

Introduction to Network Security

Chapter 6

Network Layer Protocols

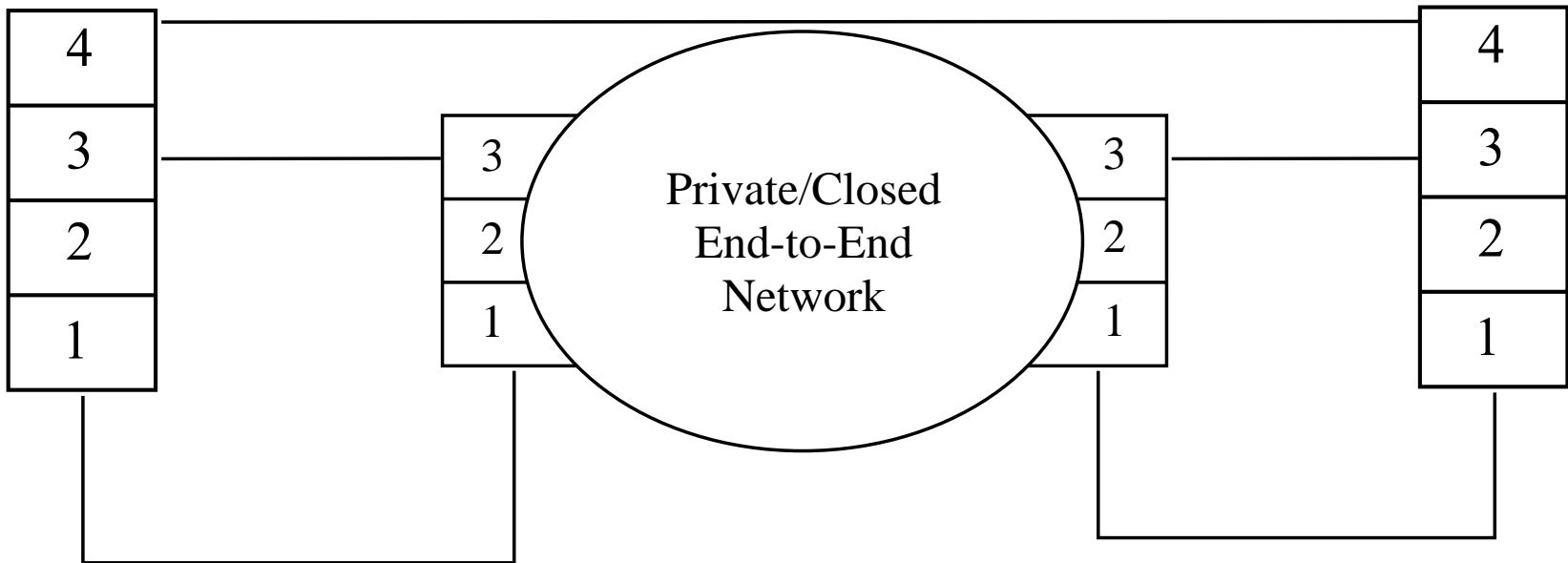
Topics

- The network layer
- IP V4
- BOOTP & DHCP
- IP V6
- Common IP countermeasures

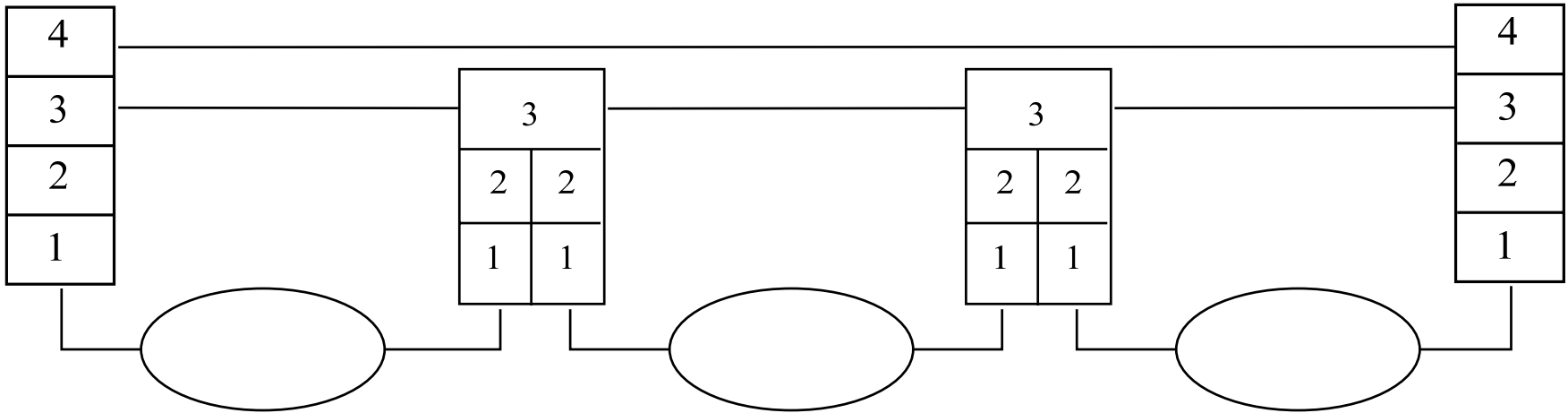
Network Layer

- Two Types:
 - Network access layer
 - Connection to a private end-to-end network
 - Used by ISPs to interconnect
 - Internetwork Layer
 - Distributed set of network layers working together
 - Used throughout the Internet

Network Access



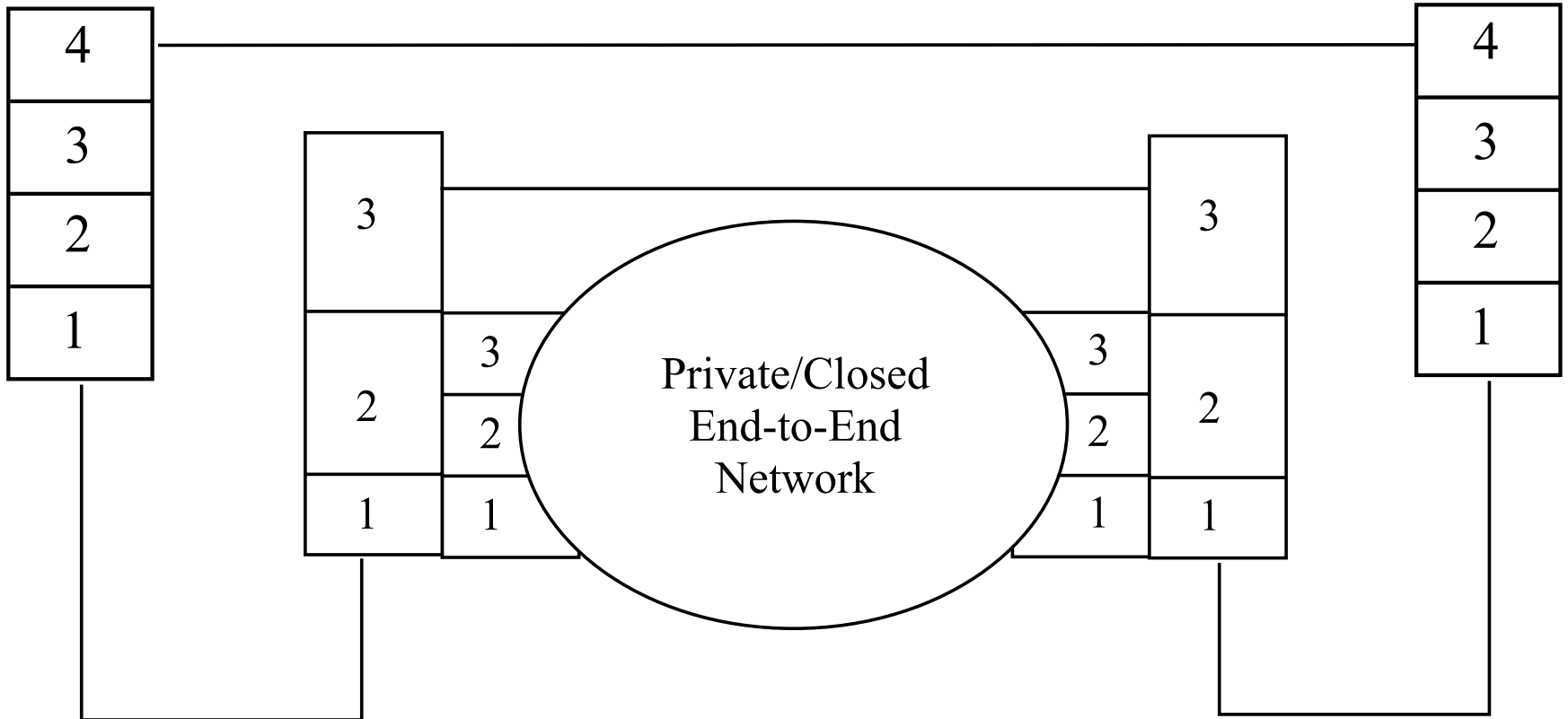
Internetwork



Differences between networks

| Differences | Remediation |
|---|--|
| Physical network layer addressing schemes | The network will need to adapt to the different address types which is more complex in devices like routers |
| Maximum and minimum packet sizes | The network layer will need to implement segmentation and reassembly |
| Network access methods | The network layer will need to provide buffering which handle different access methods, especially in a router |
| Error and flow control | The network layer will need to handle lost and delayed packets |
| Machine and user authentication | The network layer will need to provide authentication to the physical network if required |

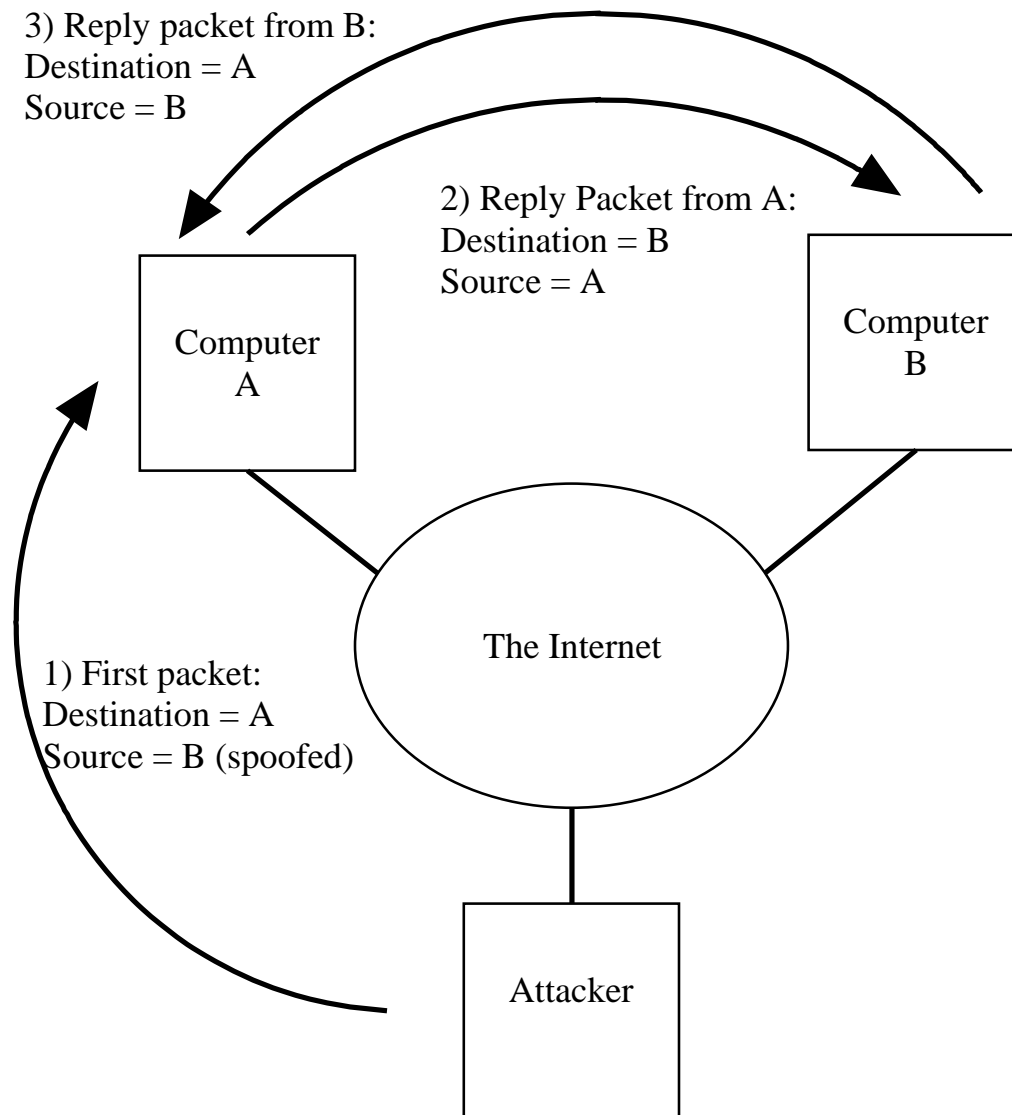
Using network access



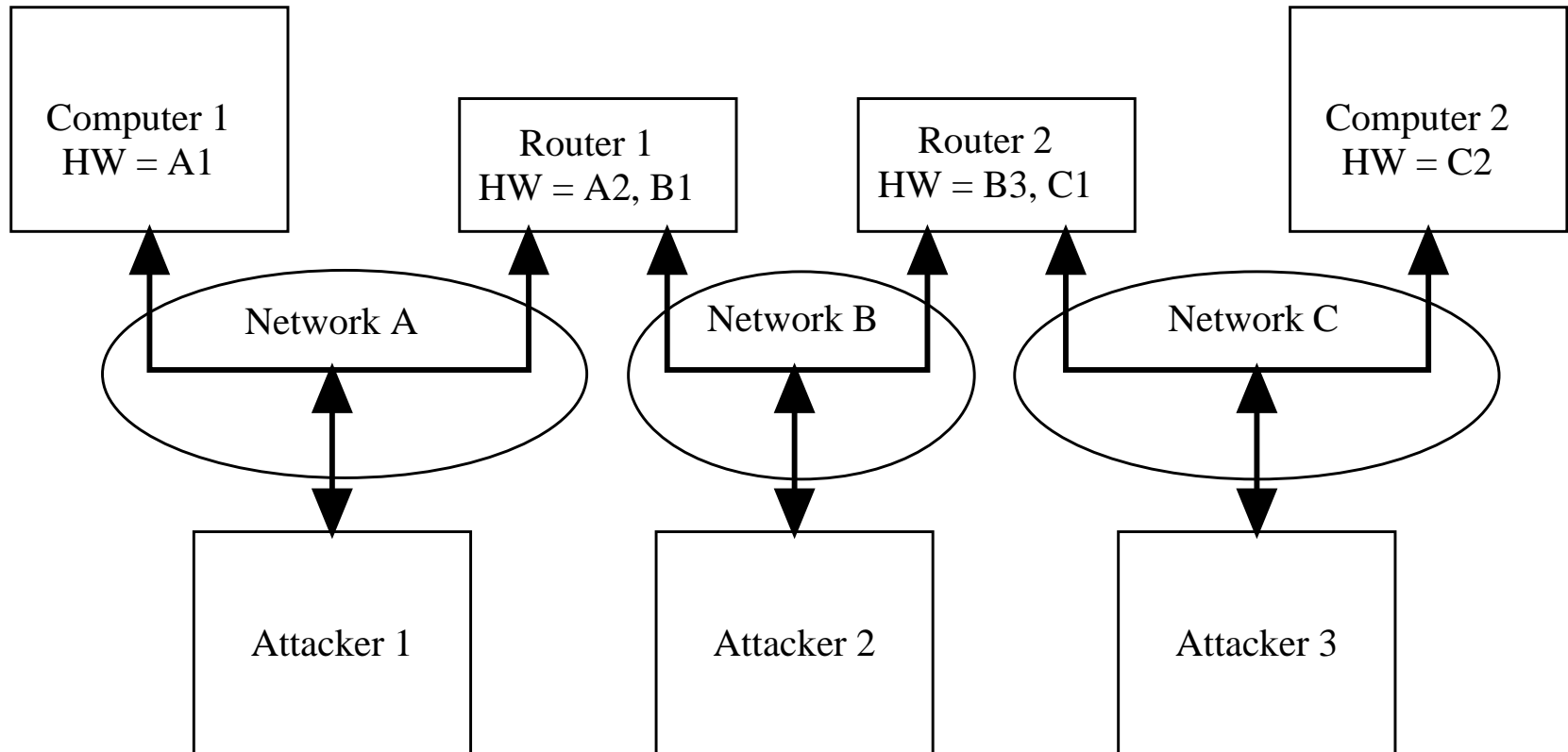
Common Attack Methods

- Address Spoofing
- Network Sniffing
- Network Scanning

Address Spoofing



Network Sniffing



Network Scanning

- Network layer is a global address space
- You can use the network layer protocols to locate targets
- Catch-22, you need to be able to locate a device to send packets to it, but that also allows someone to see if a device exists.

IP Layer Topics

- 1. Addressing
- 2. Routing
- 3. Packet Formats
- 4. ICMP Internet Control Message Protocol

Addressing

- We will look at three different parts of addressing.
 - 1. IP addresses
 - 2. Name to IP addresses translation
 - 3. IP address to station datalink address

IP Addresses

- Globally unique
- Two parts
 - Network address
 - Host address

Example IP addresses

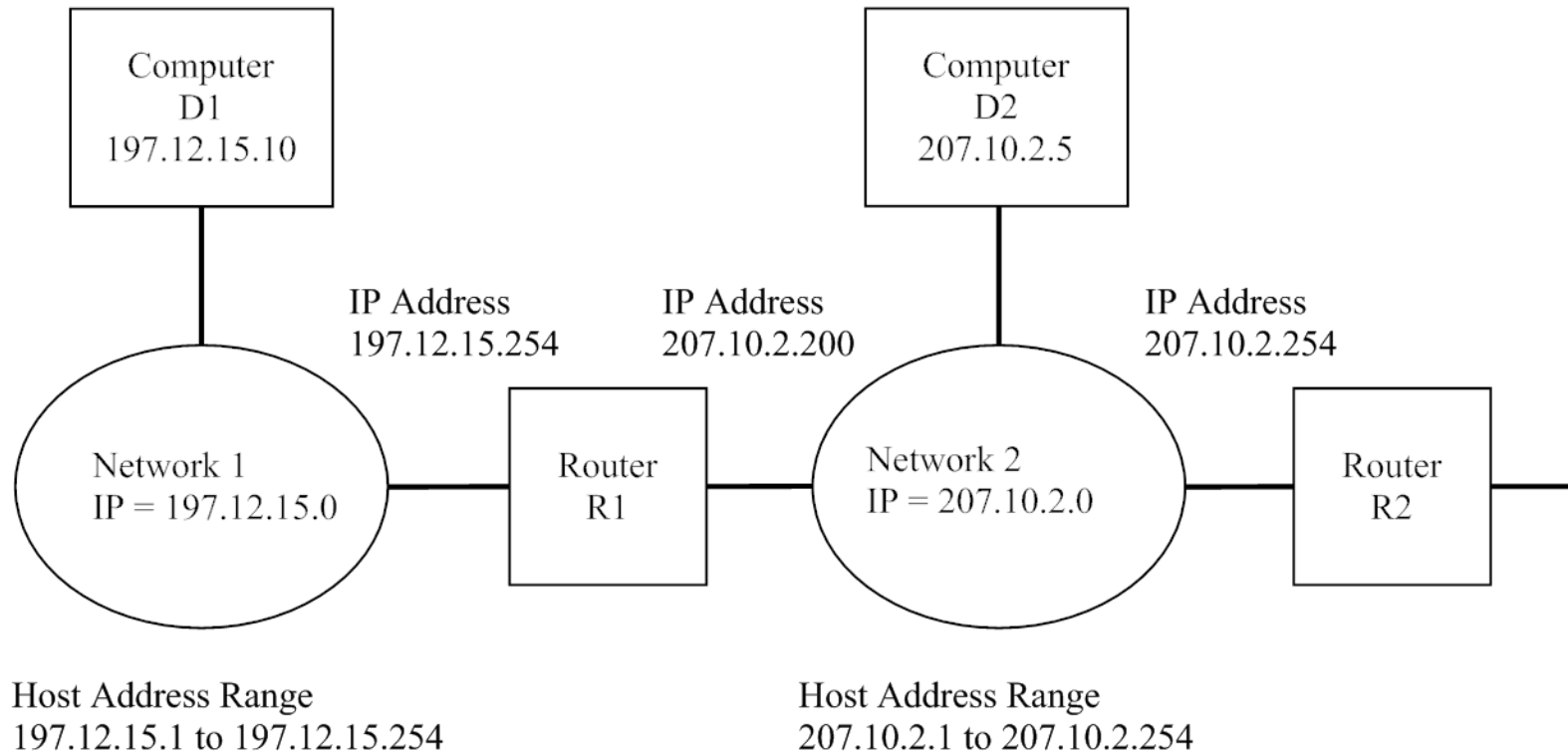


Figure 6.3 Example IP Addresses

IP Address Space

- In Version 4 the IP address is 32 Bits
- Total IP address space is 4,294,967,296

IP addresses

- The IP address is written as a four-tuple where each tuple is in decimal and are separated by a "." (called a dot). When talking about an address you pronounce the word dot. So 129.186.5.102 is pronounced 129 dot 186 dot 5 dot 102

IP Addresses

| | | |
|---|--------------------------|---------------------|
| A | 0 + Netid (7 bits) | Host ID (24 bits) |
| B | 10 + NetID (14 bits) | Host ID (16 bits) |
| C | 110 + Net ID (21 Bits) | Host ID (8 bits) |
| D | 1110 + Multicast address | |
| E | 11110 Reserved | |

IP Address Allocation

| Class | # of Addresses | % |
|-------|--------------------------|-------|
| A | $2^{31} = 2,147,483,648$ | 50% |
| B | $2^{30} = 1,073,741,824$ | 25% |
| C | $2^{29} = 536,870,912$ | 12.5% |
| D | $2^{28} = 268,435,456$ | 6.25% |
| E | $2^{28} = 268,435,456$ | 6.25% |

IP Address Distribution

| Class | First network | Last network | # of Networks | # of hosts per network |
|-------|---------------|---------------|---------------|------------------------|
| A | 1.0.0.0 | 126.0.0.0 | 126 | 16,777,214 |
| B | 128.0.0.0 | 191.255.0.0 | 16,384 | 65,534 |
| C | 192.0.0.0 | 223.255.255.0 | 2,097,152 | 254 |

IP Address Space

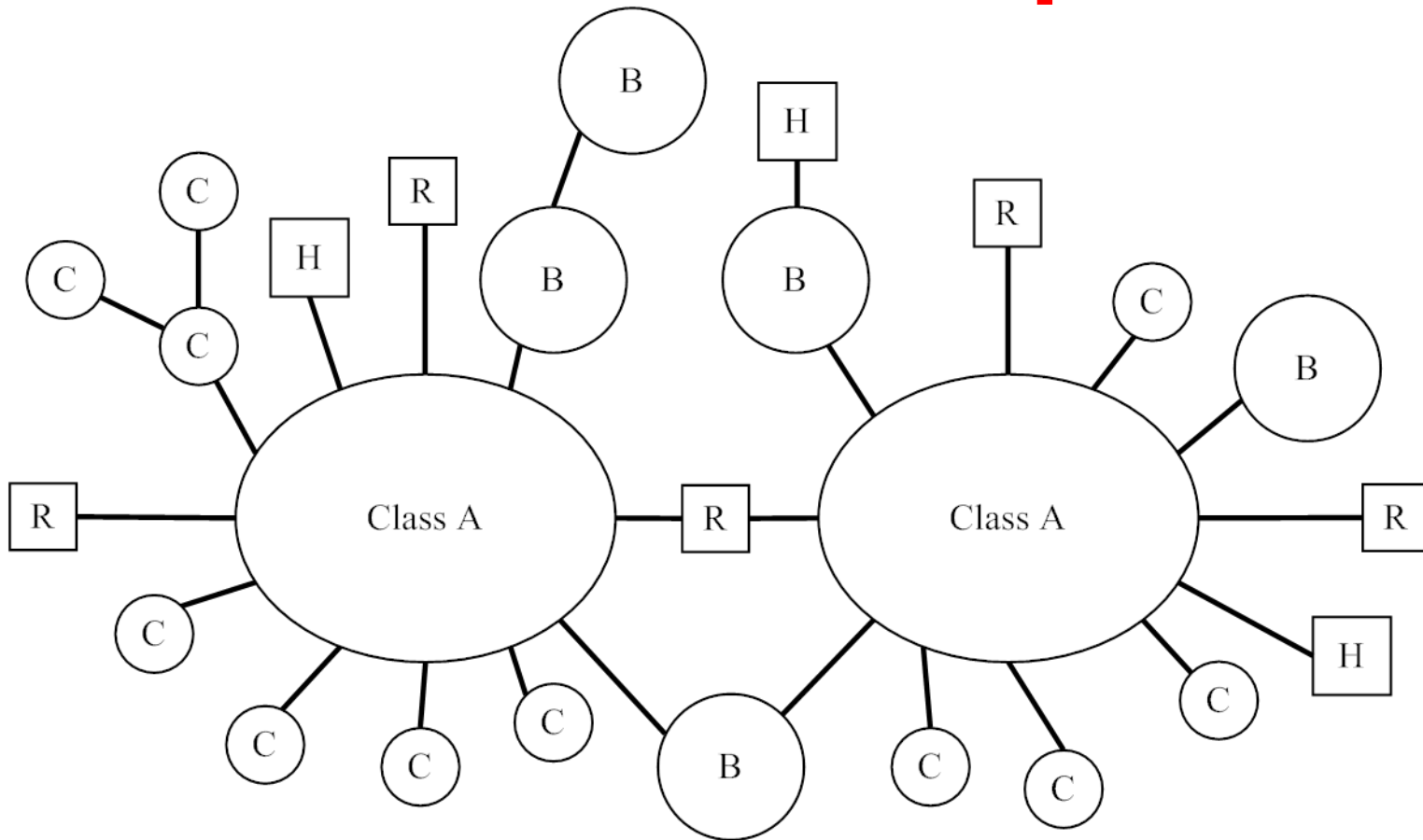


Figure 6.4 IP Address Space

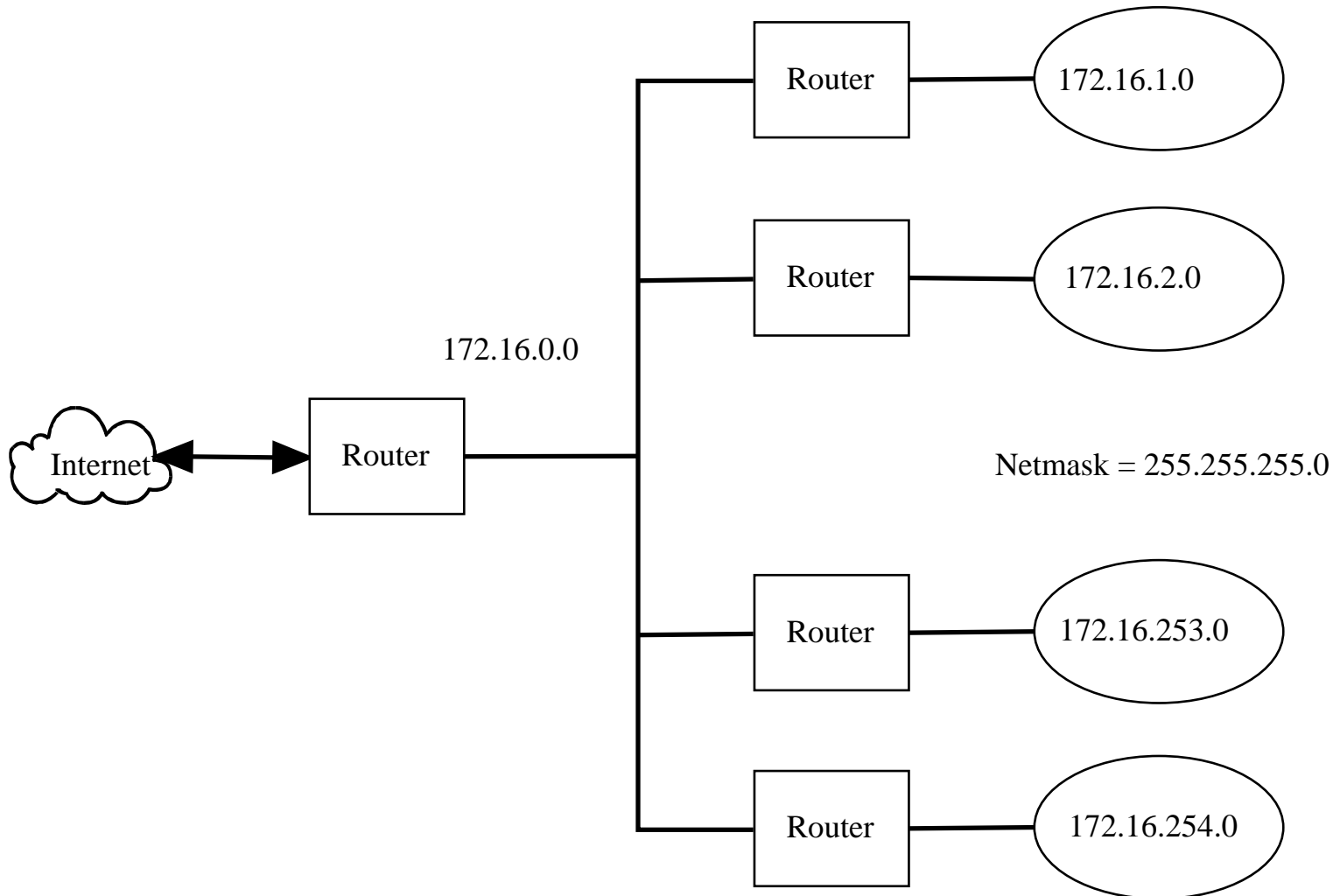
Special Addresses

| | |
|-----------------|---|
| 0.0.0.0 | This host, only at system startup, never a dest address |
| 0.0.host | Host on this net, only at system startup, never a dest address |
| 255.255.255.255 | Limited Broadcast (local net only) Never a source address |
| Net.255.255 | Directed broadcast address for net. Never a source address |
| 127.0.0.1 | Loopback |

Loopback address

- The Class A address 127.0.0.0 is reserved for loopback and is designed for testing and interprocess communications on the local machine. When a program uses the loopback address the local host returns the data without sending across the network. The address 127.0.0.0 should never be seen on the network and a host or gateway should never propagate routing information on network 127.

Subnets



Classless Addresses CIDR

| Class | Netmask | Example CIDR address |
|-------|---------------|----------------------|
| A | 255.0.0.0 | 15.35.26.234/8 |
| B | 255.255.0.0 | 129.186.34.54/16 |
| C | 255.255.255.0 | 192.168.1.30/24 |

Routing

- All hosts and gateways store routing tables
- Each row in the route table contains:
 - Destination address or address range
 - Next hop for that destination address range
 - The physical interface to use for that address range. (i.e.: which Ethernet card to use)

Example:

| Destination | Next | Interface |
|--------------------|---------------|------------------|
| 129.186.4.0 | 129.186.5.254 | en0 |

Routing

In order to route a packet:

1. IP layer finds the route table entry where the destination address matches the range given in the table.
2. If the next hop falls within the local network, the packet is sent directly to the destination. Otherwise the packet is sent to the next hop.

Next Hop Routing

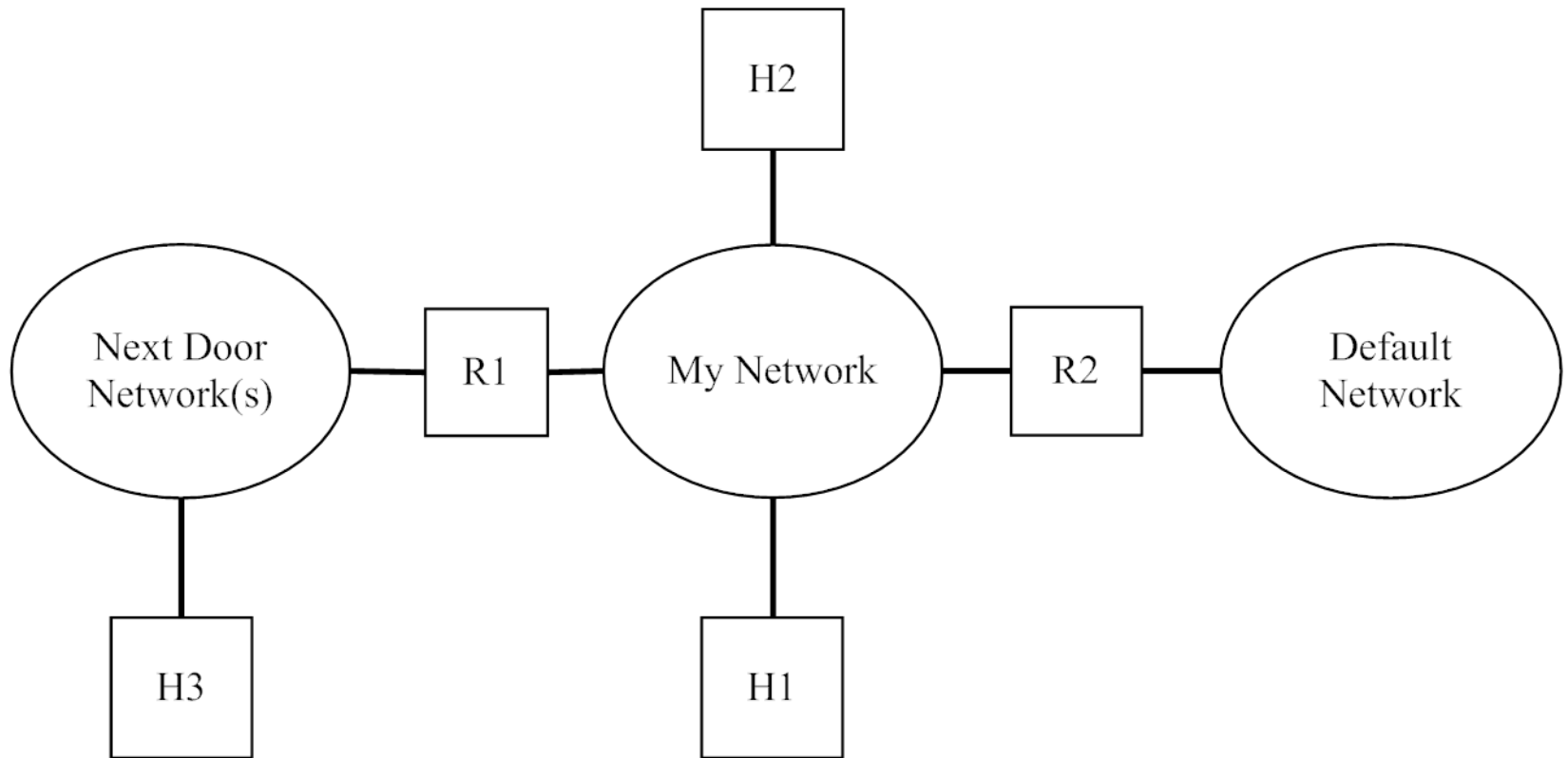


Figure 6.6 IP Next Hop Routing

Routing

Netmask

- Determines which part of the IP address is network and which part is host
- Allows for the ability to create subnetworks
- Example: a netmask of 255.255.255.0 indicates that the first 3 bytes of the IP address is the network, and the last 8 bytes is the host.
- The above netmask allows for 254 subnetworks each with up to 254 attached hosts.
- The following are examples of subnetworks:
 - 129.186.5.0 129.186.15.0 129.186.55.0

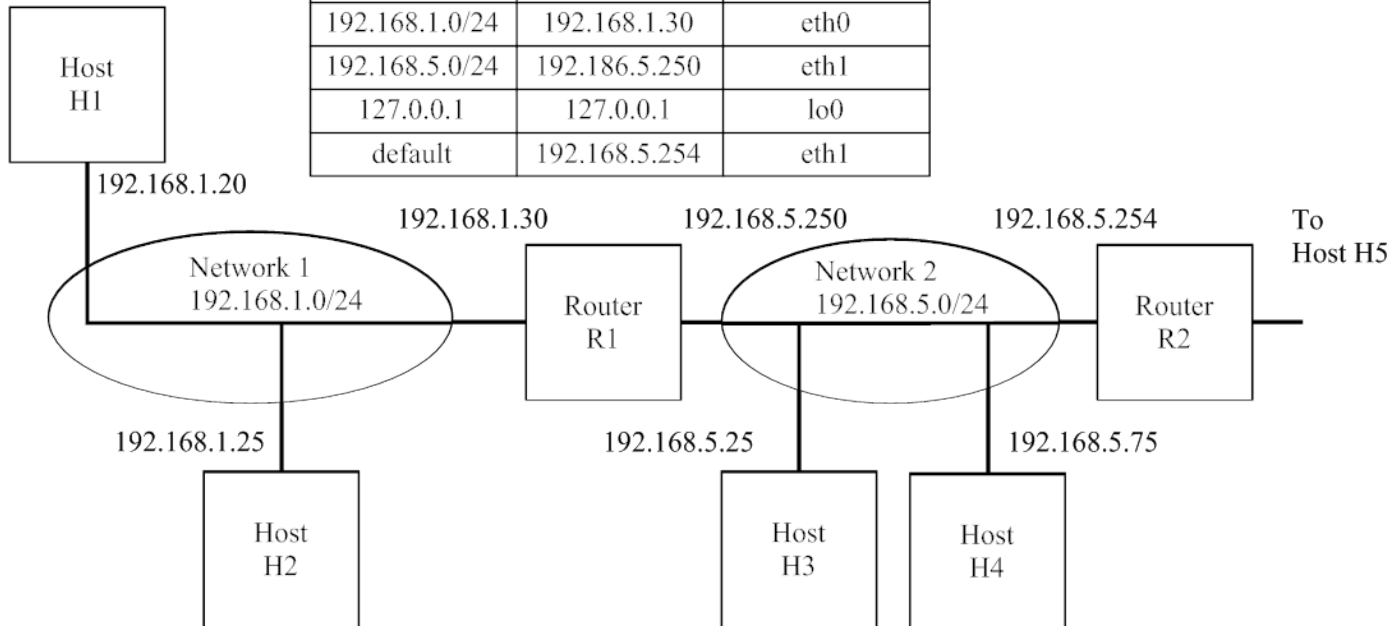
Routing

We will study routing using three scenarios:

1. A simple network with only one router
2. A network with multiple routers
3. A single network with multiple IP's

Route Table Router R1

| Destination | Next Hop | Interface |
|----------------|---------------|-----------|
| 192.168.1.0/24 | 192.168.1.30 | eth0 |
| 192.168.5.0/24 | 192.186.5.250 | eth1 |
| 127.0.0.1 | 127.0.0.1 | lo0 |
| default | 192.168.5.254 | eth1 |



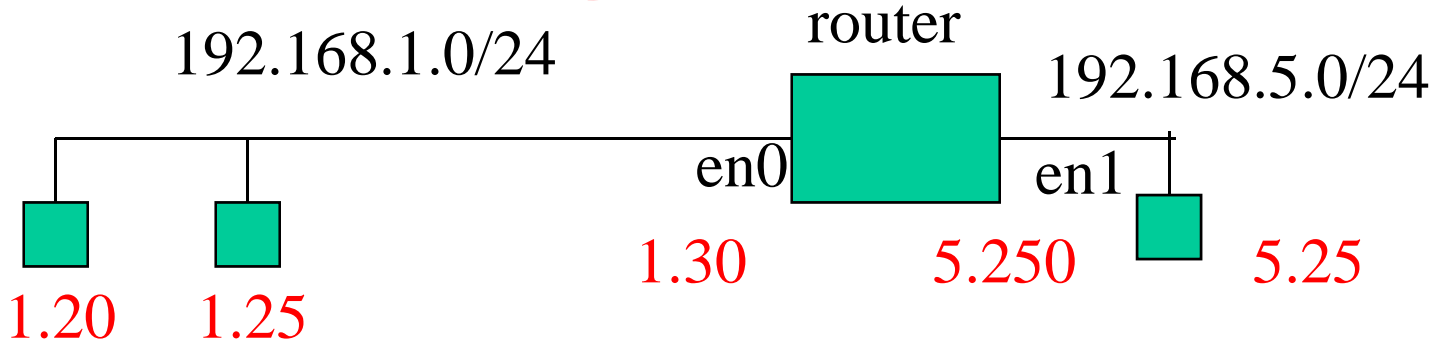
Route Table Host H1

| Destination | Next Hop | Interface |
|----------------|--------------|-----------|
| 192.168.1.0/24 | 192.168.1.20 | eth0 |
| 127.0.0.1 | 127.0.0.1 | lo0 |
| default | 192.168.1.30 | eth0 |

Route Table Host H3

| Destination | Next Hop | Interface |
|----------------|---------------|-----------|
| 192.168.5.0/24 | 192.168.5.25 | eth0 |
| 192.168.1.0/24 | 192.186.5.250 | eth0 |
| 127.0.0.1 | 127.0.0.1 | lo0 |
| default | 192.168.5.254 | eth0 |

Routing Scenario 1



Packet from H1 to H2 (same network)

| IP Address | | Hardware Address | |
|------------|------|------------------|------|
| SRC | DEST | SRC | DEST |
| H1 | H2 | H1 | H2 |

Packet from H1 to H3 (Next door network)

| IP Address | | Hardware Address | |
|------------|------|------------------|----------|
| SRC | DEST | SRC | DEST |
| H1 | H2 | H1 | R1 (EN0) |
| H1 | H2 | R1 (EN1) | H3 |

Routing Scenario 1

Steps involved in sending a packet from H1 to H2:

| Destination | Next Hop |
|--------------------|-----------------|
| 192.168.1.0/24 | 192.168.1.20 |
| Default | 192.168.1.30 |

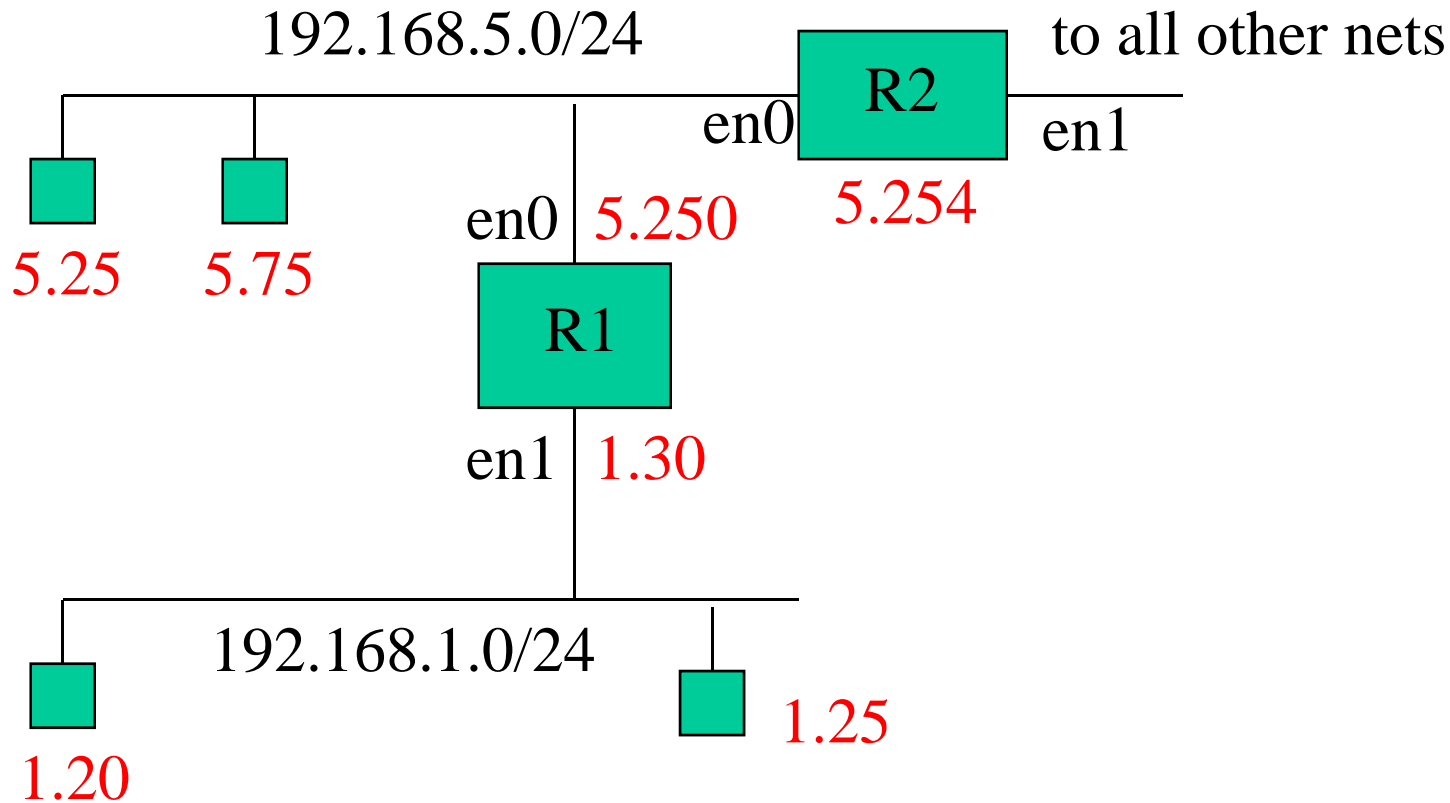
1. Route table is checked.
192.168.1.25/24 matches the 192.168.1.0 entry
2. The next hop is the host itself (192.168.1.20). This means the destination is on the local network.
3. H1 then sends an ARP packet to find the data link address of the destination
4. Once the data link address is found, the packet is sent

Routing Scenario 1

Steps involved in sending a packet from H1 to an address that is on another network:

1. Route table is checked. The destination address matches the default entry in the table.
2. The next hop is 192.168.1.30. This means the destination is on the other side of a router.
3. H1 sends an ARP packet to determine the data link address of the gateway.
4. The packet is sent to the router
5. The router's route table is checked and the packet is sent to the next hop
6. This continues until the packet reaches the final destination

Routing Scenario 2



Routing Scenario 2

Packet from H3 to H4 (same network)

| IP Address | | Hardware Address | |
|------------|------|------------------|------|
| SRC | DEST | SRC | DEST |
| H3 | H4 | H3 | H4 |

Packet from H3 to H1 (Next door network)

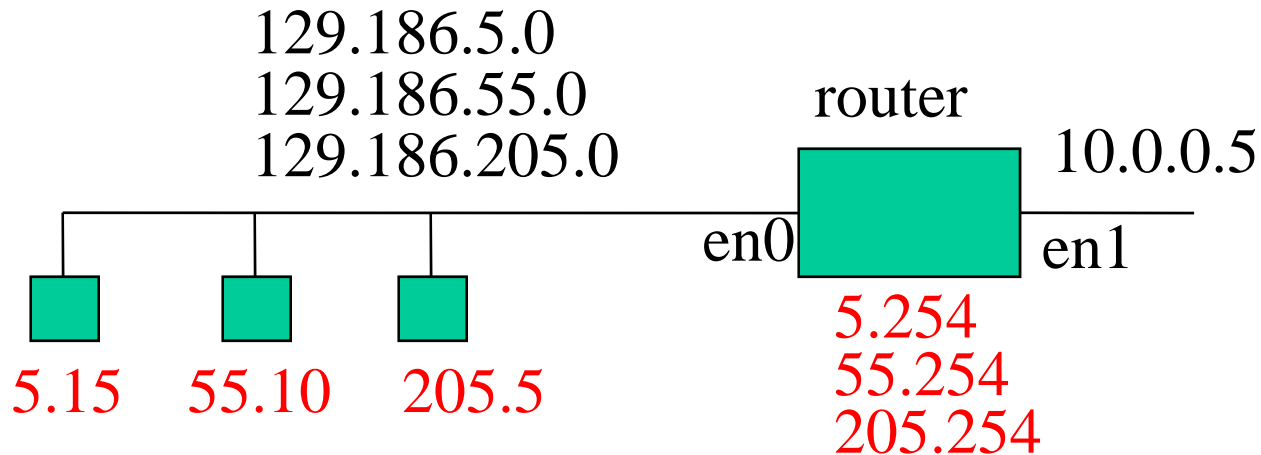
| IP Address | | Hardware Address | |
|------------|------|------------------|----------|
| SRC | DEST | SRC | DEST |
| H3 | H1 | H3 | R1 (EN1) |
| H3 | H1 | R1 (EN0) | H1 |

Packet from H3 to H5 (default network)

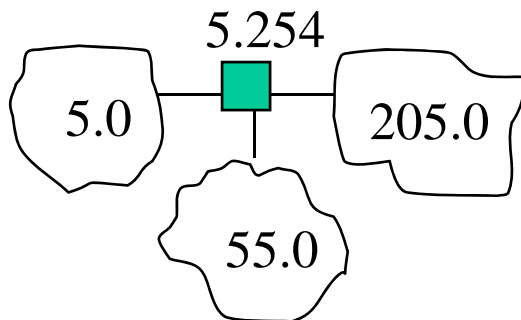
| IP Address | | Hardware Address | |
|------------|------|------------------|----------|
| SRC | DEST | SRC | DEST |
| H3 | H5 | H3 | R2 (EN0) |

Routing Example 3

Sometimes a network can have multiple IP's:



Logically, the network is viewed like this for host 5.15:



| Destination |
|-------------|
| 129.186.5.0 |
| Default |

| Next |
|---------------|
| 129.186.5.15 |
| 129.186.5.254 |

IP Packet Format

| | | | | |
|----------------|----------|------|----------------------|--------|
| VER=4 | IHL | TYPE | TOTAL LENGTH (bytes) | |
| ID | | | FLAG | OFFSET |
| TTL | PROTOCOL | | CHECKSUM | |
| SOURCE IP | | | | |
| DESTINATION IP | | | | |
| OPTION | | | | |
| DATA | | | | |

IP Packet Format

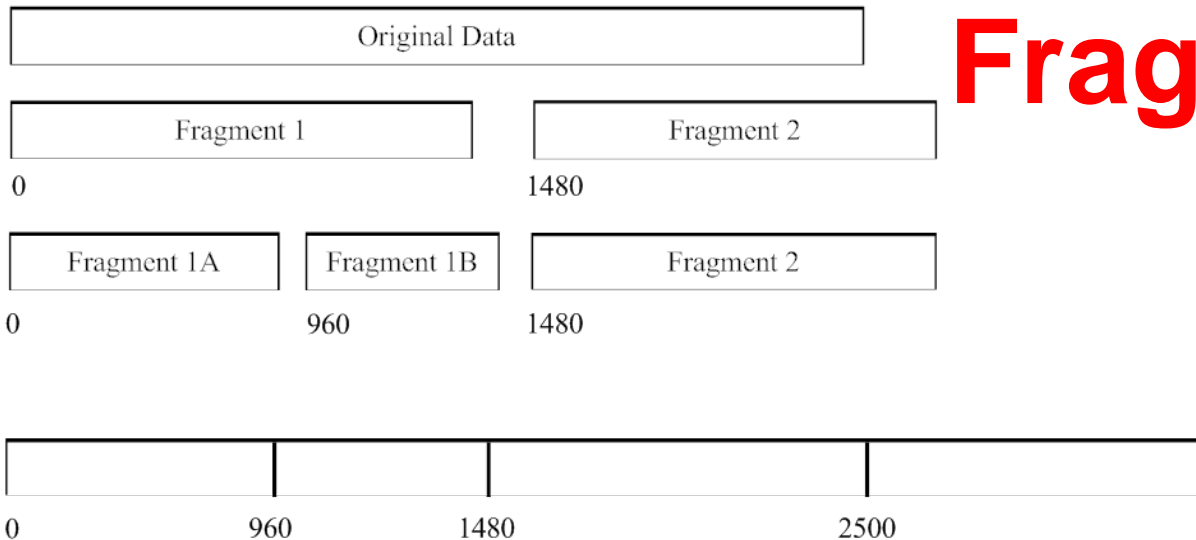
- IHL: header length in words
- Type of service: almost always 0
- Total length (bytes) includes header length.
Max packet size = 2^{11} bytes
- ID: used in fragmentation
- Flag: 0: not used
 D=1: don't fragment
 M=:1 more data. M=0: last packet of fragment
- Offset: #8 bytes
- TTL (time to live): starts at 255 then decrements after each hop
- Checksum: worthless because it must be recalculated after every router due to the TTL decrement

IP Protocol Field

- 1 Internet Control Message Protocol (ICMP)
- 3 Gateway-to-Gateway protocol
- 5 Stream
- 6 Transport Control Protocol (TCP)
- 8 Exterior Gateway Protocol
- 9 Any private interior gateway protocol
- 11 Network voice protocol
- 17 User datagram protocol (UDP)
- 20 Host Monitoring Protocol
- 22 Xerox Network System Internet Datagram Protocol
- 27 Reliable Datagram Protocol
- 28 Internet Reliable Transaction Protocol
- 30 Bulk Data Transfer Protocol
- 61 Any Host Internet Protocol

| Fields | Original Packet | Fragment 1 | Fragment 2 | Fragment 1a | Fragment 1b |
|-----------|-----------------|------------|------------|-------------|-------------|
| Ver/HLEN | 4/5 | 4/5 | 4/5 | 4/5 | 4/5 |
| Type | 0 | 0 | 0 | 0 | 0 |
| Length | 2540 | 1500 | 1060 | 1000 | 560 |
| ID | 2356 | 2356 | 2356 | 2356 | 2356 |
| Flags | 0 | 0 0 1 | 0 0 0 | 0 0 1 | 0 0 1 |
| Offset | 0 | 0 | 185 | 0 | 120 |
| TTL | 150 | computed | computed | computed | computed |
| Protocol | TCP | TCP | TCP | TCP | TCP |
| Checksum | computed | computed | computed | computed | computed |
| Source IP | IP1 | IP1 | IP1 | IP1 | IP1 |
| Dest IP | IP2 | IP2 | IP2 | IP2 | IP2 |
| Data Len | 2500 | 1480 | 1020 | 960 | 520 |

Fragmentation



Machine Address Resolution

- We now have the IP address for the destination, but we need to find the datalink address of the destination.
- There is no assigned relationship between the datalink address and the IP address.
- We need a protocol to query the network to find the data link address of a host with a given IP address.
- This protocol is called Address Resolution Protocol (ARP). The ARP protocol uses the datalink broadcast address to query all hosts on the network. The host whose IP address matches the requested address will respond with a packet that contains its data link address.

ARP Packet Format

| | | |
|-----------------------|------|-----------------------|
| HW type | | Protocol type |
| HLEN | PLEN | Operation |
| Sender HA (bytes 0-3) | | |
| Sender HA (4-5) | | Sender IP (bytes 0-1) |
| Sender IP (bytes 2-3) | | Target HA (bytes 0-1) |
| Target HA (bytes 2-5) | | |
| Target IP (bytes 0-3) | | |

ARP Packet Format

- Hardware type 1 = Ethernet
- Protocol Type 0x800 = IP
- HLEN = 6
- PLEN = 4
- Operation
 - 1 = ARP Request
 - 2 = ARP Response
 - 3 = RARP Request
 - 4 = RARP Reply

ARP Protocol

- A station that needs to find a datalink address will create an ARP packet and will fill in the sender IP and HA fields with its IP address and Hardware address. It will place the IP address of the target machine in the target IP field. The station will also fill in the first 5 fields. The ARP packet is then used as the data field in an Ethernet packet. This Ethernet packet has the broadcast address in the destination field.

ARP

- The packet is then sent out on the network. Since it is a broadcast packet all stations will receive the packet. The station whose IP address matches the target IP address will create a new ARP packet to send back to the sender. The target machine will put his address into the sender fields and will put the requestors address into the target fields. The ARP packet will then be sent as data in an Ethernet packet whose destination address is the requesting station.

ARP

- The help cut down on the traffic stations on the network can use an internal ARP table to cache ARP responses and also to cache information from ARP requests. For example when a station receives an ARP request, even if the target IP address does not match the station can store the IP address and Ethernet address found in the sender fields.

ARP

- The entries in the table have a short life. This enables changes in the mapping between IP address and Hardware address without clearing the table.
- The RARP protocol is used by diskless workstations to find their IP address from a server. They only know their own Ethernet address.

ICMP

Internet Control Message Protocol

- Designed as error control
- Provides a means for transferring messages between hosts
- Examples for use:
 - When a datagram cannot reach its destination
 - When a gateway can direct the host to send traffic on a shorter route
 - Ping

ICMP Packet Format

| | | | | |
|----------------|----------|----------|----------------------|--------|
| VER=4 | IHL | TYPE | TOTAL LENGTH (bytes) | |
| ID | | | FLAG | OFFSET |
| TTL | PROTOCOL | CHECKSUM | | |
| SOURCE IP | | | | |
| DESTINATION IP | | | | |
| Type | Code | Checksum | | |
| Parameter | | | | |
| Information | | | | |

ICMP Packet Format

- ICMP packets are carried within the data of an IP packet
- Fields:
 - Type (8 bits): message type
 - Code (8 bits): message sub-type
 - Checksum (16 bits)
 - Parameter (32 bits)
 - Information (variable)

ICMP Message Types

- 0 Echo Reply
- 3 Destination Unreachable
- 4 Source Quench
- 5 Redirect
- 8 Echo
- 12 Parameter Problem
- 13 Timestamp
- 14 Timestamp Reply
- 15 Information Request
- 16 Information Reply
- 17 Address Mask Request
- 18 Address Mask Reply

ICMP Echo (Ping)

- Type = 8 (echo)
Type = 0 (reply)
- Code = 0
- Parameter
 - ID number (2 bytes)
 - Sequence number (2 bytes)
- Optional Data

Note: the optional data field of ping has been used in the past for tunneling information through a firewall

ICMP Destination Unreachable

- Type = 3
- Code:
 - 0 Network Unreachable
 - 1 Host Unreachable
 - 2 Protocol Unreachable
 - 3 Port Unreachable
 - 4 Fragmentation needed and DF set
 - 5 Source Route Failed
- Parameter = 0
- Data = IP header + first 8 bytes of datagram

ICMP Source Quench

- Type = 4
- Code = 0
- Parameter = 0
- Data = IP header + first 8 bytes of datagram
- Sent when a packet arrives too quickly for a host to process. The packet is discarded.
- A host receiving a source quench message will slow down its rate of transmission until it no longer receives source quench messages. Then it will slowly increase its rate as long as no more source quench messages are received.

ICMP Redirect

- Type = 5

Code:

- 0 Redirect for the NET
- 1 Redirect for the Host
- 2 Redirect for type of service and net
- 3 Redirect for type of service and host

Parameter = gateway IP address

Data = IP header + first 8 bytes of datagram

- Sent when a gateway detects a host using a non-optimum route
- Original packet is not dropped
- If the host does not update its route table and continues using the non-optimum route, an ICMP redirect storm can occur

ICMP Time Exceeded

- Type = 11
- Code:
 - 0 TTL (time to live) count exceeded
 - 1 Fragment reassembly time exceeded
- Parameter = 0
- Data = IP header + first 8 bytes of datagram

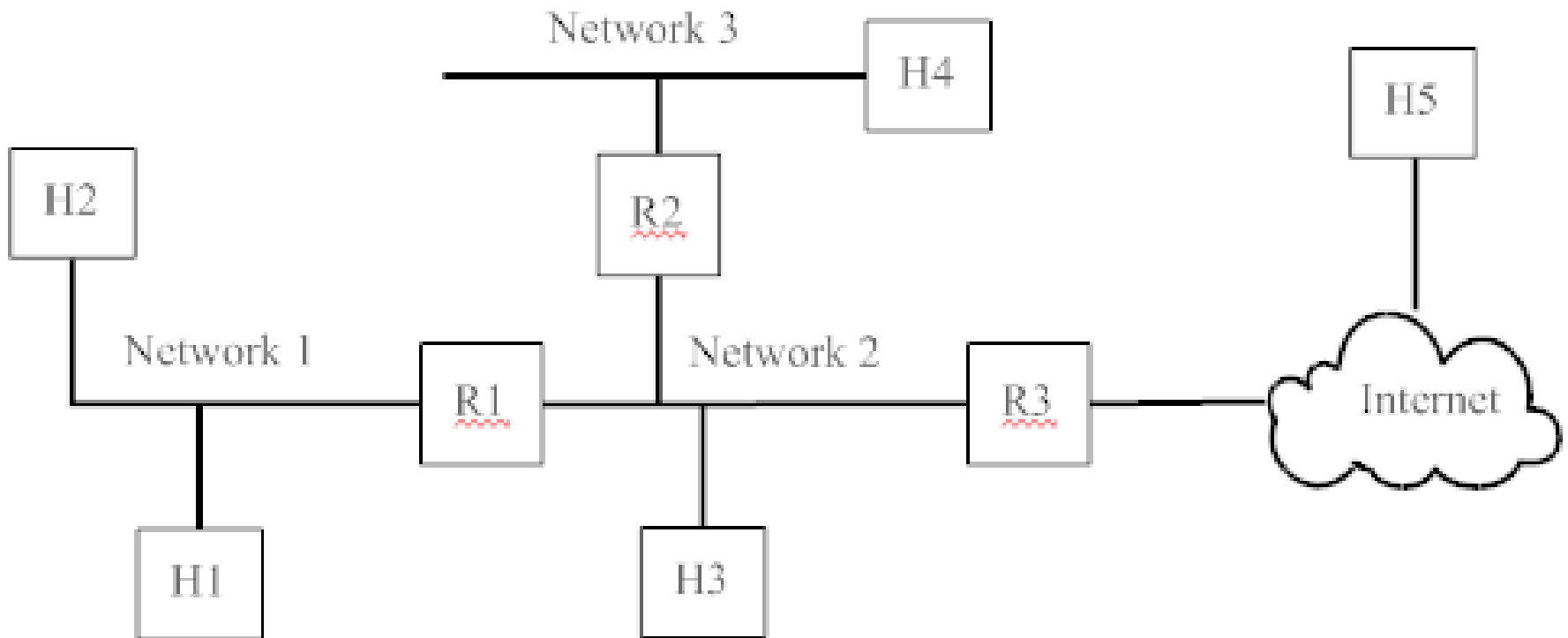
ICMP Parameter Problem

- Type = 12
- Code = 0
- Parameter (8 bits) = pointer to error
- Data = IP header + first 8 bytes of datagram
- Sent when a gateway or host finds a problem with the IP header.
- The pointer identifies the octet in the header that caused the problem

ICMP Timestamp

- Type = 13 (echo)
Type = 14 (reply)
- Code = 0
- Parameter:
 - ID number (2 bytes)
 - Sequence number (2 bytes)
- Originate timestamp
- Receive timestamp (reply only)
- Transmit timestamp (reply only)

Putting it all together



Route tables

Route Table H1 & H2

| Destination | Next Hop |
|-------------|-----------|
| Network 1 | Me |
| default | <u>R1</u> |

Route Table H4

| Destination | Next Hop |
|-------------|-----------|
| Network 3 | Me |
| default | <u>R2</u> |

Route Table H3

| Destination | Next Hop |
|-------------|-----------|
| Network 1 | <u>R1</u> |
| Network 2 | H3 |
| Network 3 | <u>R2</u> |
| default | <u>R3</u> |

Route Table Router R1

| Destination | Next Hop | Interface |
|-------------|-----------|--------------|
| Network 1 | <u>R1</u> | <u>Int 1</u> |
| Network 2 | <u>R1</u> | <u>Int 2</u> |
| Network 3 | <u>R2</u> | <u>Int 2</u> |
| default | <u>R3</u> | <u>Int 2</u> |

Route Table Router R2

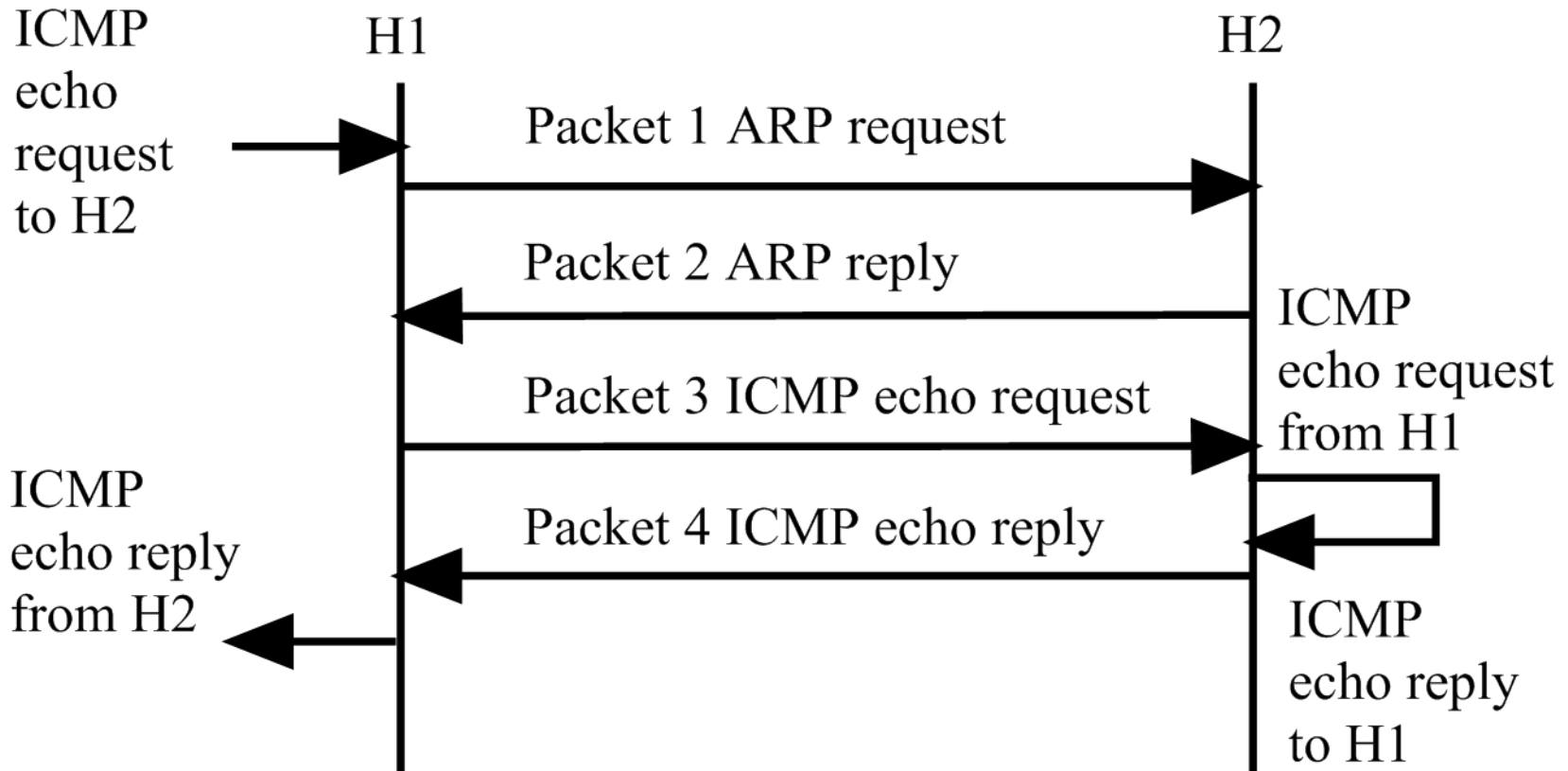
| Destination | Next Hop | Interface |
|-------------|-----------|--------------|
| Network 1 | <u>R1</u> | <u>Int 1</u> |
| Network 2 | <u>R2</u> | <u>Int 1</u> |
| Network 3 | <u>R2</u> | <u>Int 2</u> |
| default | <u>R3</u> | <u>Int 1</u> |

Route Table Router R3

| Destination | Next Hop | Interface |
|-------------|-----------|--------------|
| Network 1 | <u>R1</u> | <u>Int 1</u> |
| Network 2 | <u>R3</u> | <u>Int 1</u> |
| Network 3 | <u>R2</u> | <u>Int 1</u> |
| default | Next Hop | <u>Int 2</u> |

Scenario 1 (H1 to H2)

Network 1



Scenario 1 (H1 to H2)

| Packet | Hardware Addresses | | IP Addresses | | Payload |
|--------|--------------------|-----|--------------|-----|---------|
| | DST | SRC | DST | SRC | |
| 1 | Broadcast | H1 | N/A | N/A | ARP |
| 2 | H1 | H2 | N/A | N/A | ARP |
| 3 | H2 | H1 | H2 | H1 | ICMP |
| 4 | H1 | H2 | H1 | H2 | ICMP |

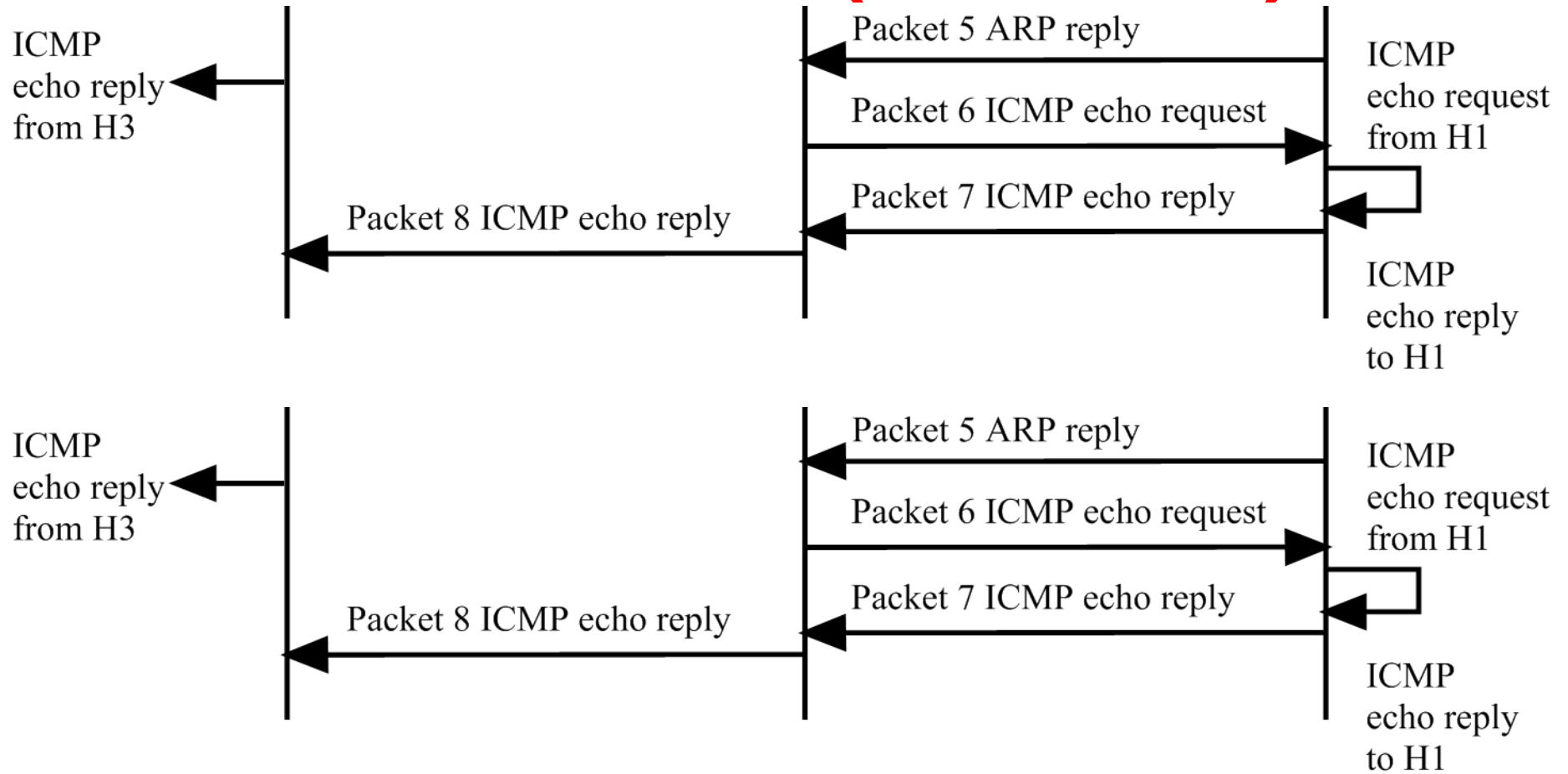
ARP table for H1

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P2 | H2 | H2 |

ARP table for H2

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P1 | H1 | H1 |

Scenario 2 (H1 to H3)



Scenario 2 (H1 to H3)

| Packet | Hardware Addresses | | IP Addresses | | Payload |
|--------|--------------------|------------|--------------|-----|---------|
| | DST | SRC | DST | SRC | |
| 1 | Broadcast | H1 | N/A | N/A | ARP |
| 2 | H1 | R1 (Int 1) | N/A | N/A | ARP |
| 3 | R1 (Int 1) | H1 | H3 | H1 | ICMP |
| 4 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 5 | R1 (Int 2) | H3 | N/A | N/A | ARP |
| 6 | H3 | R1 (Int 2) | H3 | H1 | ICMP |
| 7 | R1 (Int 2) | H3 | H1 | H3 | ICMP |
| 8 | H1 | R1 (Int 1) | H1 | H3 | ICMP |

ARP table for H1

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | H2 | H2 |
| After P2 | R1 | R1 (Int 1) |

ARP table for H3

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P4 | R1 | R1 (Int 2) |

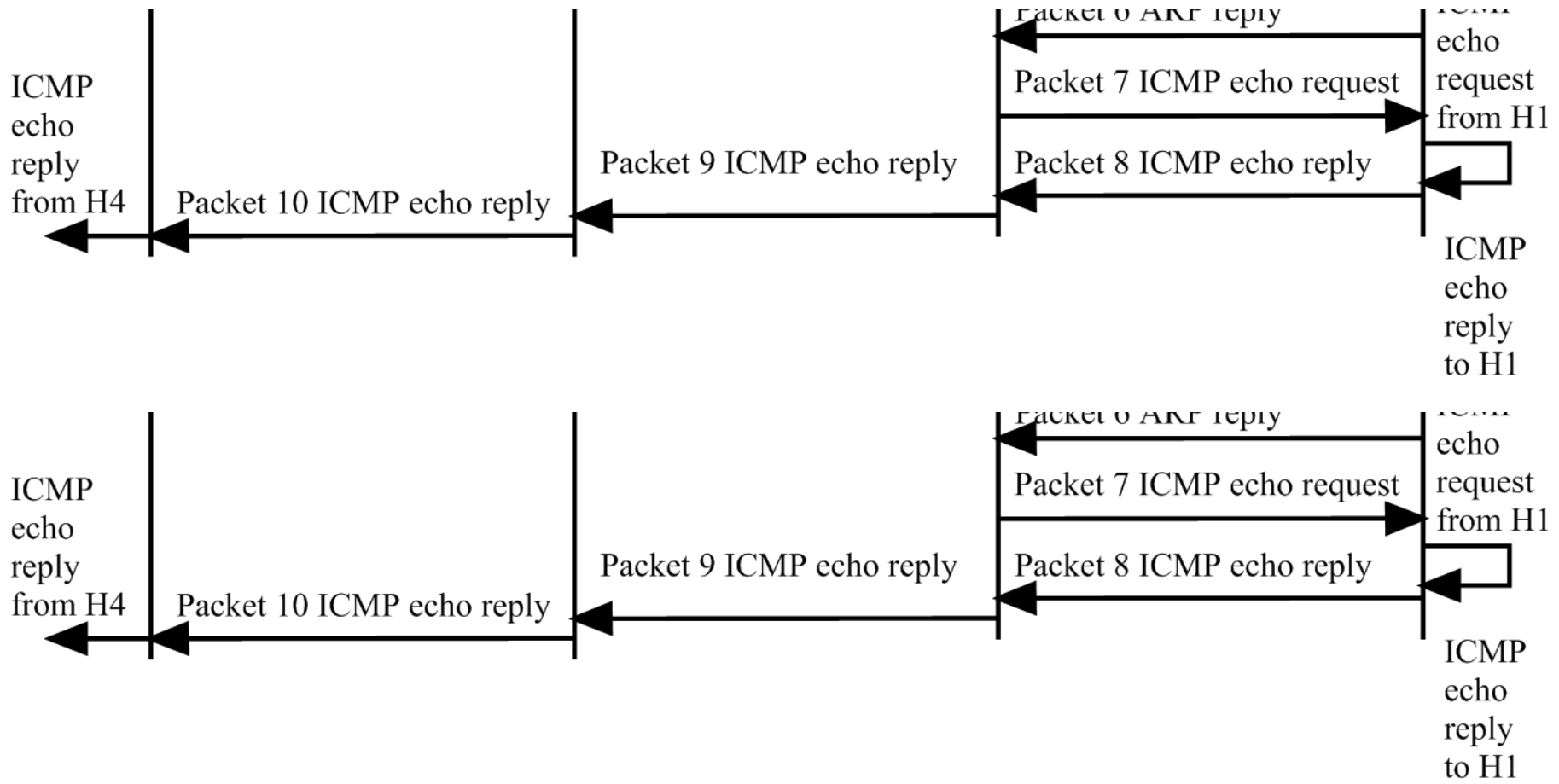
ARP table for R1 (int 1)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P1 | H1 | H1 |

ARP table for R1 (int 2)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P5 | H3 | H3 |

Scenario 3 (H1 to H4)



Scenario 3 (H1 to H4)

| Packet | Hardware Addresses | | IP Addresses | | Payload |
|--------|--------------------|------------|--------------|-----|---------|
| | DST | SRC | DST | SRC | |
| 1 | R1 (Int 1) | H1 | H4 | H1 | ICMP |
| 2 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 3 | R1 (Int 2) | R2 (Int 1) | N/A | N/A | ARP |
| 4 | R2 (Int 1) | R1 (Int 2) | H4 | H1 | ICMP |
| 5 | Broadcast | R2 (Int 2) | N/A | N/A | ARP |
| 6 | R2 (Int 2) | H4 | N/A | N/A | ARP |
| 7 | H4 | R2 (Int 2) | H4 | H1 | ICMP |
| 8 | R2 (Int 2) | H4 | H1 | H4 | ICMP |
| 9 | R1 (Int 2) | R2 (Int 1) | H1 | H4 | ICMP |
| 10 | H1 | R1 (Int 1) | H1 | H4 | ICMP |

ARP table for H1

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H2 | H2 |
| | R1 | R1 (Int 1) |

ARP table for H4

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P5 | R2 | R2 (Int 2) |

ARP table for R1 (int 1)

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H1 | H1 |
| | | |

ARP table for R1 (int 2)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | H3 | H3 |
| After P3 | R2 | R2 (Int 1) |

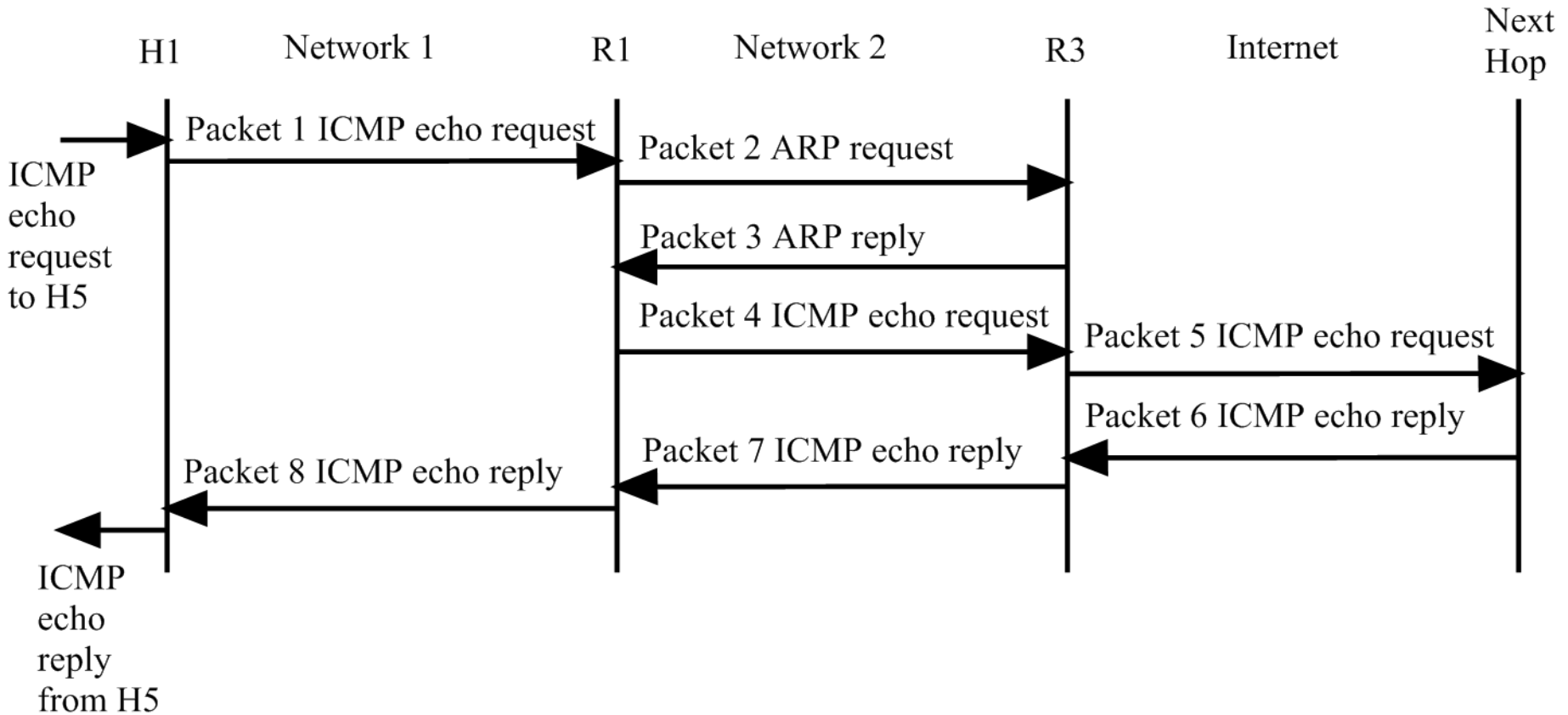
ARP table for R2 (int 1)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P2 | R1 | R1 (Int 2) |

ARP table for R2 (int 2)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P6 | H4 | H4 |

Scenario 4 (H1 to H5)



Scenario 4 (H1 to H5)

| Packet | Hardware Addresses | | IP Addresses | | Payload |
|--------|--------------------|------------|--------------|-----|---------|
| | DST | SRC | DST | SRC | |
| 1 | R1 (Int 1) | H1 | H5 | H1 | ICMP |
| 2 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 3 | R1 (Int 2) | R3 (Int 1) | N/A | N/A | ARP |
| 4 | R3 (Int 1) | R1 (Int 2) | H5 | H1 | ICMP |
| 5 | Next hop | R3 (Int 2) | H5 | H1 | ICMP |
| 6 | R3 (Int 2) | Next hop | H1 | H5 | ICMP |
| 7 | R1 (Int 2) | R3 (Int 1) | H1 | H5 | ICMP |
| 8 | H1 | R1 (Int 1) | H1 | H5 | ICMP |

ARP table for H1

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H2 | H2 |
| | R1 | R1 (Int 1) |

ARP table for H4

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | Empty | Empty |
| | R2 | R2 (Int 2) |

ARP table for R1 (int 1)

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H1 | H1 |
| | | |

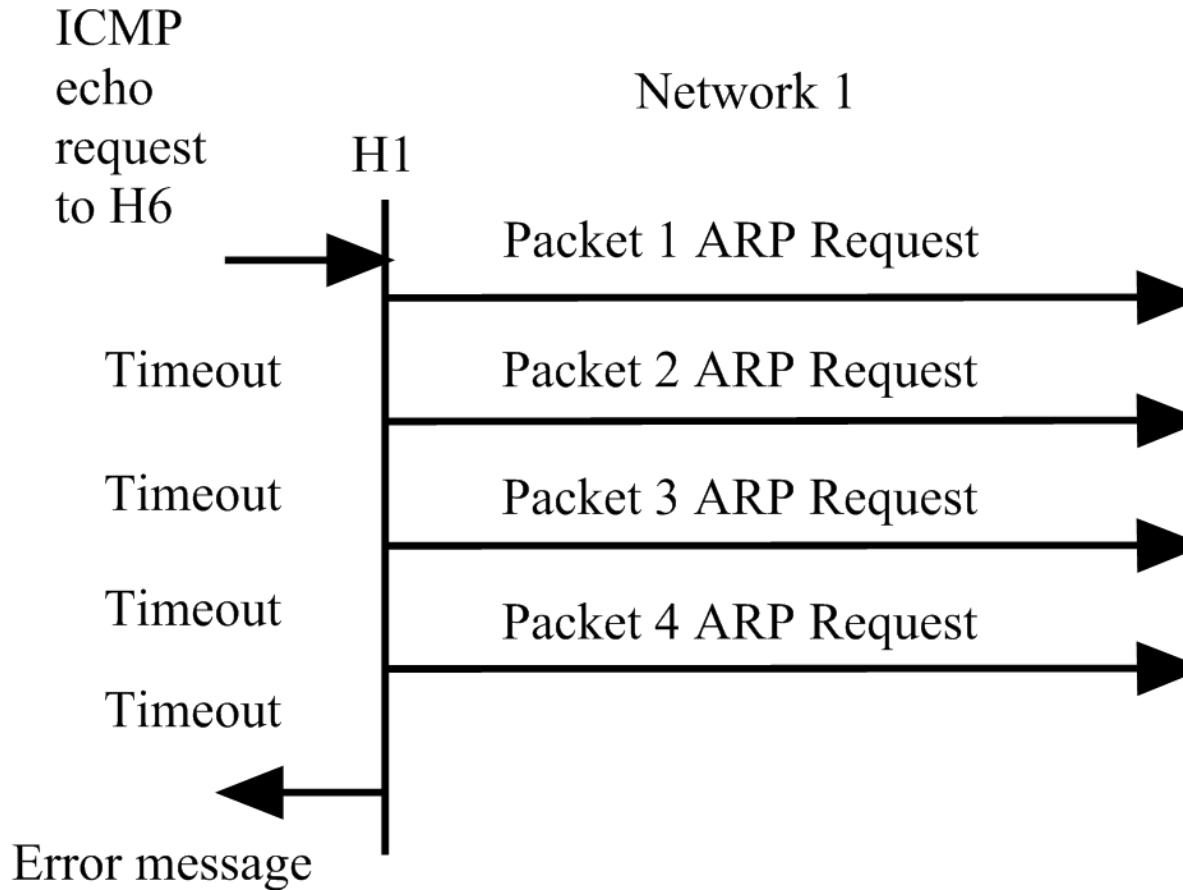
ARP table for R1 (int 2)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | H3 | H3 |
| | R2 | R2 (Int 1) |
| After P3 | R3 | R3 (Int 1) |

ARP table for R3 (int 1)

| Time | Destination | HW Address |
|----------|-------------|------------|
| Start | Empty | Empty |
| After P2 | R1 | R1 (Int 2) |

Scenario 5 (H1 to no host on net 1)



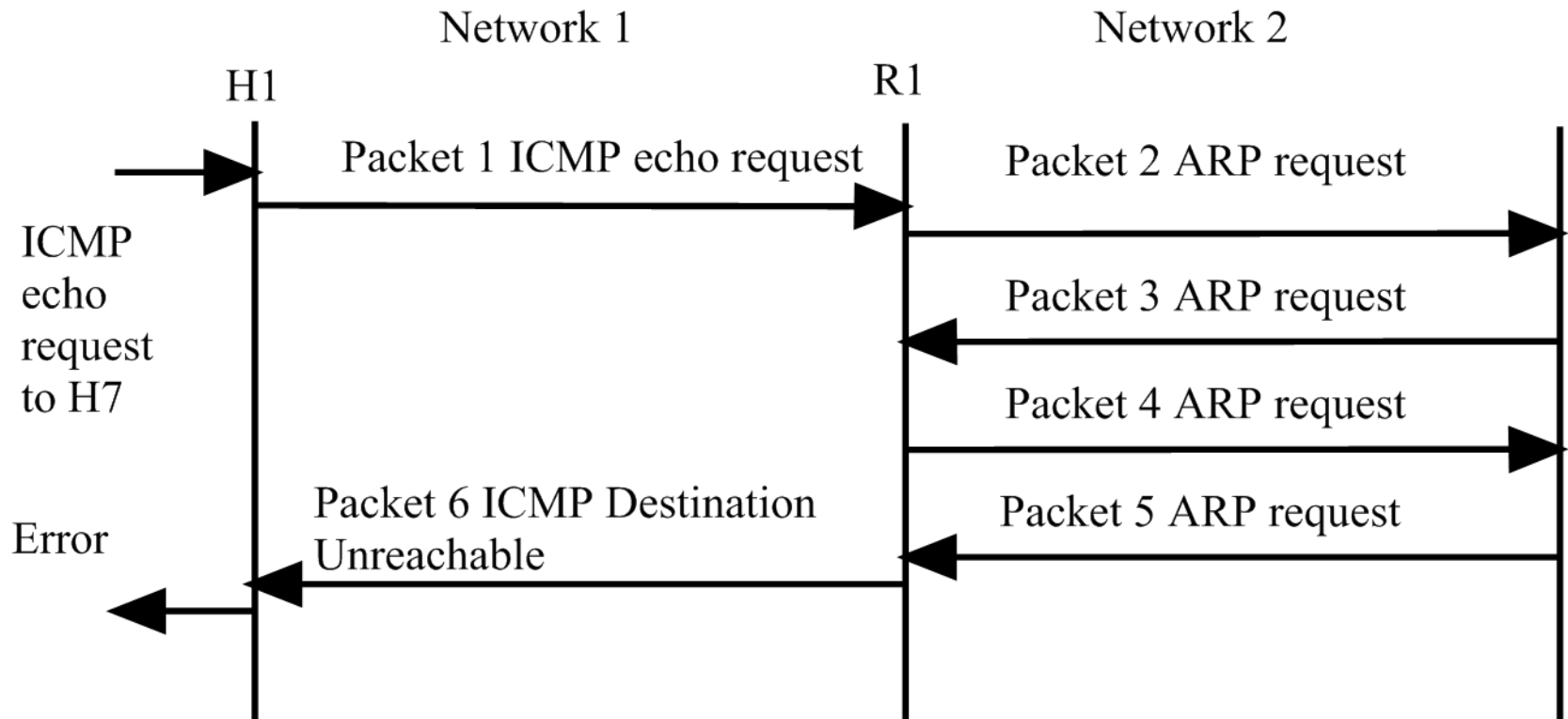
Scenario 5 (H1 to no host on net 1)

| Packet | Hardware Addresses | | IP Addresses | | Payload |
|--------|--------------------|-----|--------------|-----|---------|
| | DST | SRC | DST | SRC | |
| 1 | Broadcast | H1 | N/A | N/A | ARP |
| 2 | Broadcast | H1 | N/A | N/A | ARP |
| 3 | Broadcast | H1 | N/A | N/A | ARP |
| 4 | Broadcast | H1 | N/A | N/A | ARP |

ARP table for H1

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H2 | H2 |
| | R1 | R1 (Int 1) |

Scenario 6 (H1 to no host on net 2)



Scenario 6 (H1 to no host on net 2)

| Packet | Hardware Addresses | | IP Addresses | | Payload |
|--------|--------------------|------------|--------------|-----|---------|
| | DST | SRC | DST | SRC | |
| 1 | R1 (Int 1) | H1 | H7 | H1 | ICMP |
| 2 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 3 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 4 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 5 | Broadcast | R1 (Int 2) | N/A | N/A | ARP |
| 6 | H1 | R1 (Int 1) | H1 | R1 | ICMP |

ARP table for H1

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H2 | H2 |
| | R1 | R1 (Int 1) |

ARP table for R1 (int 2)

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H3 | H3 |
| | R2 | R2 (Int 1) |
| | R3 | R3 (Int 1) |

ARP table for R1 (int 1)

| Time | Destination | HW Address |
|-------|-------------|------------|
| Start | H1 | H1 |

Header Based

- There have been some IP header attacks.
- Most famous is the ping of death
- Most have been fixed
- Fewer ARP and ICMP header attacks

Protocol Based

- Even though the IP protocol is simple, the routing is complex.
- There are a large number of protocol based attacks involving sending packets the confuse the receiver or interjects packets into the receiver.
- They work because there is no authentication of the sender and receiver.

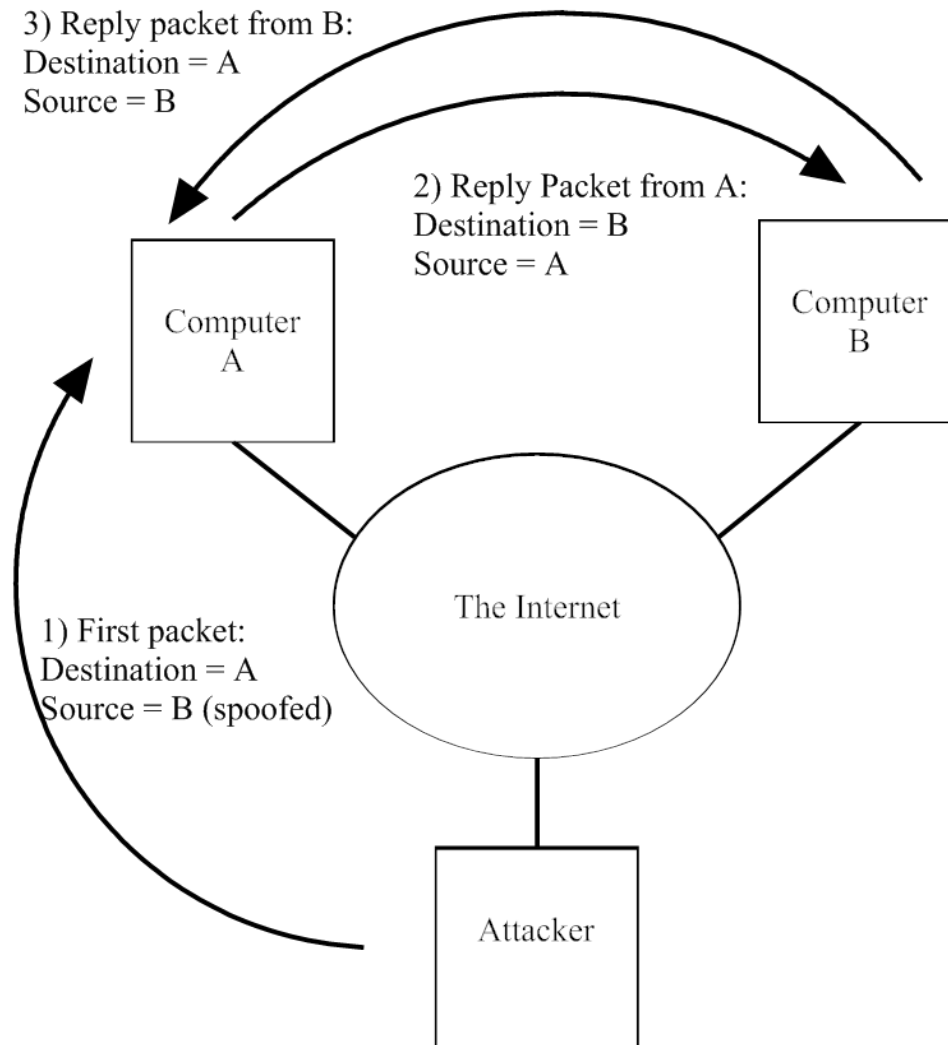
Protocol Based

- ICMP:
 - Using redirect
- ARP
 - ARP cache poisoning (better classified as an authentication attack)

Authentication Based

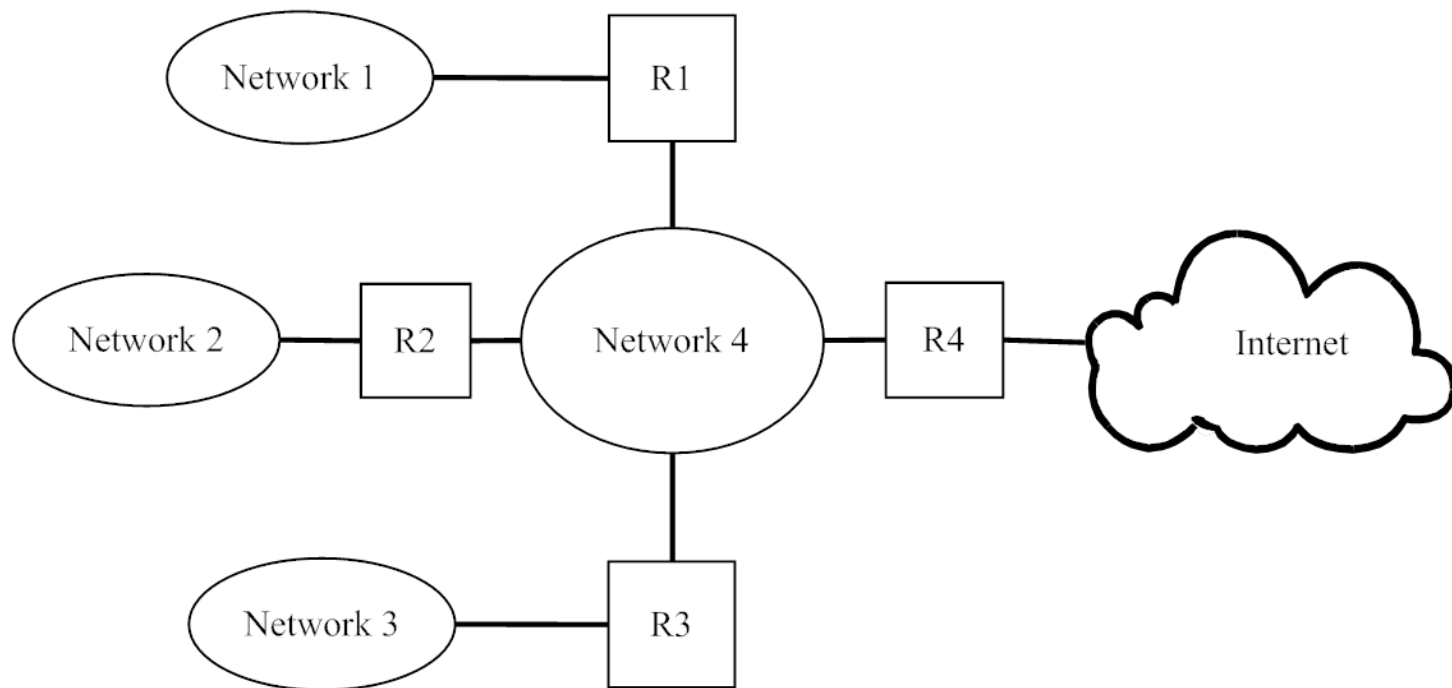
- This is a big problem, since we often use the IP address as authentication.
- IP
 - Address spoofing is very difficult to implement unless you can “see” the traffic
 - IP address spoofing is very hard to stop if the attacker is in the right place.
- ARP
- DHCP

IP Spoofing



IP Spoofing Mitigation

- Check source IP address before allowing packet into the Internet



Traffic Based

- Sniffing is a problem
- Broadcast traffic can cause flooding
- Flooding is a problem with unicast packets also. They can cause routers to hosts to quit.

ARP Broadcast Flood

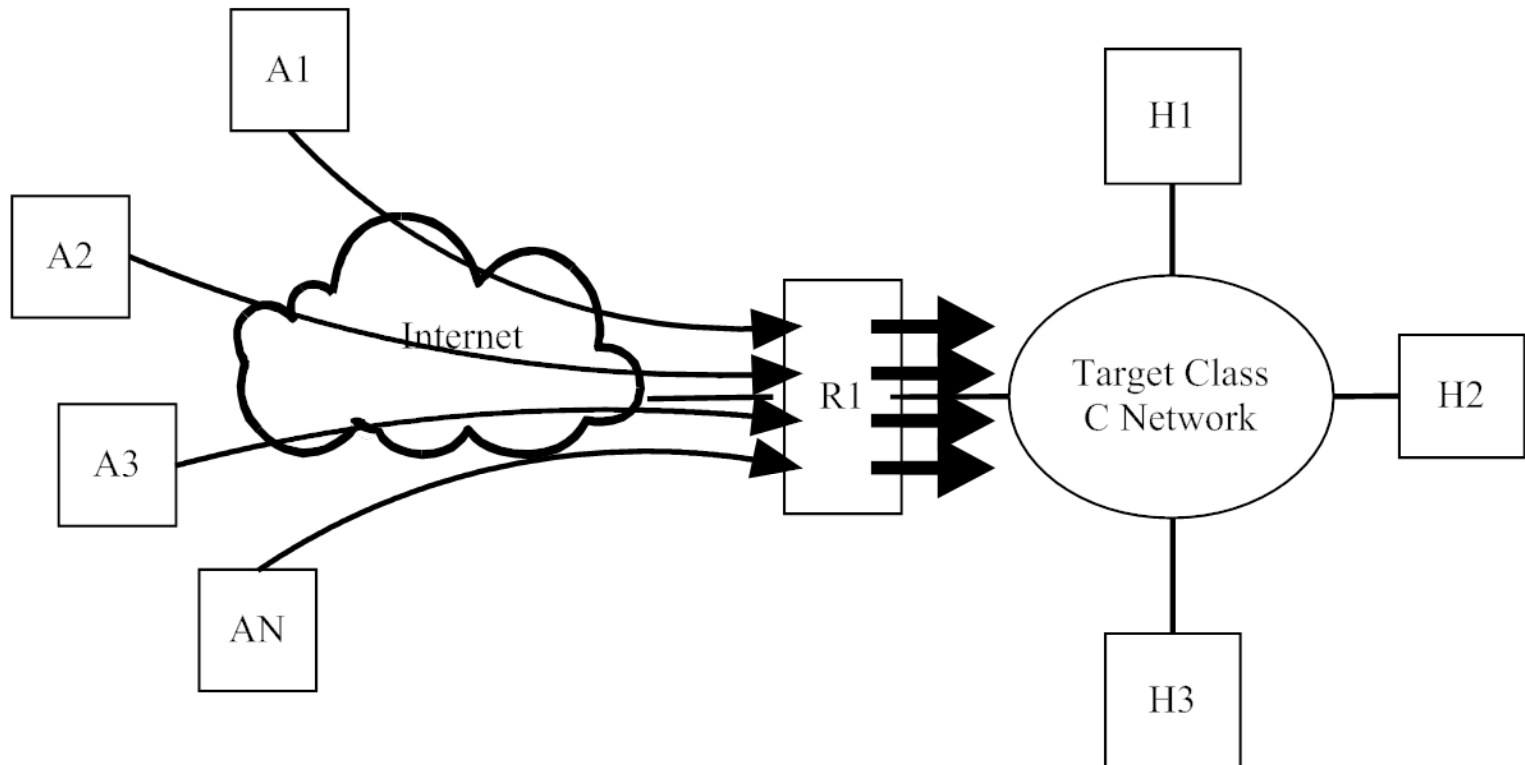


Figure 6.23 ARP Broadcast Flood Attack

BOOTP

- Bootstrap Protocol
- Allows a networked machine to automatically acquire an IP address
- Client-server program
- Server has configuration file which contains a one-to-one mapping between the hardware address of the client and an IP address
- Used for networked laser printers and other diskless machines

BOOTP

- BOOTP server provides client with:
 - IP address
 - Subnet mask
 - IP address of a router
 - IP address of a nameserver

BOOTP

Sample configuration for a printer

hp255:\

:hn:ht=ether:vm=rfc1048:\



Hardware address

:ha=0800094ce9f5:\



IP

:ip=129.186.5.7:\



Netmask

:sm=255.255.255.0:\



Gateway

:gw=129.186.5.254:\



Logging device

:lg=129.186.5.2:\

:T144="hp.printer":

BOOTP Protocol

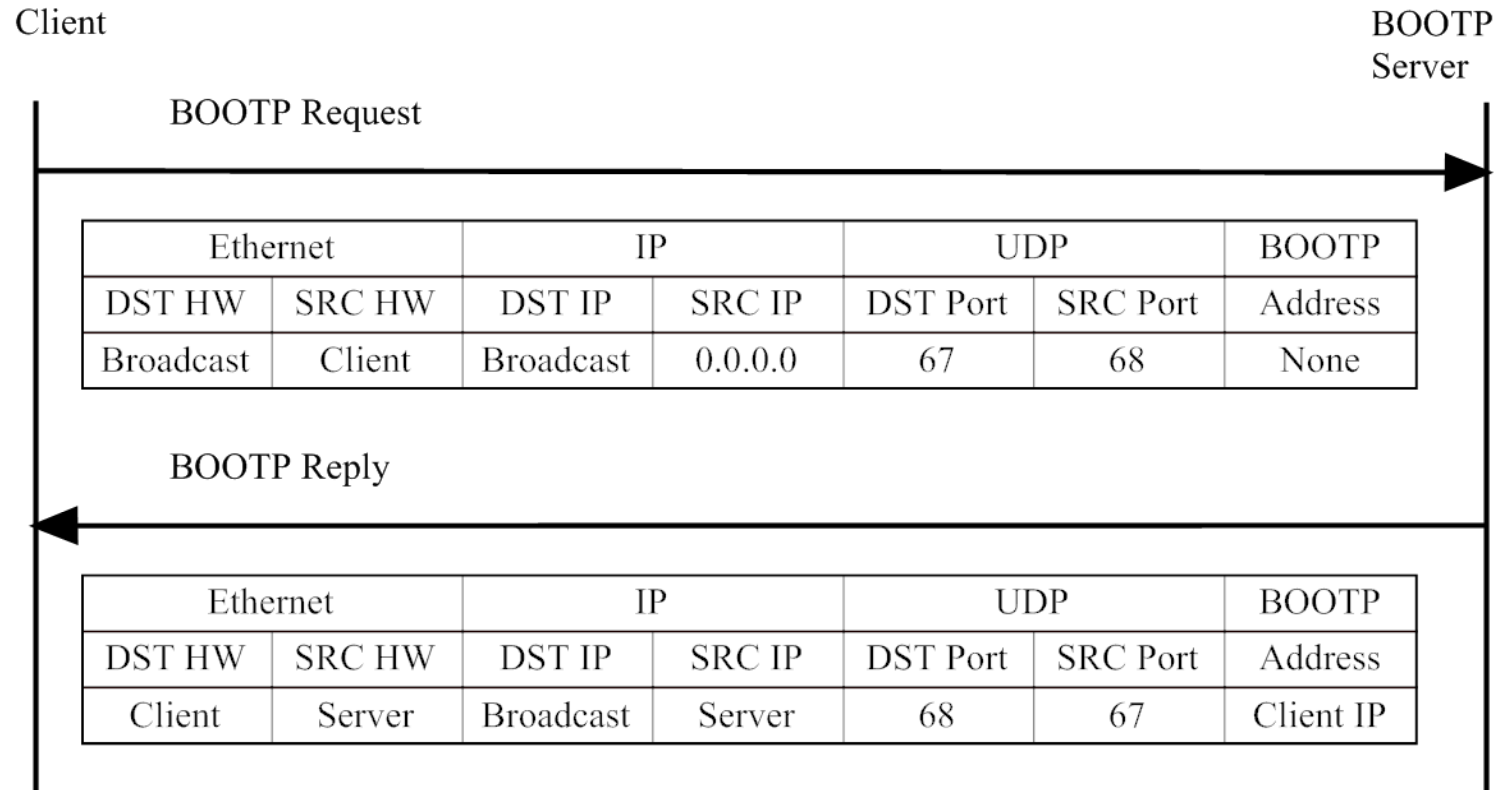
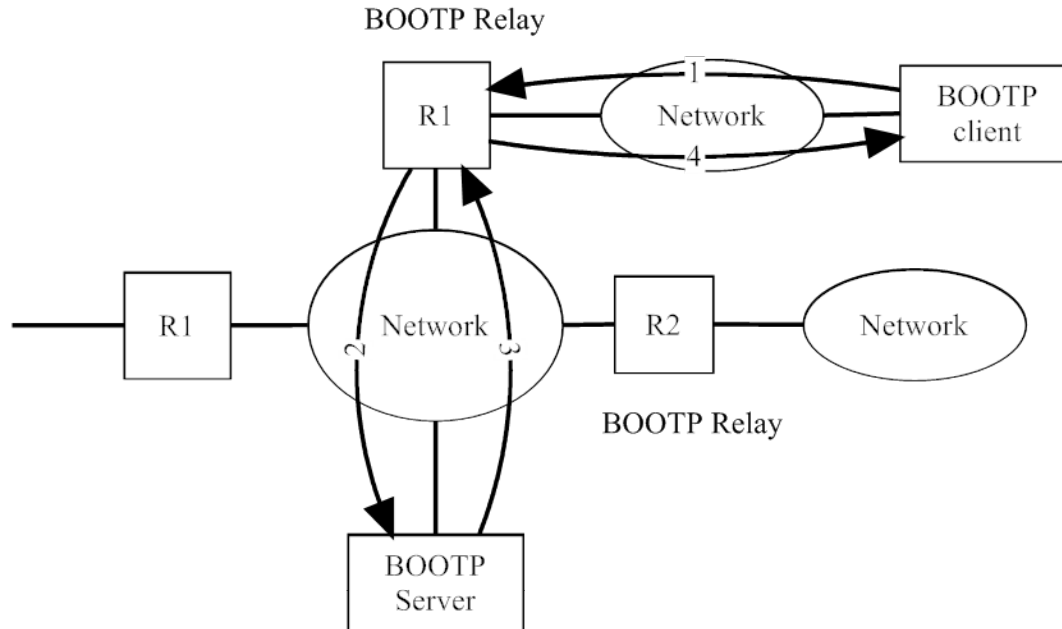


Figure 6.24 BOOTP Protocol

BOOTP

- Note that the client must broadcast it's request, since it does not know who the local router is
- The server cannot use ARP to determine the client's hardware address, so it gets it from the client's request packet
- BOOTP relay
 - Used when client and server are on different subnets
 - Relay receives requests, appends its address, sends requests to server
 - Server replies to relay who then replies to client

BOOTP Relay



| | Ethernet | | IP | | UDP | | BOOTP |
|--------|-----------|--------|-----------|---------|----------|----------|-----------|
| Packet | DST HW | SRC HW | DST IP | SRC IP | DST Port | SRC Port | Address |
| 1 | Broadcast | Client | Broadcast | 0.0.0.0 | 67 | 68 | none |
| 2 | Server | Relay | Server | Relay | 67 | 68 | none |
| 3 | Relay | Server | Relay | Server | 68 | 67 | Client IP |
| 4 | Client | Relay | Broadcast | Relay | 68 | 67 | Client IP |

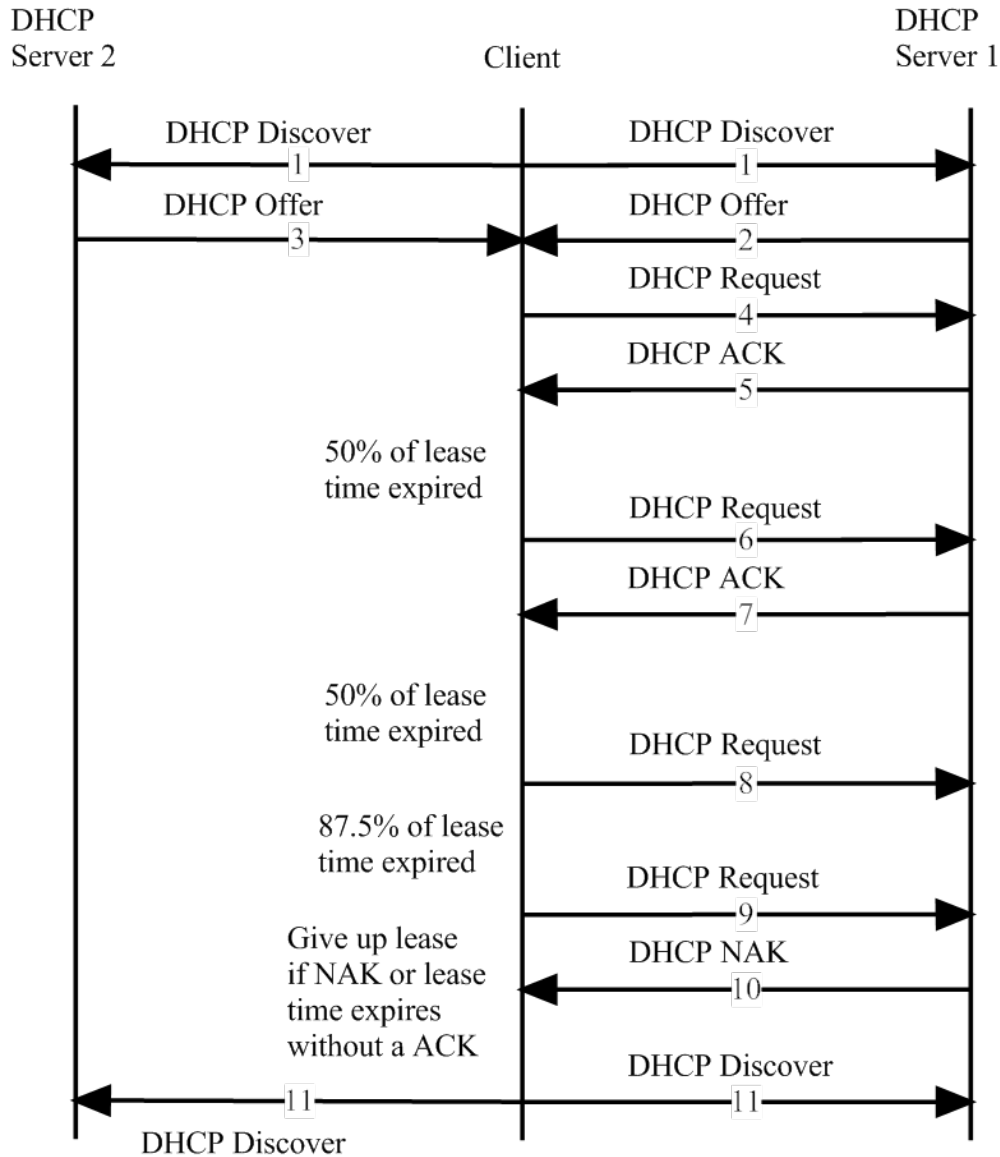
DHCP

- Dynamic Host Configuration Protocol
- An enhancement to BOOTP
- Leases IP addresses to hosts requesting an address
- Dynamic leases (not a one-to-one mapping)

DHCP

- Two databases for each DHCP server:
 - Static IP pool (like bootp)
 - Dynamic pool
- Server checks static pool for match before dynamic pool
- Dynamic pool
 - Addresses are temporary (default lease = 1 hr)
 - After lease expires, client must ask for a renewal
 - If renewal is rejected, client must give up the IP address

DHCP Operation



DHCP Operation

| Packet | Ethernet | | IP | | UDP | | DHCP |
|--------|-----------|----------|-----------|----------|----------|----------|----------|
| | DST HW | SRC HW | DST IP | SRC IP | DST Port | SRC Port | |
| 1 | Broadcast | Client | Broadcast | 0.0.0.0 | 67 | 68 | Discover |
| 2 | Client | Server 1 | Broadcast | Server 1 | 68 | 67 | Offer |
| 3 | Client | Server 2 | Broadcast | Server 2 | 68 | 67 | Offer |
| 4 | Server 1 | Client | Server 1 | 0.0.0.0 | 67 | 68 | Request |
| 5 | Client | Server 1 | Broadcast | Server 1 | 68 | 67 | ACK |
| 6 | Server 1 | Client | Server 1 | Client | 67 | 68 | Request |
| 7 | Client | Server 1 | Broadcast | Server 1 | 68 | 67 | ACK |
| 8 | Server 1 | Client | Server 1 | Client | 67 | 68 | Request |
| 9 | Server 1 | Client | Server 1 | Client | 67 | 68 | Request |
| 10 | Client | Server 1 | Broadcast | Server 1 | 68 | 67 | NAK |
| 11 | Broadcast | Client | Broadcast | 0.0.0.0 | 67 | 68 | Discover |

DHCP Operation

- Client sends DHCP discover up to 5 times at 2 sec intervals until the DHCP offer is received. If it fails, it will try again after 5 minutes
- The DHCP offer contains the lease time
- After the offer is sent, the server locks that IP address
- Client chooses one offer and sends a DHCP request to the server. (If there are multiple servers, the client may receive more than one offer)
- Server responds with DHCP ack, and creates the binding between the HW address and IP address
- Client can now use the IP address

DHCP Operation

- Before 50% of the lease has expired, the client must send another DHCP request to ask for renewal
- If the server responds with a DHCP ack, the client resets its timer
- If the server responds with a DHCP nak, the client must immediately stop using the IP address and find another server
- If the server does not respond, the client sends another DHCP request after 87.5% of lease has expired
- If the lease expires before the server responds, the client gives up the IP address
- Client sends DHCP release to give up IP address (can do this at any time)

DHCP Packet Format

| | | | |
|--|---------------|---------------|-----------|
| Op Code | Hardware type | Hardware Len | Hop Count |
| ID | | | |
| Number of Seconds | | Flag + Unused | |
| Client IP Address | | | |
| Client IP Address (used in reply packet) | | | |
| Server IP Address | | | |
| Gateway IP Address | | | |
| Client Hardware Address (16 bytes) | | | |
| Server Name (64 bytes) | | | |
| Boot File Name (128 bytes) | | | |
| Options (contains DHCP message types) | | | |

Figure 6.27 DHCP/BOOTP Header Format

Header based attacks

- Very simple header, no attacks

Protocol / Auth based attacks

- BOOTP is a simple protocol
 - An attacker could try and give false information causing a host to get the wrong IP address.
(really an authentication attack)
- DHCP is more complex
 - An attacker could give false information
 - An attacker could reserve all of the addresses
 - An attacker could send fake release packets

Traffic Based

- Sniffing is not an issue since the information is not a secret
- Not any real good flooding based attacks due to the slow nature of the protocol

Ipv6 Topics

- Overview
- Packet Format
- ICMP V6

Reasons for IPv6

- IPv4 uses 32 bits for addresses
- Real time/streaming traffic (voice, audio)
- Security issues with IPv4

IPv6 – Larger Address Space

- Header format – separates state information from dynamic routing info to simplify router actions
- New Options
- Quality of Service
- Added Security

IPv6 Address Space

- 128 bits (16 bytes)
- 4 hex digits: `xx:xx:xx:xx:xx:xx:xx:xx`
- Can abbreviate by removing leading zeros
 - `:0F:` \Rightarrow `:F:`
 - `xx:0:0:0:AD64:0:0:xx` \Rightarrow `xx::AD64:0:0:xx`
- CIDR Rules also supported (/ nbr of bits)

IPv6 Address Types

- Address types:
 - Unicast: $A \Rightarrow B$
 - Anycast: same first part; subnet broadcast
 - Multicast
- IPv6 Address Format:


| | |
|-------------|---------|
| Type Prefix | Address |
|-------------|---------|

IPv6 Address Format

- Common Type Prefixes
 - 010 = Provider based Unicast
 - 100 = Geographic Unicast
 - 1111 1110 10 = Link Local
 - 1111 1110 11 = Site Local
 - 1111 1111 = Multicast
 - 0000 010 = IPX
 - 0000 001 = NSAP

Provider Based Unicast

| | | | | | |
|-----|----------|----------|------------|--------|------|
| 3 | 5 | 16 | 24 | 32 | 48 |
| 010 | Registry | Provider | Subscriber | Subnet | Node |



| | |
|-------|----------|
| 11000 | INTERNIC |
| 01000 | RIPNIC |
| 10100 | APNIC |

- A = 8 bits = 010 + Registry
- B = variable (16 bits recommended) = Provider
- C = 24 bits = Subscriber
- D = variable (32 bits recommended) = Subnet
- E = variable (48 bits recommended) = Node
 - If Ethernet, Ethernet MAC address recommended

Reserved Addresses

- Starts with: 0000 0000
- Unspecified Address= ::
- Loopback= ::1
- IPv4 Address:
 - 0000 0000 | 88 0's | 32 bit IPv4 Address

Local Address

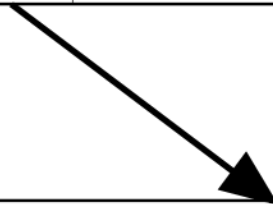
- Starts with: 1111 1110
- Link Local:
 - 10 | 70 0's | 48 bit node address |
- Site Local:
 - 11 | 38 0's | 32 bit subnet | 48 bit node |

Multicast

- Starts with: 1111 1111
- 4 bits = flag
- 4 bits = scope (node local, link local, site local, organization, global)
- 112 bits = Group ID

IPv6 Header

| | | |
|-------------------------|----------------------------|---------------------|
| Base Header 40 bytes | Payload (65,535 bytes max) | |
| | Optional Extension Header | Upper Layer Payload |



| | | | |
|------------------------|----------|-------------|-----------|
| Version | Priority | Flow Label | |
| Payload Length | | Next Header | Hop Limit |
| Source IP address | | | |
| Destination IP address | | | |

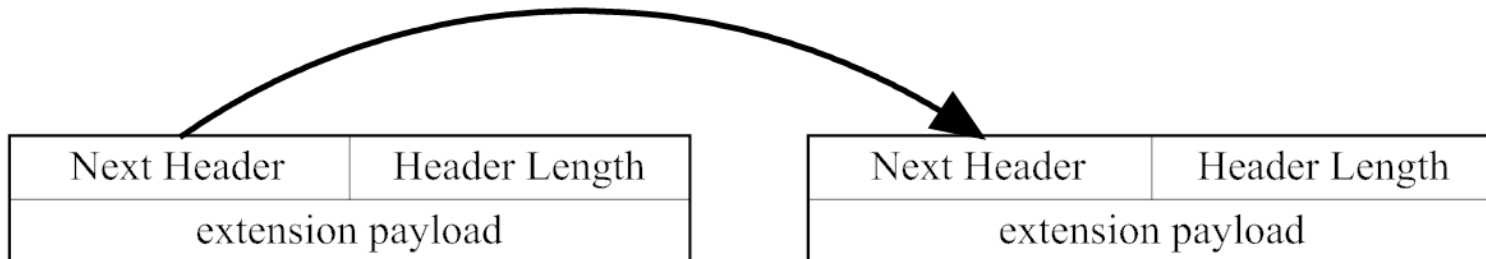
IPv6 Header

Priority Codes

| | | |
|---|-----------------|---------|
| 0 | None | |
| 1 | Background | News |
| 2 | Unattended | email |
| 3 | reserved | |
| 4 | attended bulk | Web |
| 5 | Reserved | |
| 6 | Interactive | Telnet |
| 7 | Control Traffic | routing |

Next header Codes

| Code | Next Header |
|------|---------------|
| 0 | hop by hop |
| 2 | ICMP |
| 6 | TCP |
| 17 | UDP |
| 44 | Fragmentation |
| 50 | Encrypted |
| 51 | Authenticated |
| 53 | None |



Packet Format

- 40 byte base header; N byte Extension Headers
 - 4 bits = 6 (IP version)
 - 4 bits = Priority
 - 24 bits = Flow label
 - 16 bits = Length
 - 8 bits = Next Header
 - 8 bits = Hop Limit
 - 128 bits = Source Address
 - 128 bits = Destination Header

Next Header Codes

- 2 = ICMP
- 6 = TCP
- 17 = UDP
- 43 = Source Routing
- 44 = Fragmentation
- 50 = Encrypted
- 51 = Authentication

Priority (Part 0-7)

- Congestion Controlled
 - 0 = None
 - 1 = background (news)
 - 2 = unattended (email)
 - 3 = reserved
 - 4 = Attended bulk (HTTP/FTP)
 - 5 = Reserved
 - 6 = Interactive
 - 7 = Control traffic (routing)

Priority (8-15)

- Noncongestion Controlled
 - 8 = Most redundancy
 - :
 - 15 = Least redundancy

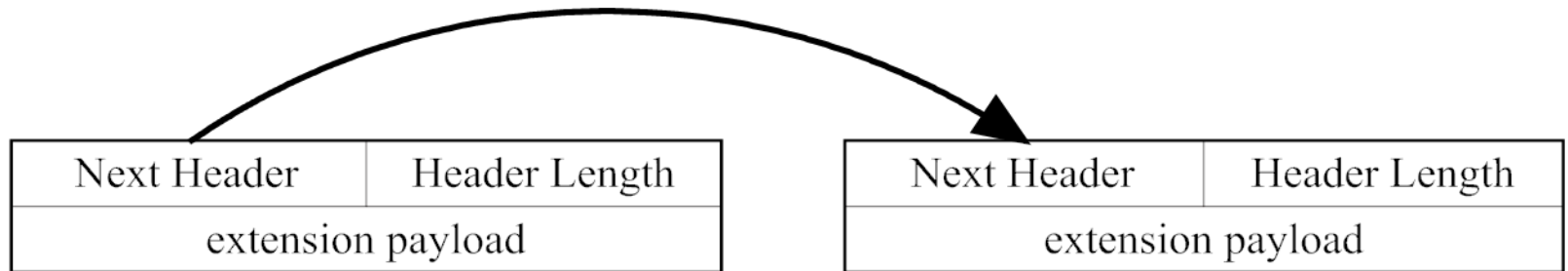
Flow Label

- Flow Label + Source Address is unique
- Router can cache “Flow Label + Source Address” to speed up routing
- TCP routing can take up to 70% of the processing with IPv4

Items not in IPv6 Headers

- ID/Offset = only needed if handling fragmentation/reassembly (not needed by routers)
- No checksum = minimal value

Extension Headers



- Can be chained
- If Next Header = 59, last header

Extension Header Types

- 1 = Hop by Hop Option
- 2 = Jumbo Payload (if payload > 65535 bytes)
 - up to $2^{32}-1$
- 3 = Source Routing
- 4 = Fragmentation (use Path MTU Discovery)
- 5 = Authentication (Authenticates sender)
- 6 = Encrypted

ICMPv6

- ICMPv6: Internet Control Messaging Protocol
- Many of the TCP/IP protocols (ARP, etc) are covered by ICMPv6 so are no longer needed

Error Reporting Packet Format

| | | |
|---|---|---|
| A | B | C |
| D | | |
| E | | |

- A = Type
- B = Code
- C = Checksum
- D = Other Information
- E = Data

Error Reporting Types

- 1 = Destination Unreachable
- 2 = Packet too big
- 3 = Time exceeded
- 4 = Parameter problem
- 137 = Redirection

Type 1 Codes

- 0 = No path
- 1 = Communications is prohibited
- 2 = Source routing is impossible
- 3 = Destination address is unreachable
- 4 = Port

Type 2 & 3 Codes

- Type 2 Codes
 - 0 = MTU exceeded
- Type 3 Codes
 - 0 = Hop Count
 - 1 = Fragment timeout

Type 4 & 5 Codes

- Type 4 Codes
 - 0 = Header
 - 1 = Extension Header
- Type 5 Codes
 - 0 = Router finds better path

General countermeasures

- Since IP is so ingrained in the Internet it is hard to provide security. There are a few general countermeasures.
 - IP Filtering
 - Network Address Translation (NAT)
 - Virtual Private Network (VPN)
 - Encrypted IPV4 & IPV6 (IPSec)

IP Filtering

- Routers can be configured to filter out packets based on:
 - IP Address (black listing)
 - Hard to keep list current
 - Hard to get off the list (DOS)
 - Port numbers
 - Rogue protocols use multiple ports
 - Protocol types (TCP, UDP, ICMP)
 - Course grain filtering

Network Address Translation

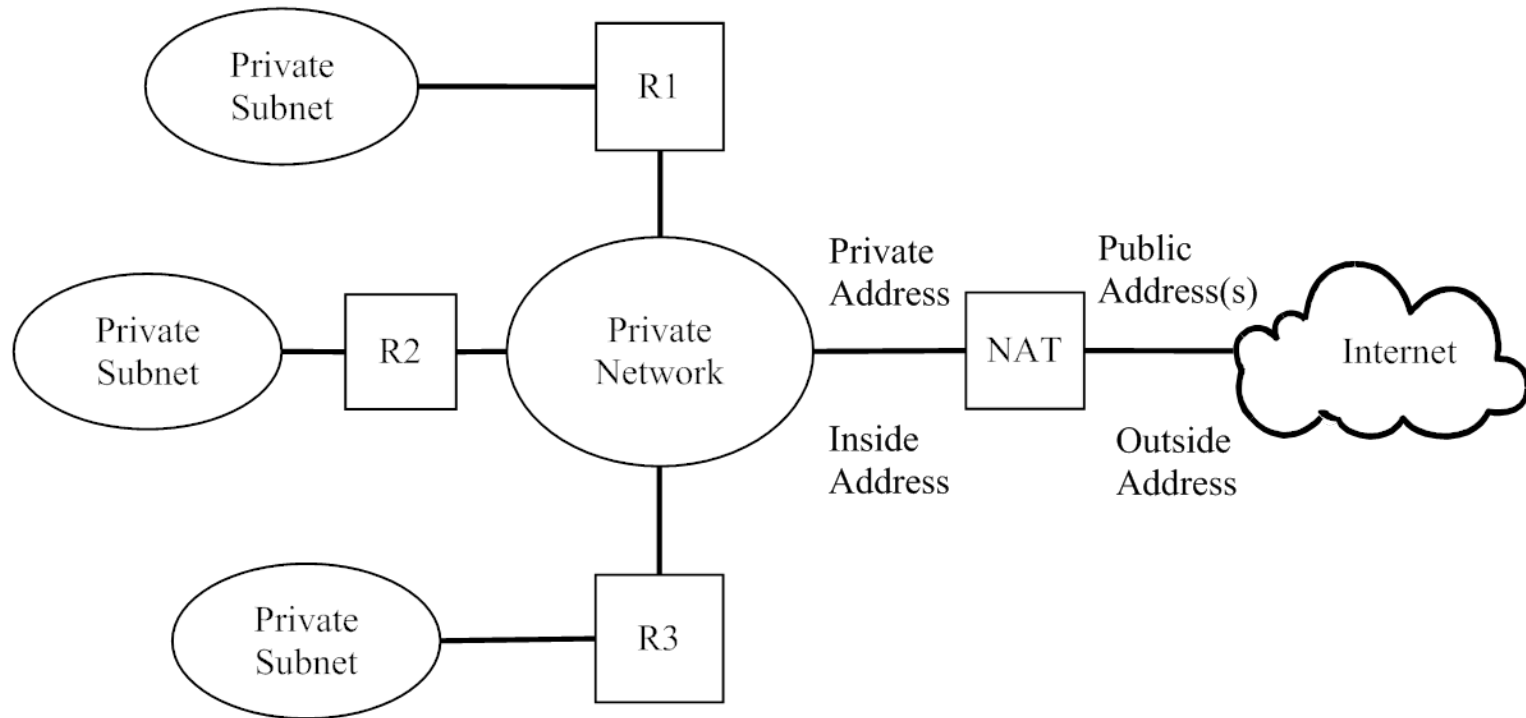


Figure 6.30 Private Network

Network Address Translation

- Used to extend the address space
 - Internal address ranges
 - 10/8 10.0.0.0
 - 172.16/12 172.16.0.0 (16 class B networks)
 - 192.168/16 192.168.0.0 (class B network)
- Static NAT
- Dynamic NAT

NAT

- Not really designed as a security device
- Does not provide security and is often coupled with a firewall

Static NAT

- One to one mapping of external addresses to internal addresses
- Used when a small number of machines need Internet access.
- NAT looks like a router to the inside machines and the destination to outside machines

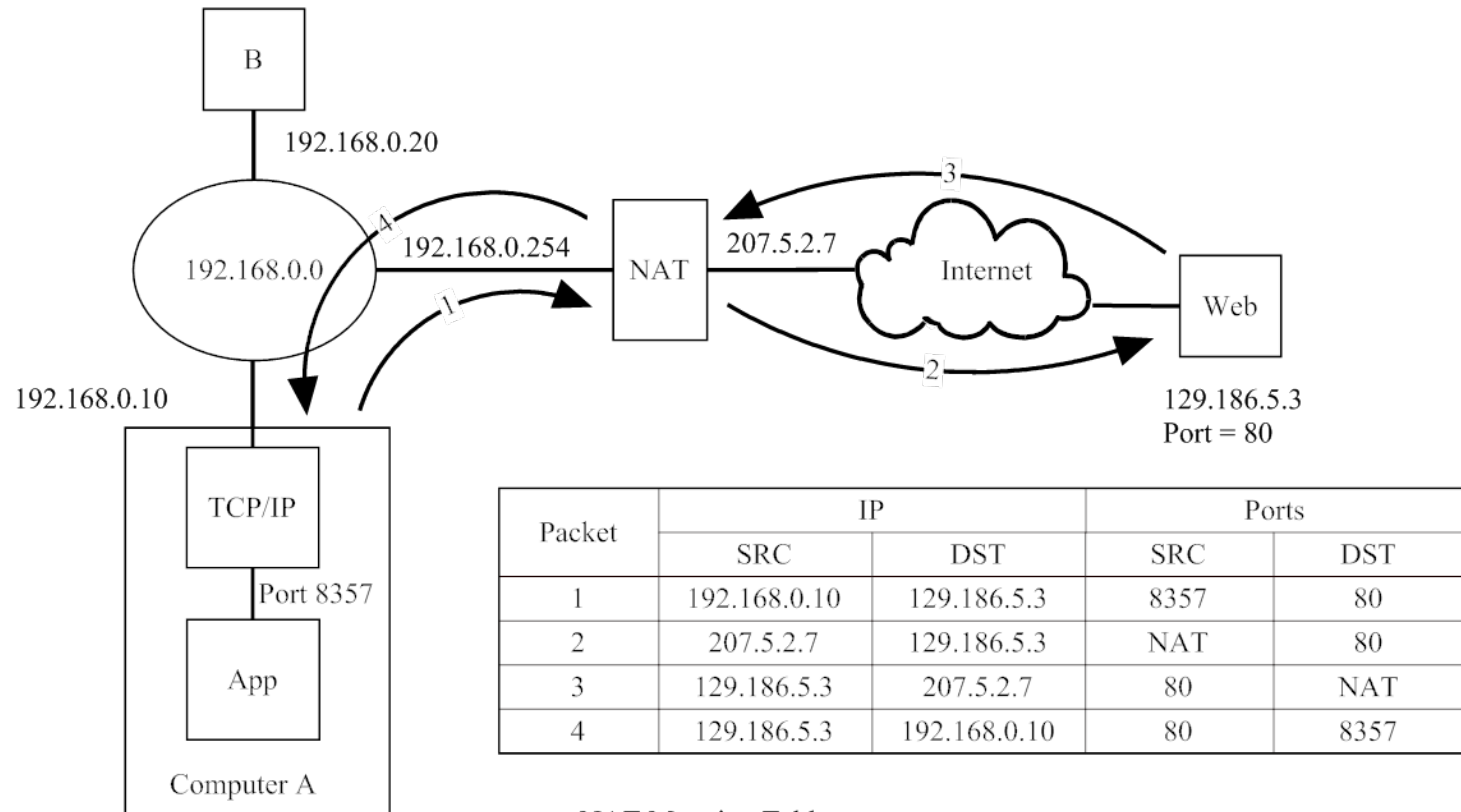
Static NAT

| Public | Port | Private | Port |
|---------------|------|---------------|------|
| 129.186.5.100 | 80 | 192.168.20.30 | 80 |
| 129.186.5.150 | 25 | 192.168.20.50 | 80 |
| | | | |

Dynamic NAT

- More machines on the inside than IP addresses on the outside.
- Used for outgoing access
- Can use tunnels for servers or combine with static NAT
- Inside can have same address range as a valid outside network (overlapping)

Dynamic NAT (Port mapping)



NAT Mapping Table

| Public IP | Ports | | Private IP | Ports | |
|-------------|-------|-----|--------------|-------|-----|
| | SRC | DST | | SRC | DST |
| 129.186.5.3 | NAT | 80 | 192.168.0.10 | 8357 | 80 |

Public servers

- Servers need a public address
 - Two networks
 - Tunneling

Public & Private Networks

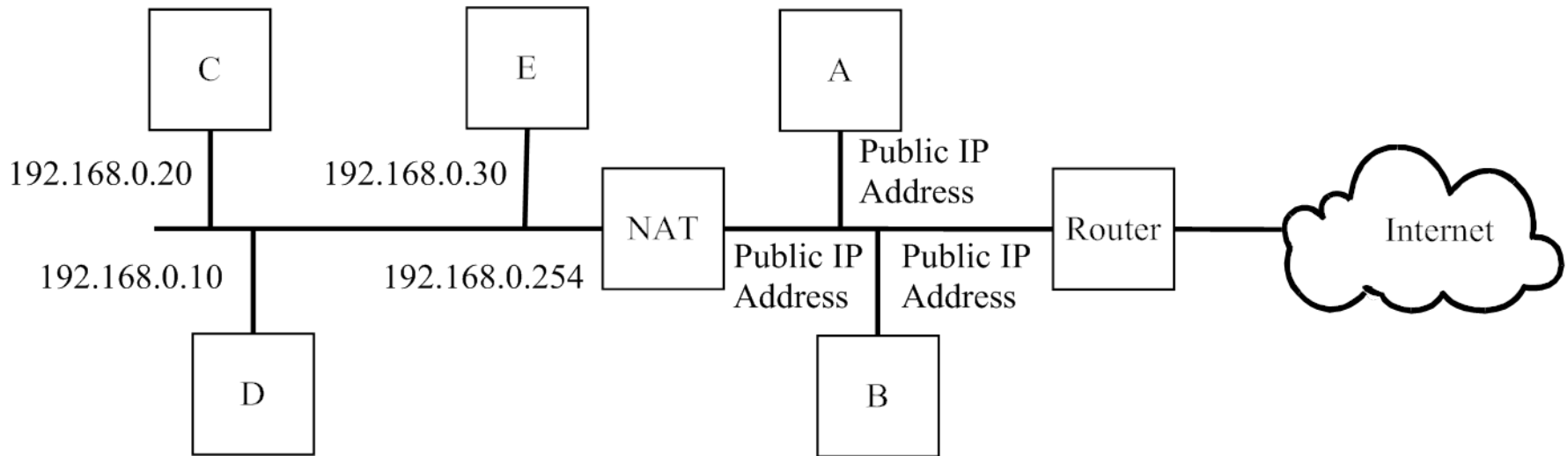
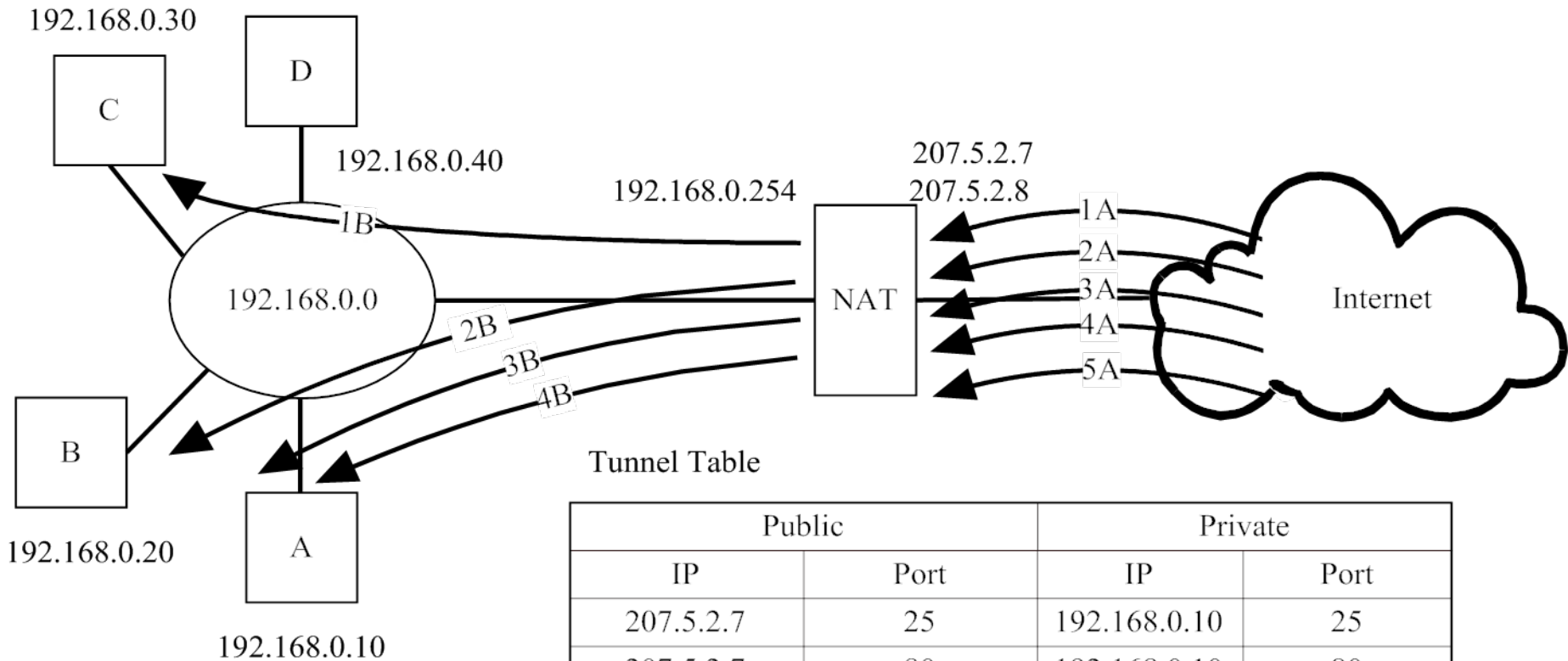


Figure 6.32 Public Servers and a Private Network

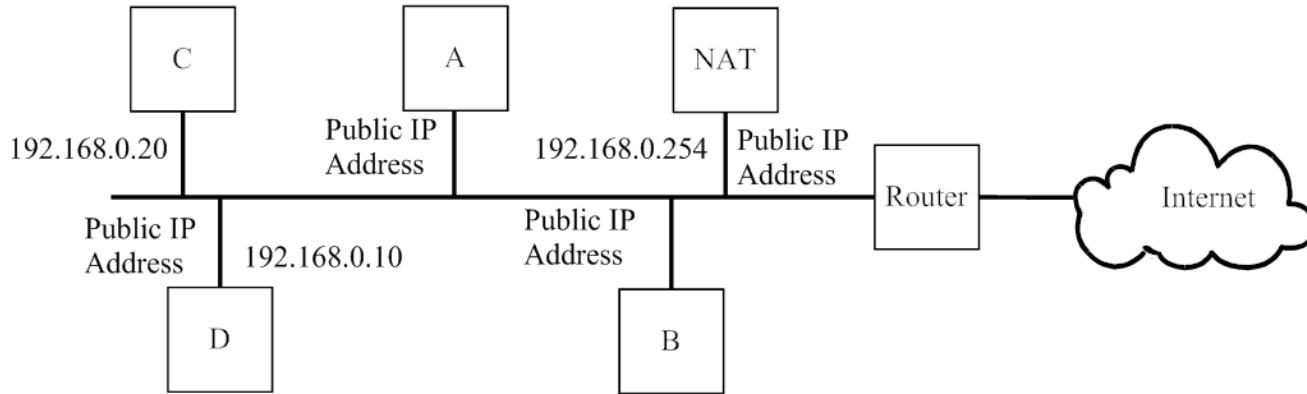
Tunneling through a NAT



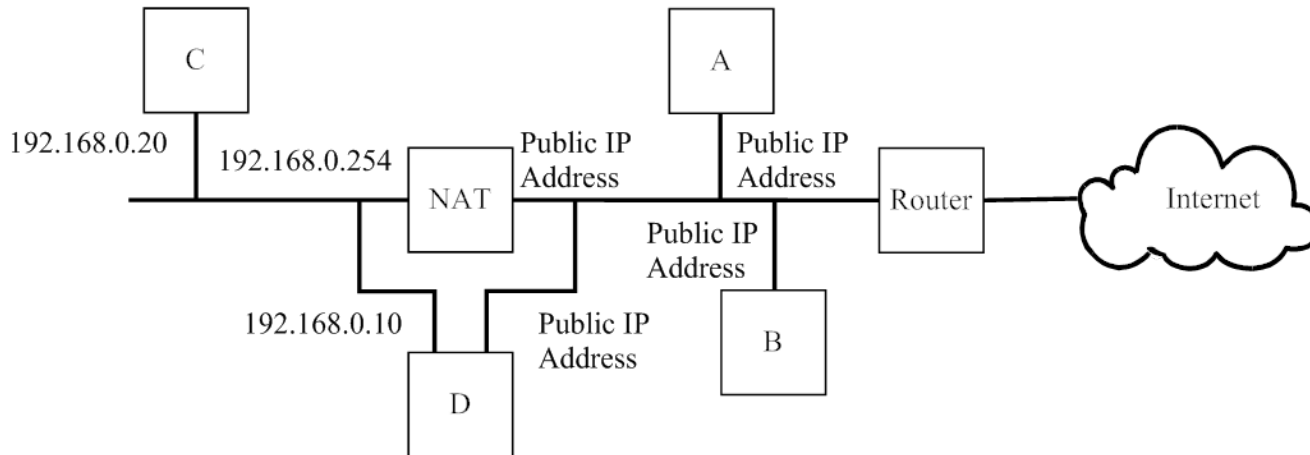
Tunneling through a NAT

| Packet | IP | | Ports | |
|--------|----------|--------------|-------|-----|
| | SRC | DST | SRC | DST |
| 1A | Internet | 207.5.2.8 | 8357 | 100 |
| 1B | Internet | 192.168.0.30 | 8357 | 80 |
| 2A | Internet | 207.5.2.8 | 7384 | 80 |
| 2B | Internet | 192.168.0.20 | 7384 | 80 |
| 3A | Internet | 207.5.2.7 | 2345 | 80 |
| 3B | Internet | 192.168.0.10 | 2345 | 80 |
| 4A | Internet | 207.5.2.7 | 2554 | 25 |
| 4B | Internet | 192.168.0.10 | 2554 | 25 |
| 5A | Internet | 207.5.2.7 | 6623 | 22 |

Pass-by NAT



Physical Configuration



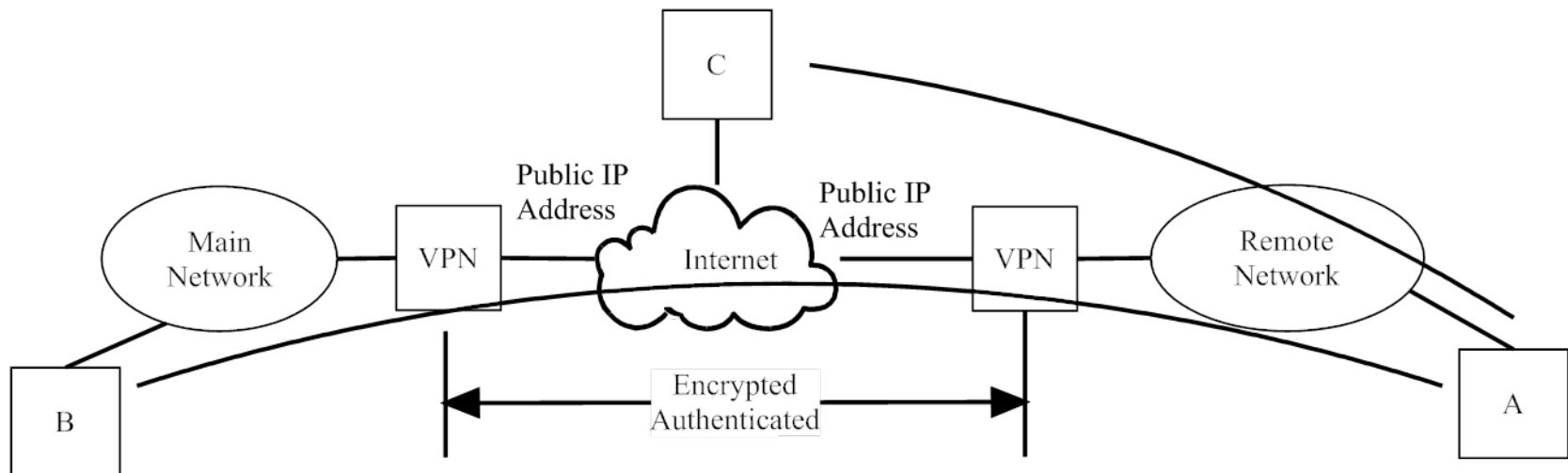
Logical Configuration

Virtual Private Network

- Used to create encrypted tunnels between devices
- Uses many different protocols
 - SSH
 - IPSEC
 - Proprietary

Network to network VPN

VPN only when talking to target network
Other traffic goes directly to destination

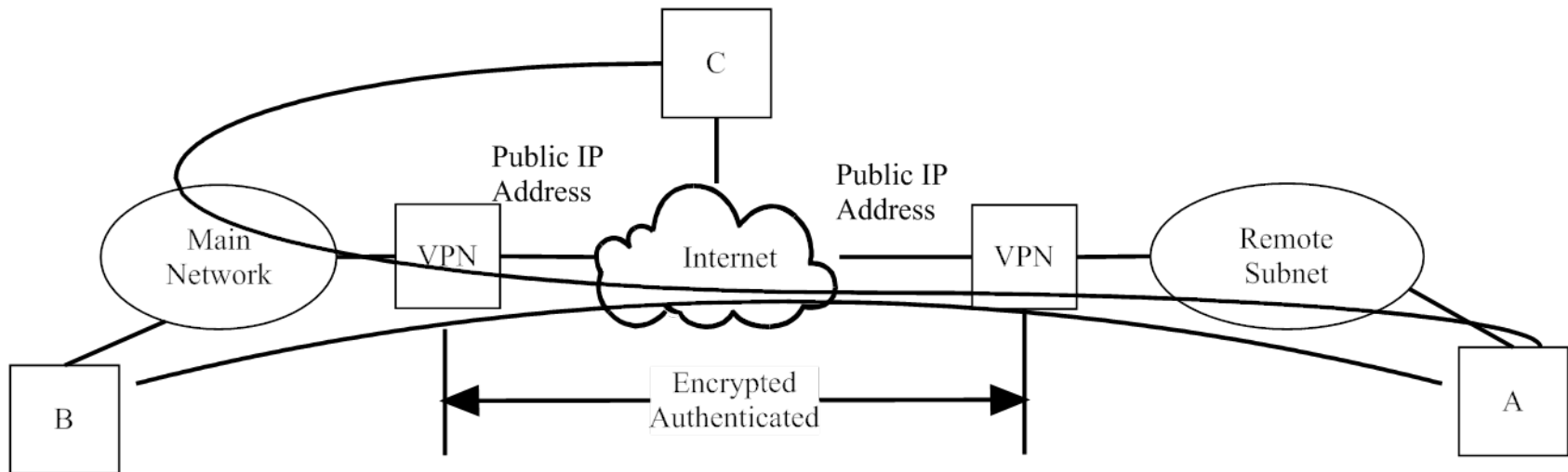


Remote Network

Network to network VPN

Always uses VPN

All traffic is routed through target network



Remote Subnet

Client to client VPN

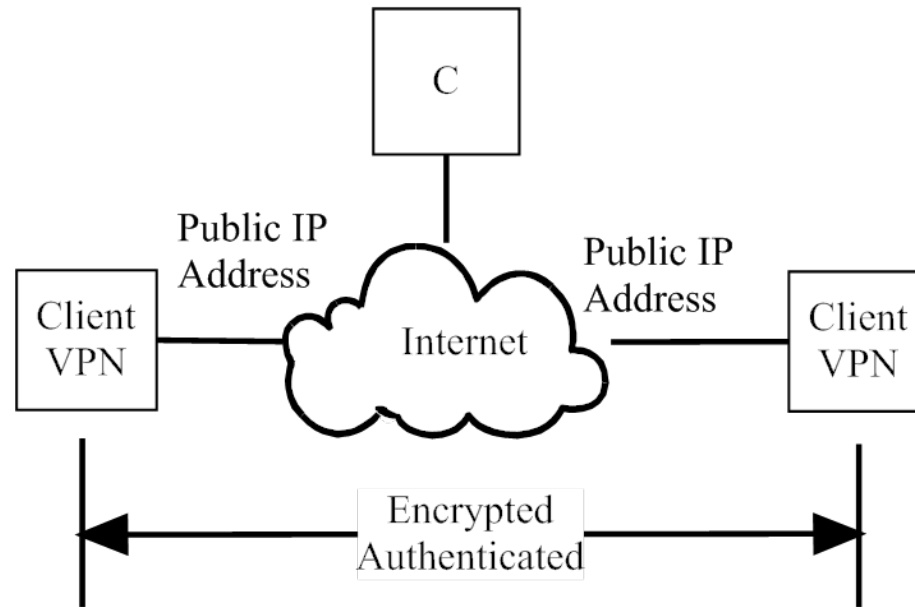
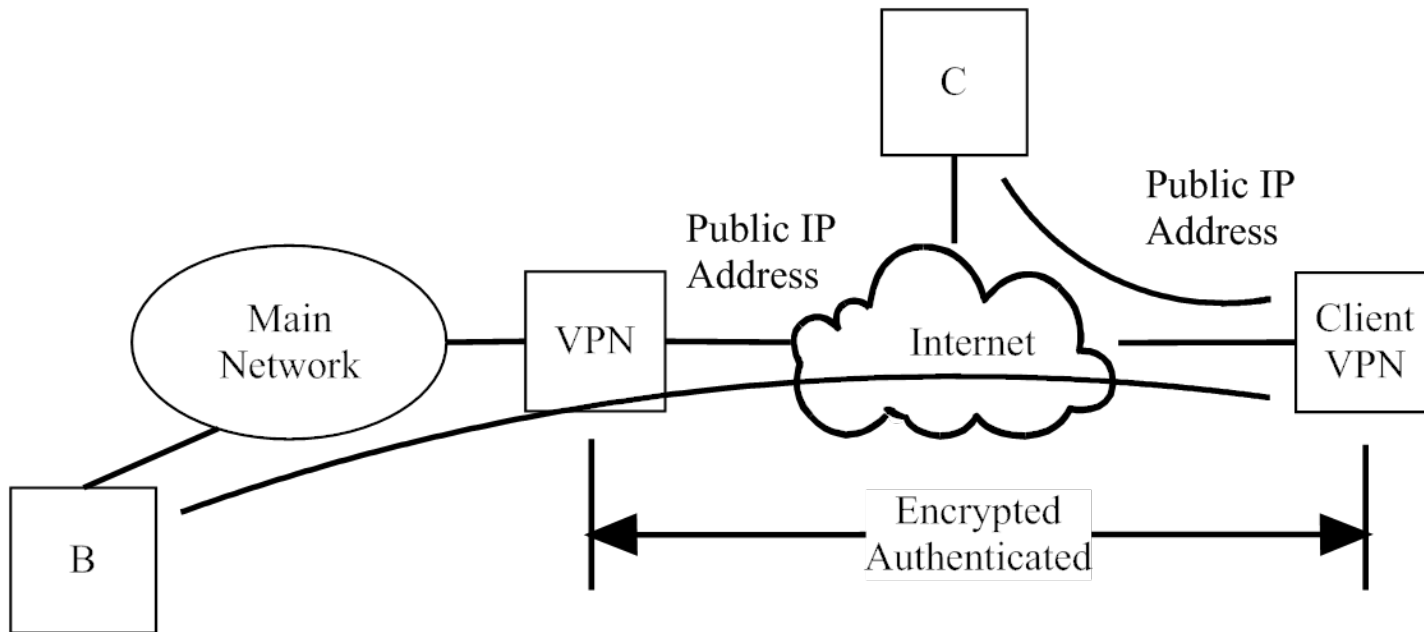


Figure 6.36 Client to Client VPN

Client to Network

Always uses VPN

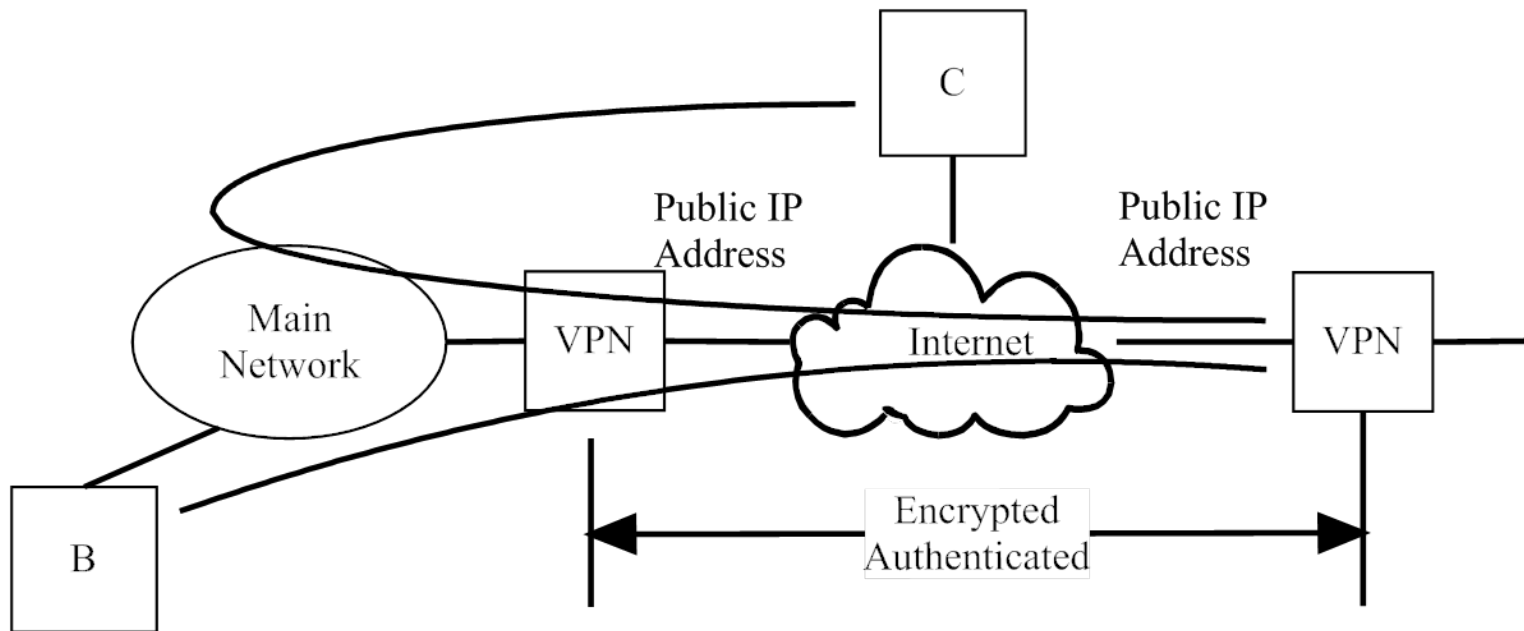
All traffic is routed through target network



Client to network

Always uses VPN

All traffic is routed through target network



Remote Subnet

IPSEC

- Two Purposes
 - Authentication: sender & receiver (prevents IP spoofing)
 - Encryption: data privacy
- IPSEC is not end-to-end

IPSEC

- AH = Authentication Header Not used much)
- ESP = Encapsulating Security Payload
- Not Specified in IPSEC Policy
 - Encryption Algorithms
 - Key Management
 - Domain of Interpretation

IPSEC Services

| AH | ESP | BOTH | IPSEC Service |
|----|-----|------|--------------------------------------|
| X | X | X | Access Control |
| X | | | Connectionless Integrity |
| X | | | Data Origin Authentication |
| X | X | X | Reject of Replay |
| | X | | Confidentiality |
| | X | | Limited Traffic Flow Confidentiality |

AH

| Size | Field |
|----------|---------------------|
| 8 bits | Next |
| 8 bits | Length of Header |
| 16 bits | Reserved |
| 32 bits | Security Parameters |
| 32 bits | Sequence Number |
| Variable | Authentication Data |

Authentication Data: MD5 (1-way Hash)

AH Use: End-to-End or End-to-Intermediate Node

IPv4 Use of AH in IPSEC

Original IPv4 Packet

| | | |
|--------|---------|------|
| IP Hdr | TCP Hdr | Data |
|--------|---------|------|

Transport Mode IPv6 Packet

| | | | |
|--------|----|---------|------|
| IP Hdr | AH | TCP Hdr | Data |
|--------|----|---------|------|

Tunnel Mode IPv6 Packet

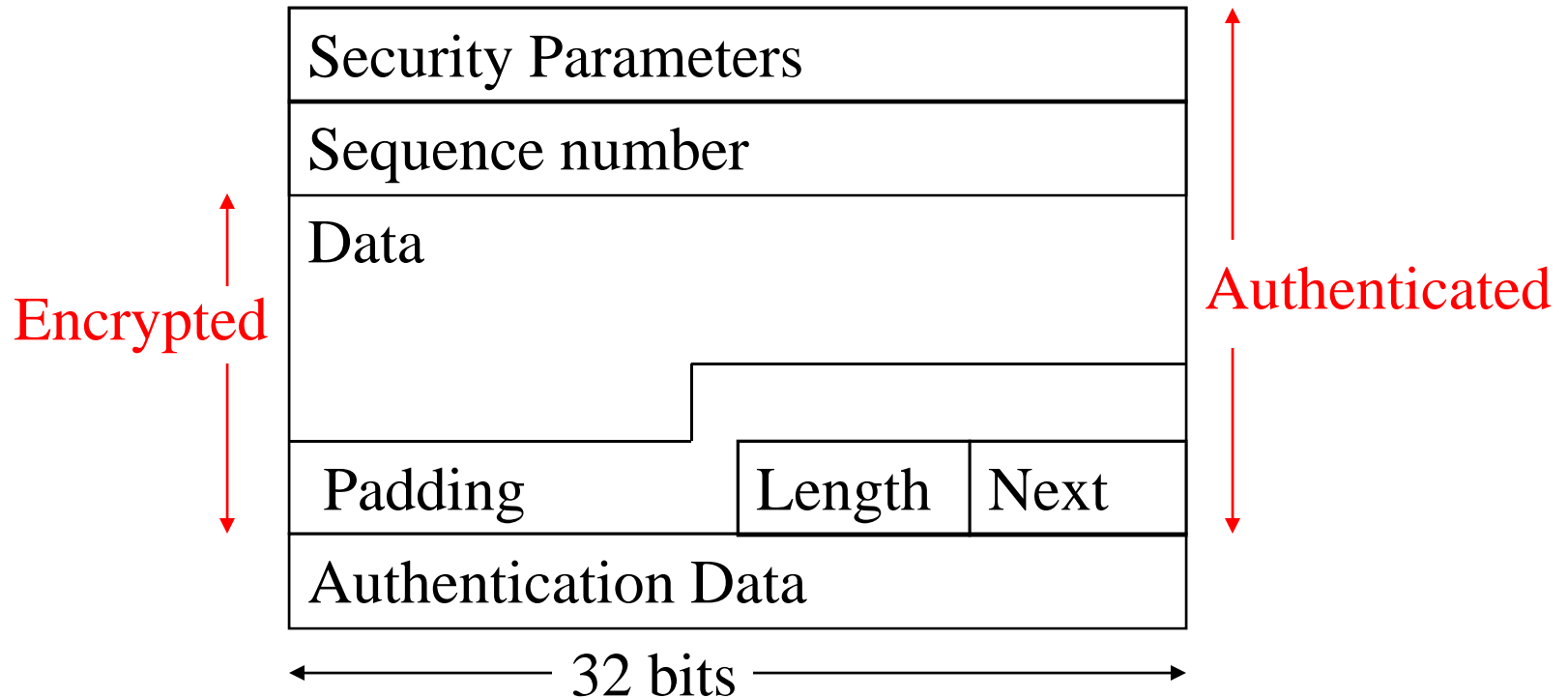
| | | | | |
|------------|----|--------|---------|------|
| New IP Hdr | AH | IP Hdr | TCP Hdr | Data |
|------------|----|--------|---------|------|

<-----Original Packet----->

ESP

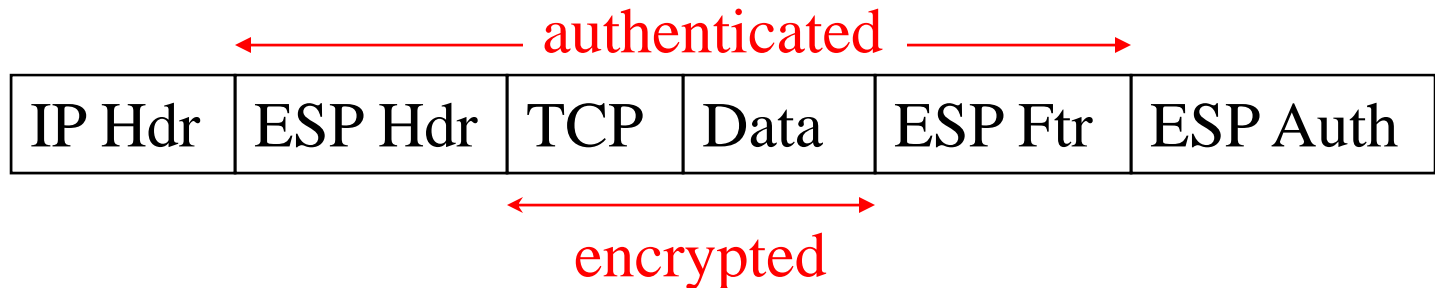
- Encapsulating Security Payload
 - Security Parameters: help identify the encryption algorithm (eg: DES, blowfish)
 - Sequence number: an ever increasing number used for replay
 - Authentication data: a hash of everything, proves non-alteration
 - Data, Padding, Length, and Next fields are all encrypted

Encapsulating Security Payload



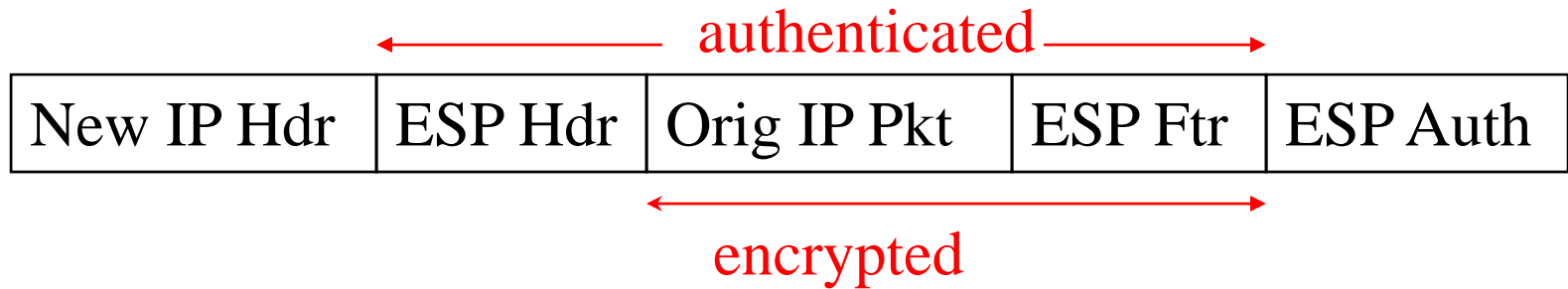
Encapsulating Security Payload

- There are two ways encryption can be handled:
 - Transport Level (end-to-end)
 - Tunnel mode (also referred to as VPN)
- Packet format for IPv4:



Encapsulating Security Payload

- Packet format for IPv6:



- Tunneling mode:

