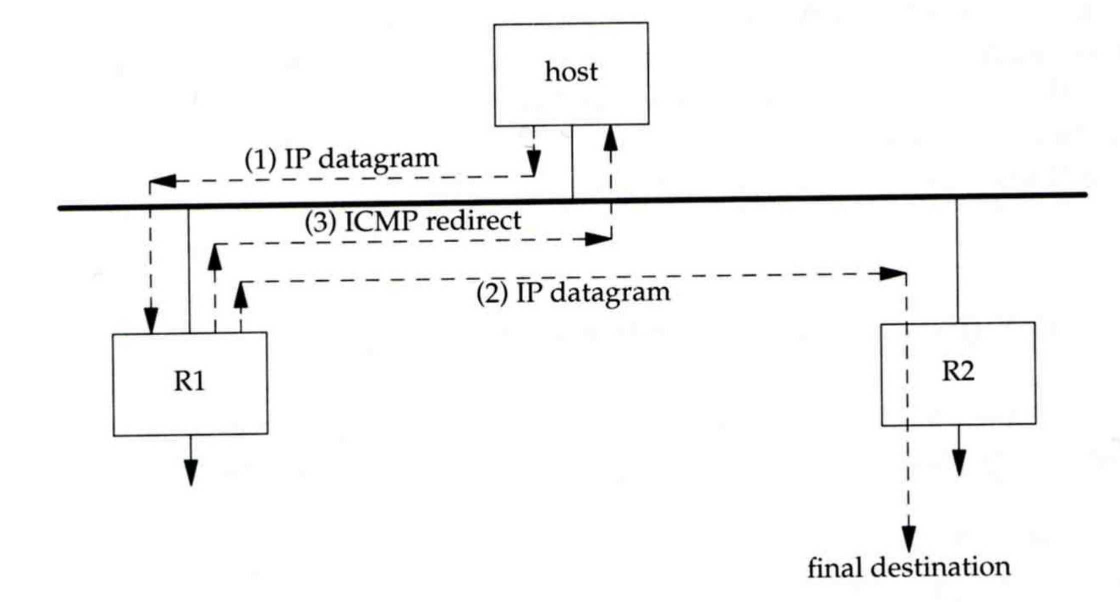
- **If the IP datagram is generated on the host**
  - "host unreachable" or "network unreachable"
- **If the IP datagram is being forwarded**
  - ICMP "host unreachable" error message is generated and sends back to sending host
  - ICMP message
    > Type = 3, code = 0 for host unreachable
    > Type = 3, code = 1 for network unreachable



ICMP Redirect Error Message (1)

> Concept

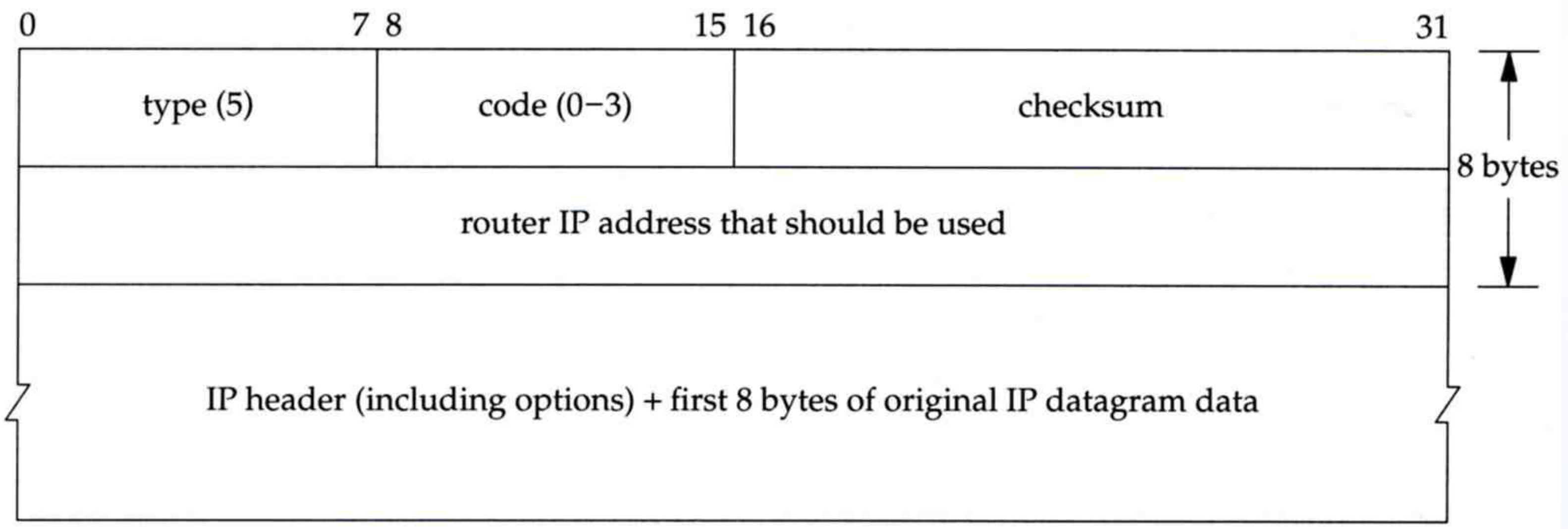
- Used by router to inform the sender that the datagram should be sent to a different router
- This will happen if the host has a choice of routers to send the packet to
- Ex:
  - R1 found sending and receiving interface are the same



ICMP Redirect Error Message (2)

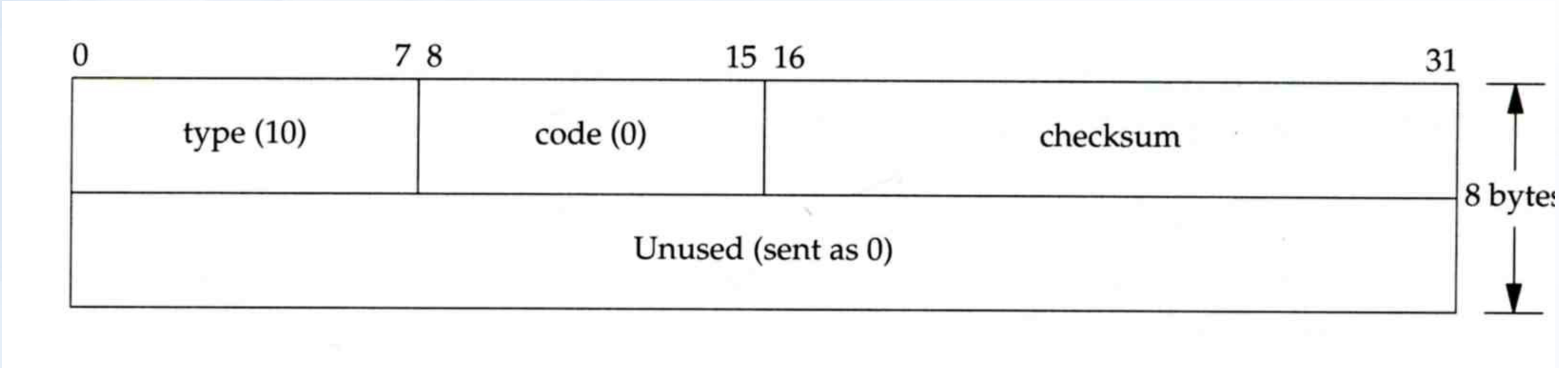
> ICMP redirect message format

- Code 0: redirect for network
- Code 1: redirect for host
- Code 2: redirect for TOS and network (RFC 1349)
- Code 3: redirect for TOS and hosts (RFC 1349)



ICMP Router Discovery Messages (1)

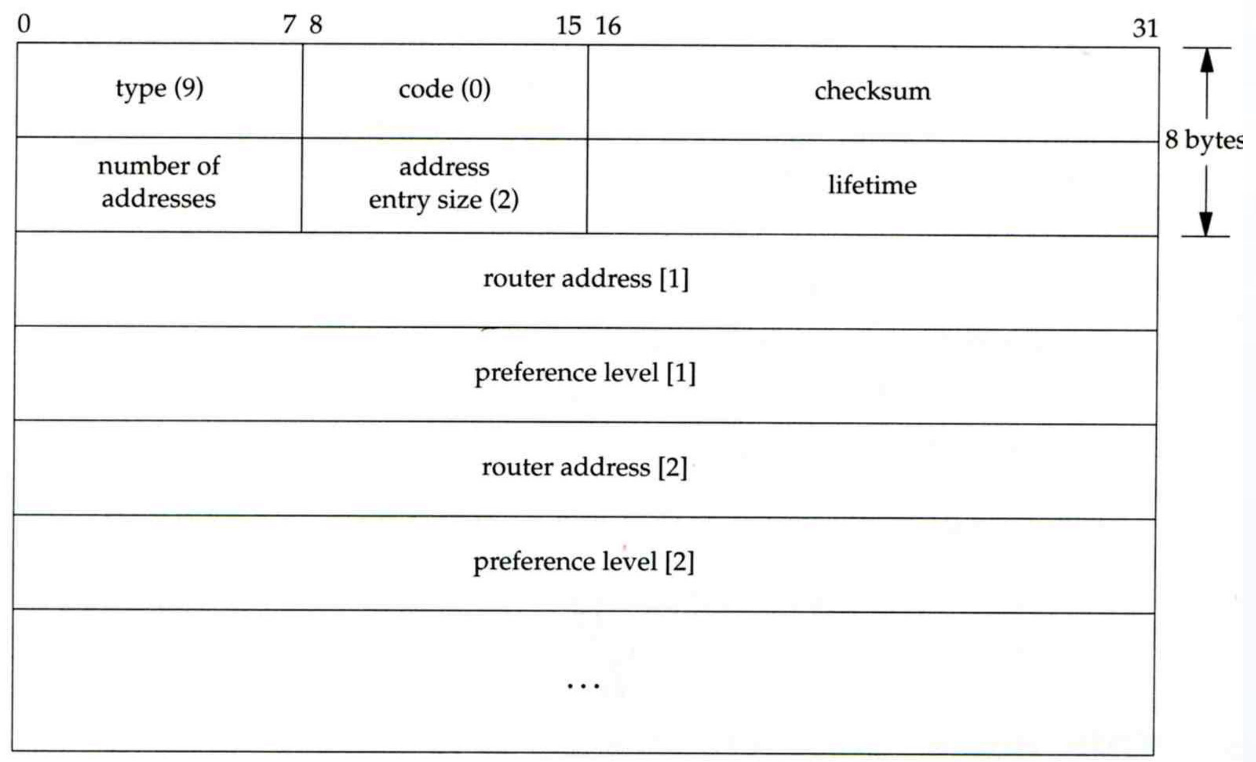
- Dynamic update host's routing table
  - **ICMP router solicitation message (懇求)**
    - Host broadcast or multicast after bootstrapping
  - **ICMP router advertisement message**
    - Router response
    - Router periodically broadcast or multicast
- Format of ICMP router solicitation message



ICMP Router Discovery Messages (2)

> Format of ICMP router advertisement message

- **Router address**
  - Must be one of the router's IP address
- **Preference level**
  - Preference as a default router address

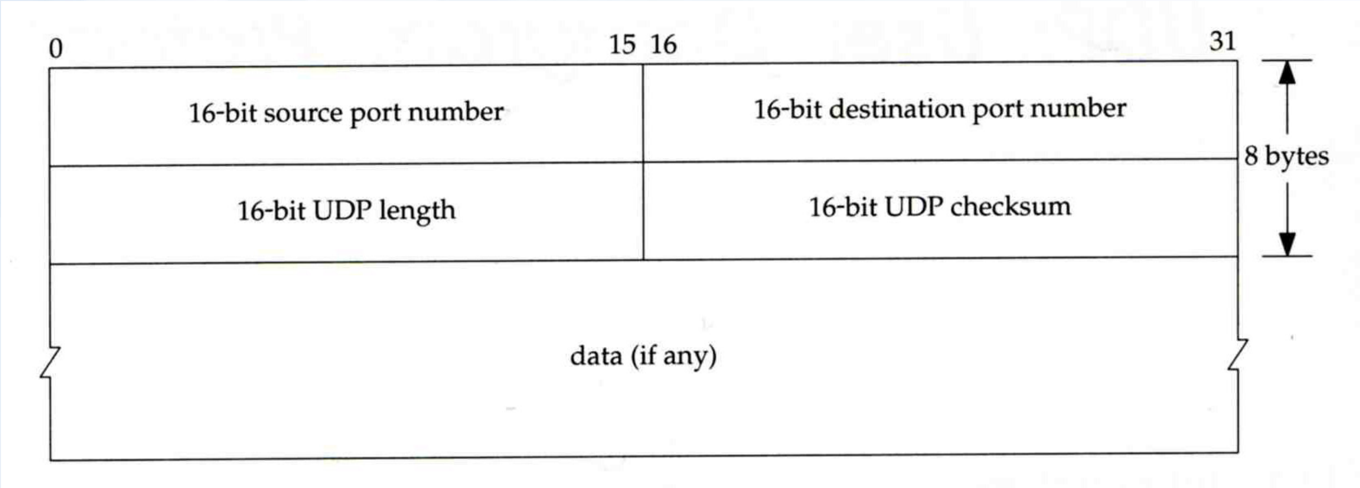


UDP –
User Datagram Protocol

UDP

> No reliability
  - **Datagram-oriented, not stream-oriented protocol**

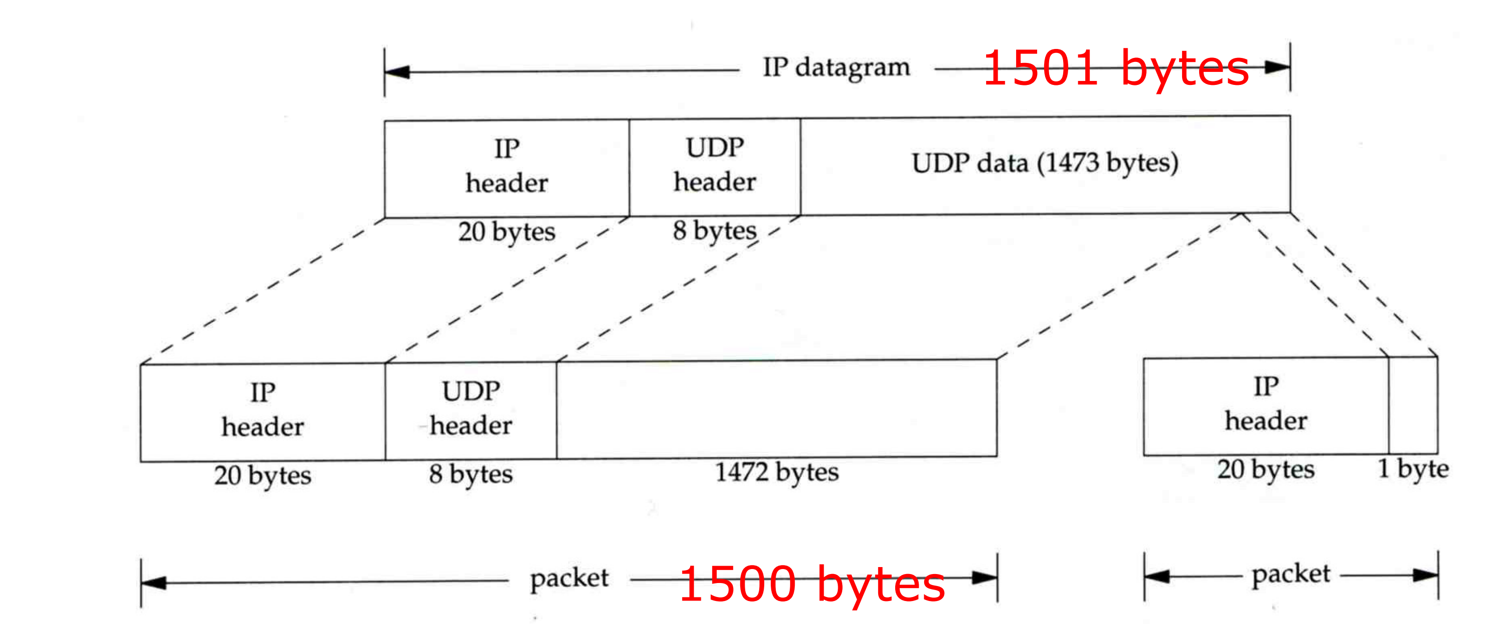
> UDP header
  - **8 bytes**
    - Source port and destination port
      > Identify sending and receiving process
    - UDP length: ≧ 8



IP Fragmentation (1)

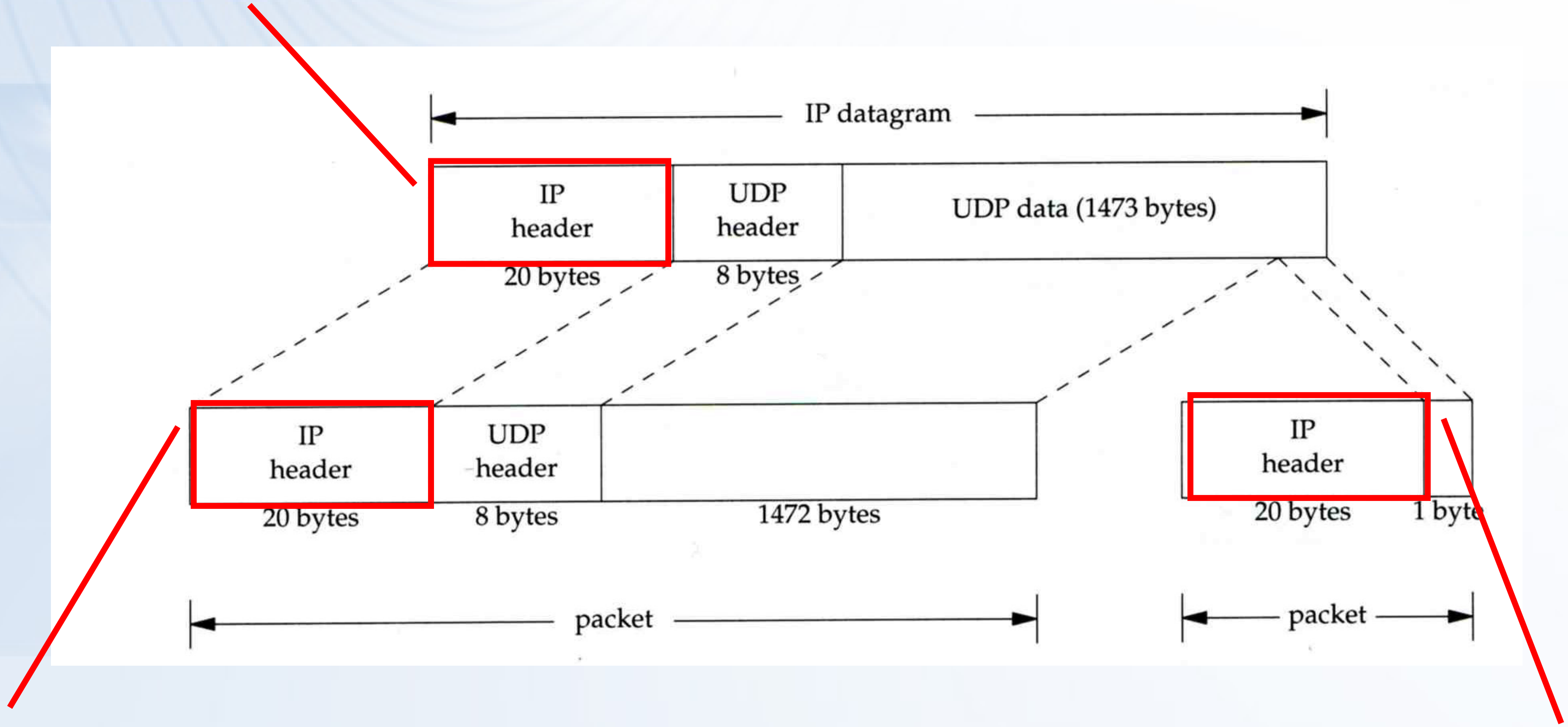
> MTU limitation

- Before network-layer to link-layer
  - IP will check the size and link-layer MTU
  - Do fragmentation if necessary
- Fragmentation may be done at sending host or routers
- Reassembly is done only in receiving host



IP Fragmentation (2)

identification:	which unique IP datagram
flags:	more fragments?
fragment offset	offset of this datagram from the beginning of original datagram



identification:	the same
flags:	more fragments
fragment offset	0

identification:	the same
flags:	end of fragments
fragment offset	1480

IP Fragmentation (3)

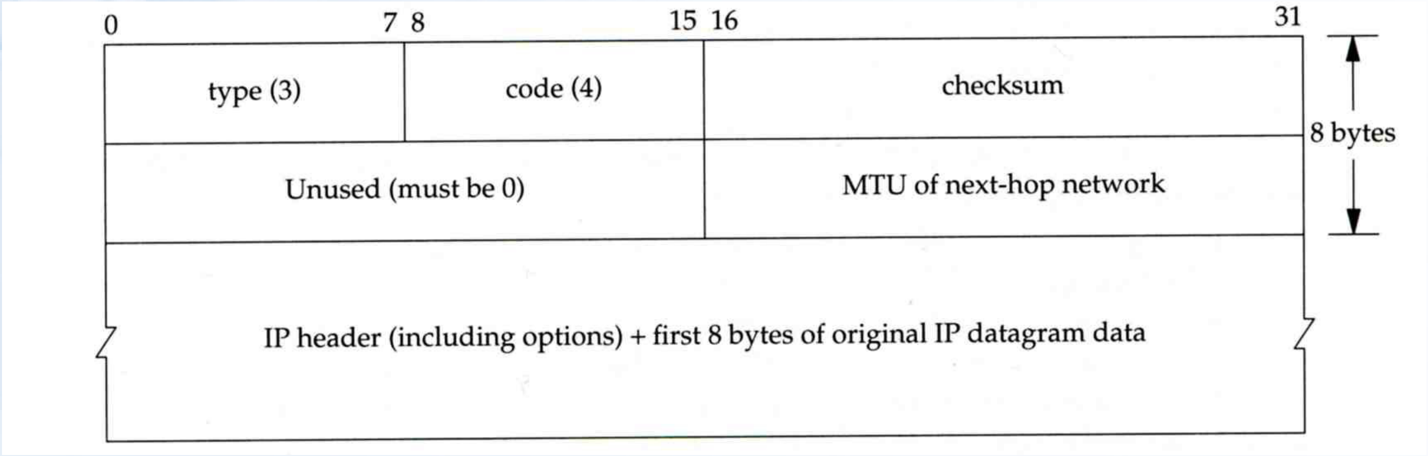
> Issues of fragmentation

- **One fragment lost, entire datagram must be retransmitted**
- **If the fragmentation is performed by intermediate router, there is no way for sending host how fragmentation did**

- **Fragmentation is often avoided**
  - There is a "don't fragment" bit in flags of IP header

ICMP Unreachable Error – Fragmentation Required

- Type=3, code=4
  - Router will generate this error message if the datagram needs to be fragmented, but the "don't fragment" bit is turn on in IP header
- Message format



ICMP Source Quench Error

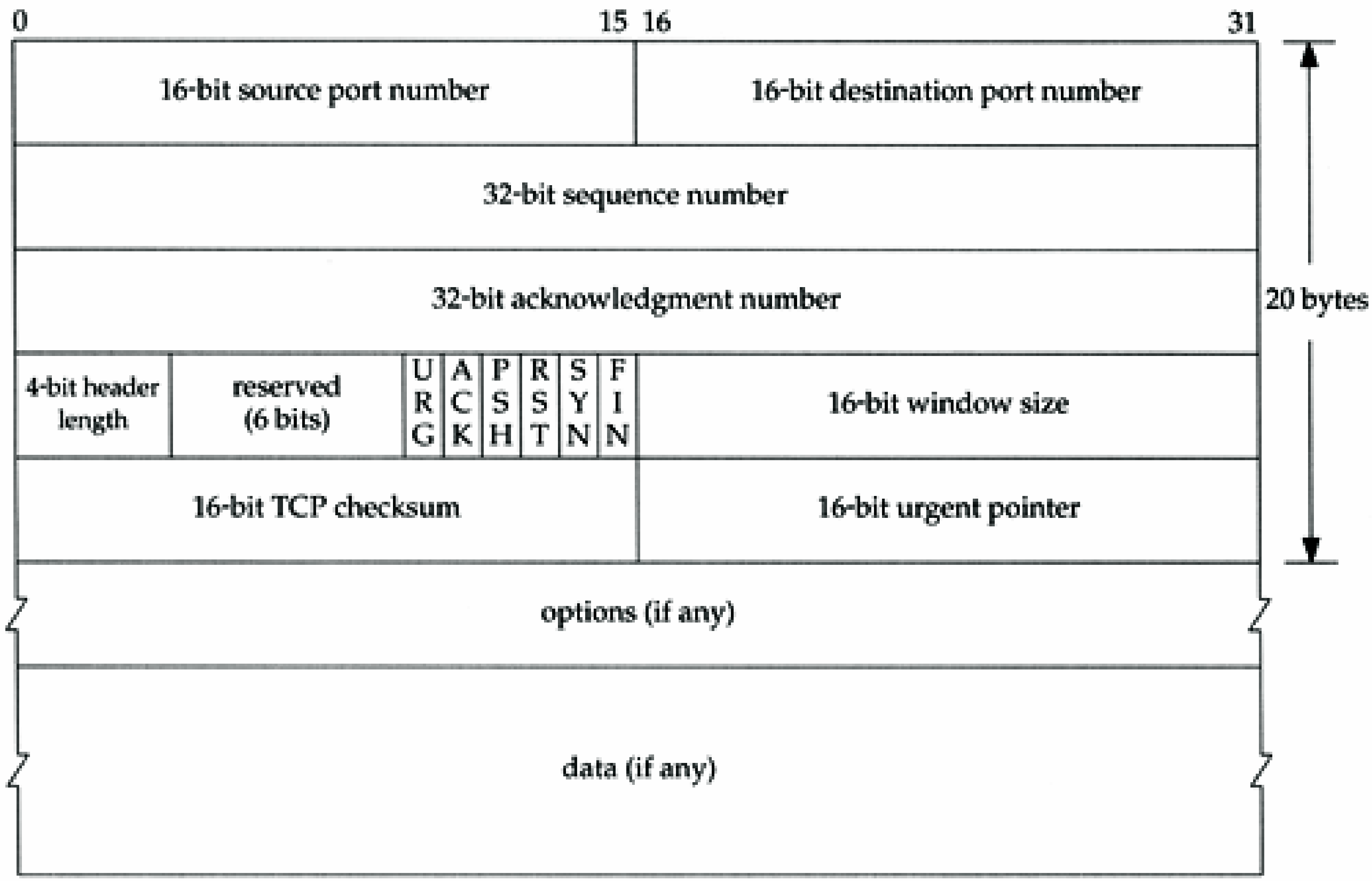
> Type=4, code=0
  - **May be generated by system when it receives datagram at a rate that is too fast to be processed**
  - **Host receiving more than it can handle datagram**
    - Send ICMP source quench or
    - Throw it away
  - **Host receiving UDP source quench message**
    - Ignore it or
    - Notify application

TCP –
Transmission Control Protocol

TCP

- Services
  - **Connection-oriented**
    - Establish TCP connection before exchanging data
  - **Reliability**
    - Acknowledgement when receiving data
    - Retransmission when timeout
    - Ordering
    - Discard duplicated data
    - Flow control

TCP Header (1)

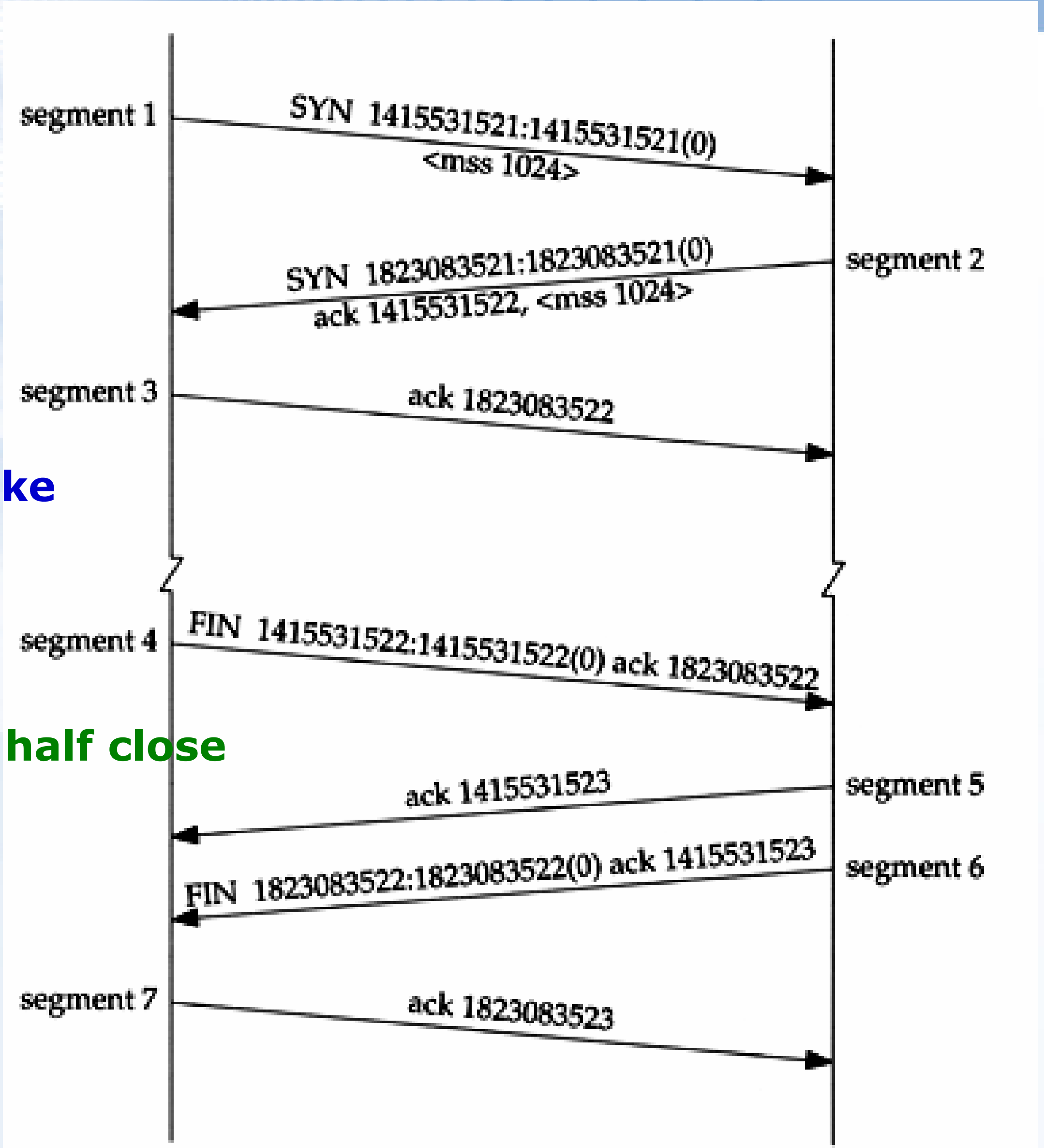

0
15 16
31
16-bit source port number
16-bit destination port number
32-bit sequence number
32-bit acknowledgment number
4-bit header length
reserved (6 bits)
URG
ACK
PSH
RST
SYN
FIN
16-bit window size
16-bit TCP checksum
16-bit urgent pointer
options (if any)
data (if any)
20 bytes

TCP Header (2)

> Flags

- **SYN**
  - Establish new connection
- **ACK**
  - Acknowledgement number is valid
  - Used to ack previous data that host has received
- **RST**
  - Reset connection
- **FIN**
  - The sender is finished sending data

TCP connection establishment and termination



Three-way handshake

TCP's half close