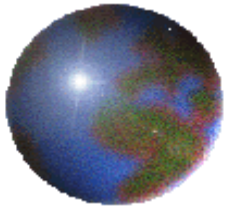


# CEN445
# Network Protocols & Algorithms

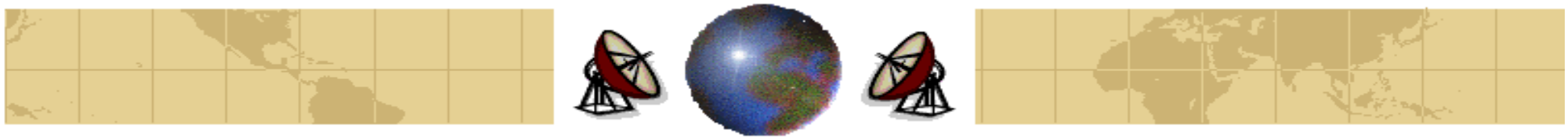
Chapter 5

## Network Layer

***Prepared by***

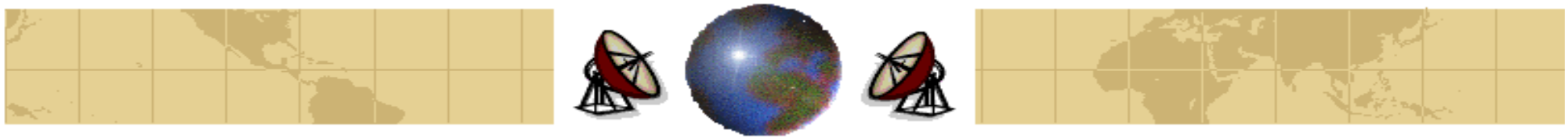
***Dr. Mohammed Amer Arafah***

***Summer 2008***



# *The Network Layer in the Internet*

- The IP Protocol
- IP Addresses
- Internet Control Protocols
- OSPF – The Interior Gateway Routing Protocol
- BGP – The Exterior Gateway Routing Protocol
- Internet Multicasting
- Mobile IP
- IPv6

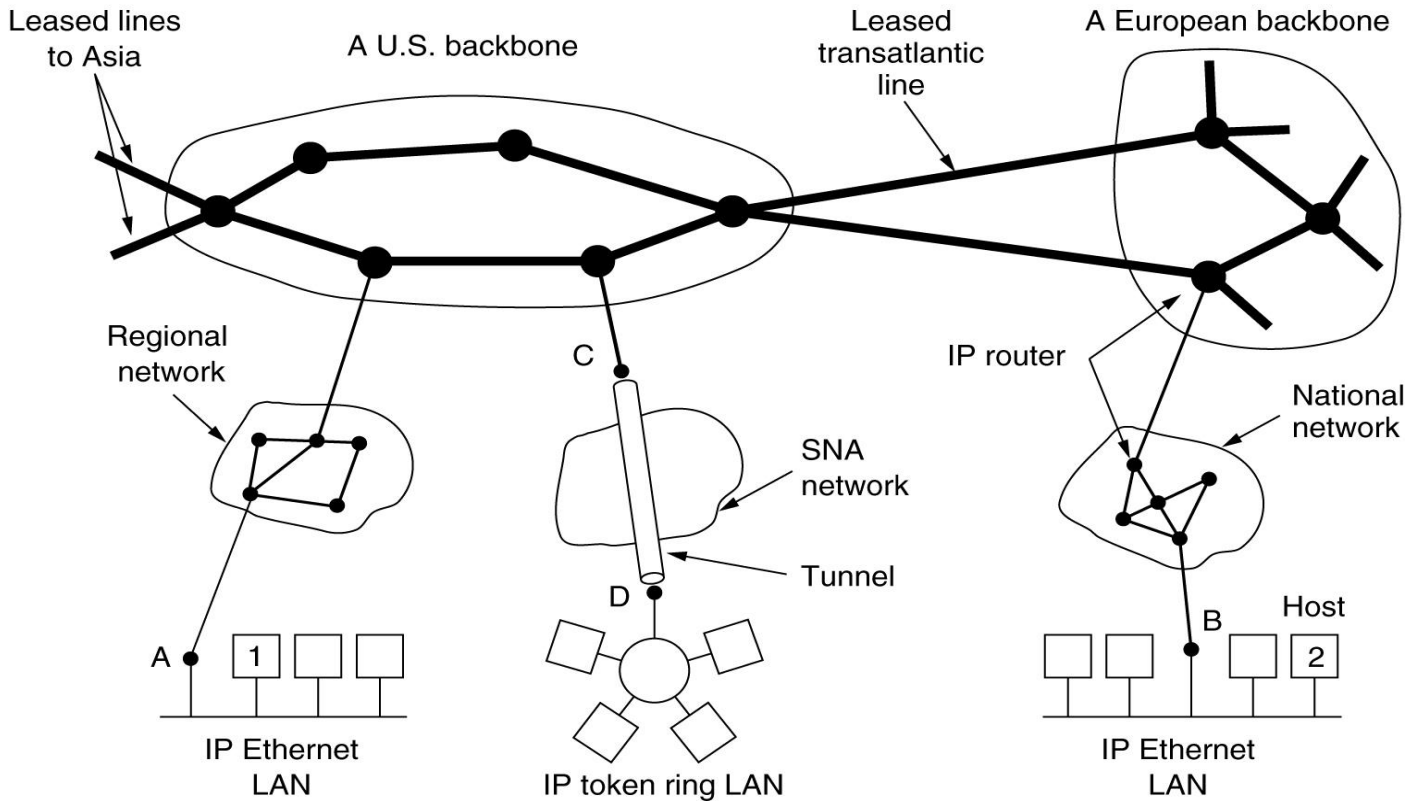


# *The Network Layer in the Internet*

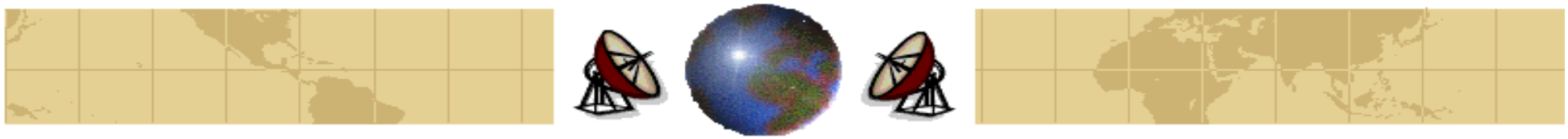
- The Internet can be viewed as a collection of **Autonomous Systems (ASes)** that are connected together.

- There is no real structure, but several **major backbones** exist. These are constructed from high-bandwidth lines and fast routers. Attached to the backbones are **regional networks**, and attached to these regional networks are the **LANs**.

# Collection of Subnetworks

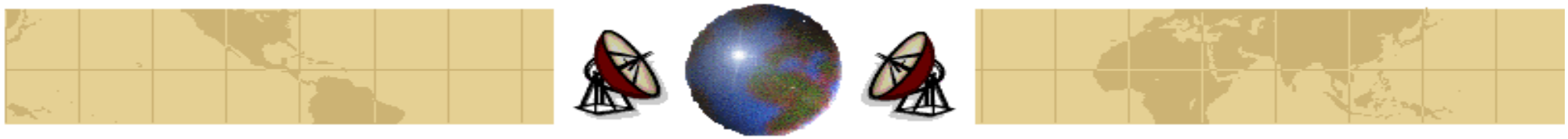



The Internet is an interconnected collection of many networks



# *Design Principles for Internet*

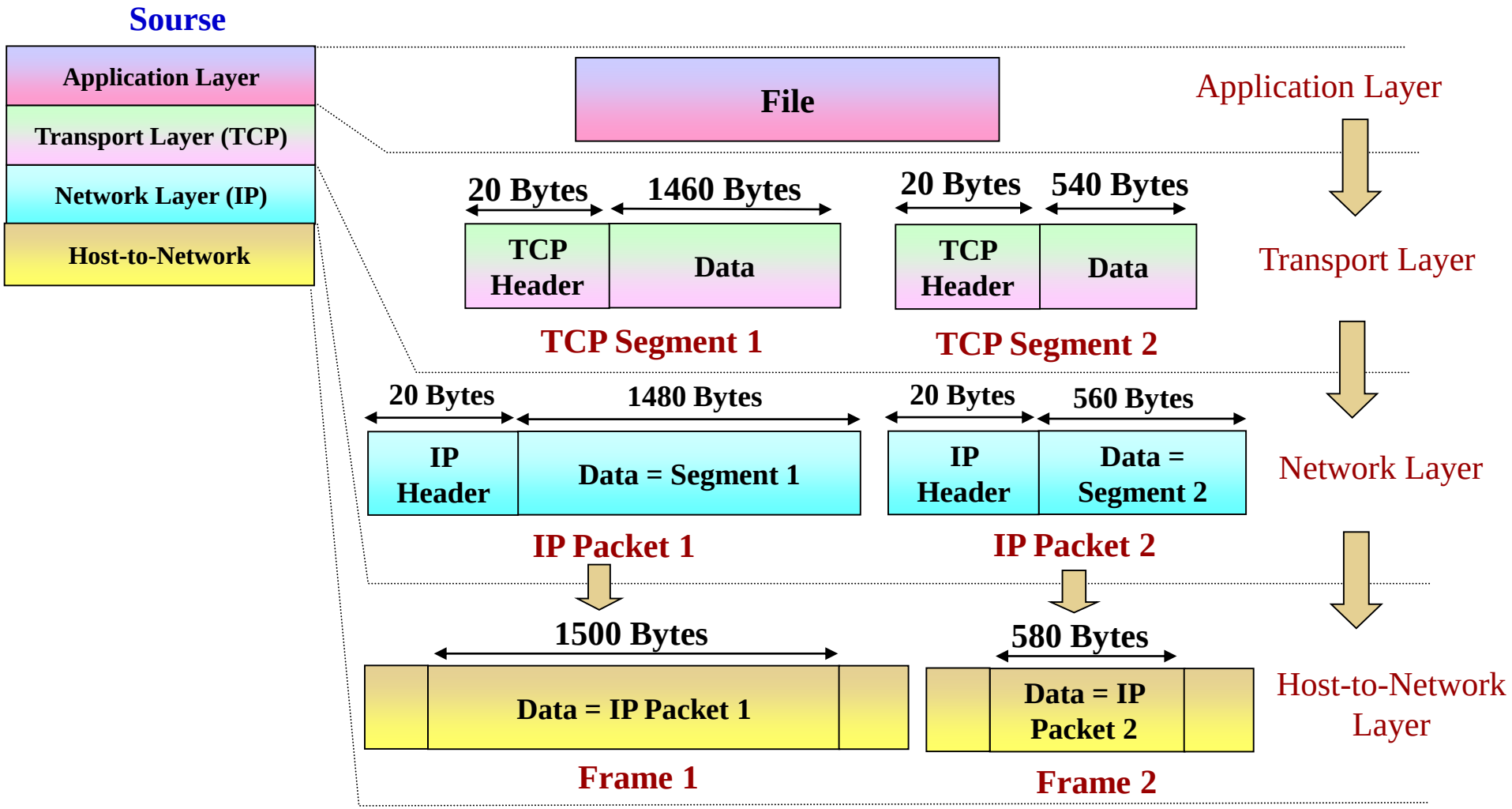
- Make sure it works.
- Keep it simple.
- Make clear choices.
- Exploit modularity.
- Expect heterogeneity.
- Avoid static options and parameters.
- Look for a good design; it need not be perfect.
- Be strict when sending and tolerant when receiving.
- Think about scalability.
- Consider performance and cost.



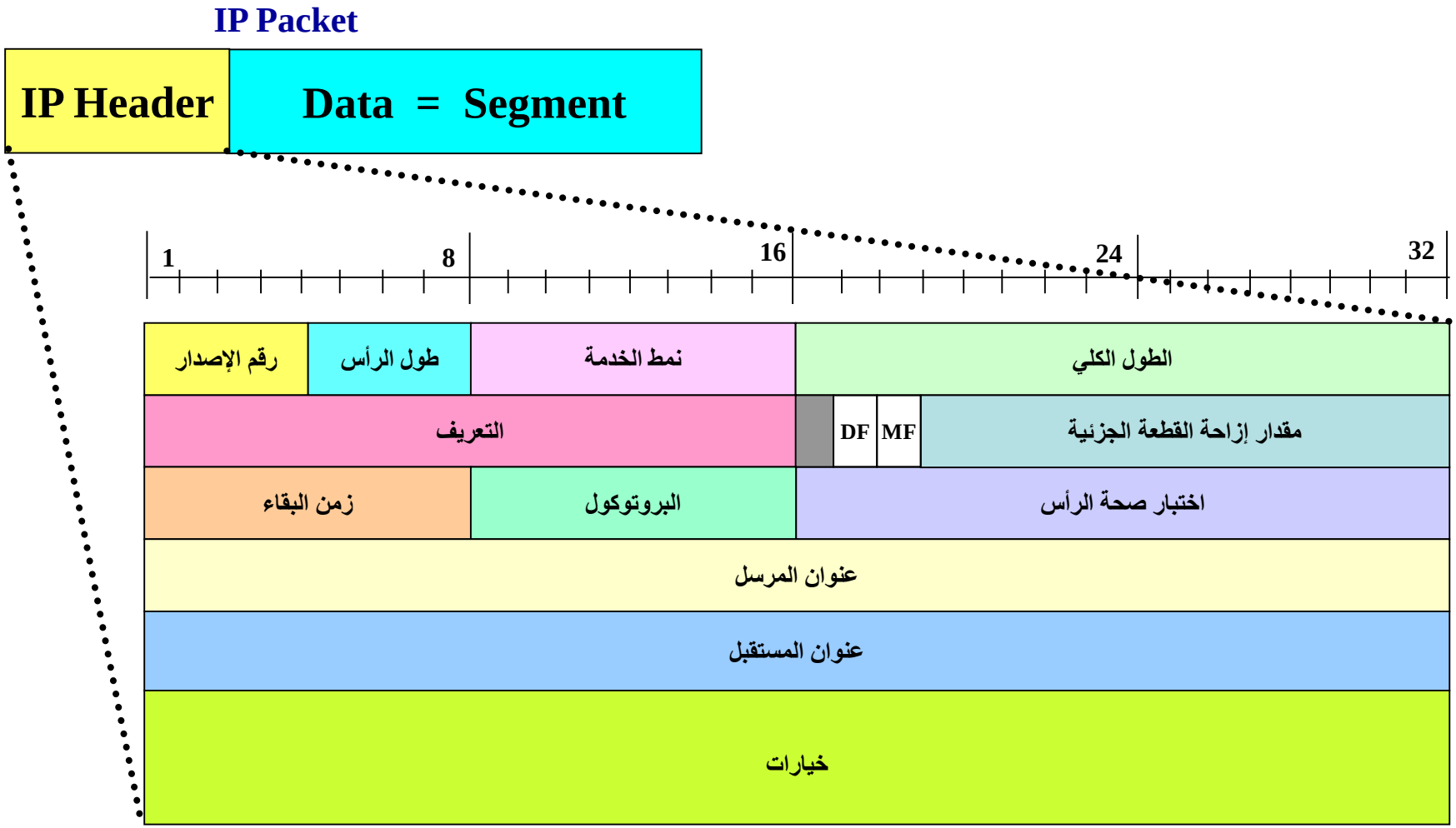
# The Network Layer in the Internet

- The glue that holds the Internet together is the network layer protocol, **IP** (**Internet Protocol**).
- It was designed from beginning with internetworking in mind.
- Its **job** to provide a best efforts way to transport datagrams from source to destination, without regard to whether or not these machines are on the same network, or whether or not there are other networks in between them.
- **Communication in the Internet works as follows**. The transport layer takes data streams and breaks them up into datagrams. Each datagram is transmitted through the Internet, possibly being fragmented into smaller units as it goes. When all pieces gets to destination machine, they are reassembled by the network layer into the original datagram. This datagram is then handled to the transport layer.

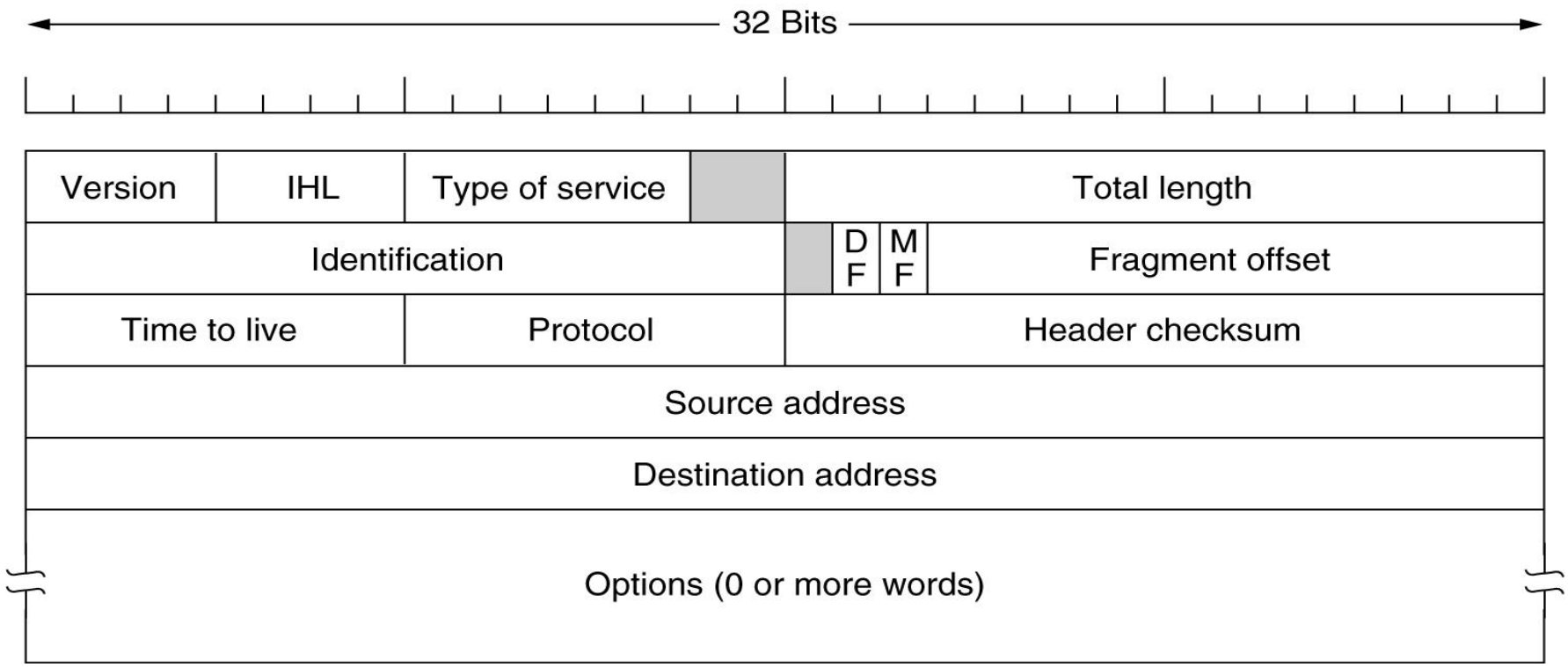
# The Network Layer in the Internet



# Format of IP Datagram



# Format of IP Datagram




The IPv4 (Internet Protocol) header

# Format of IP Datagram

## Version Field (4-bit):

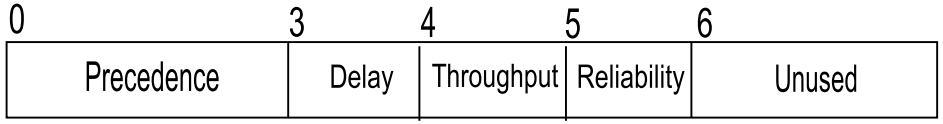
- It keeps track of which version of protocol the datagram belongs to.

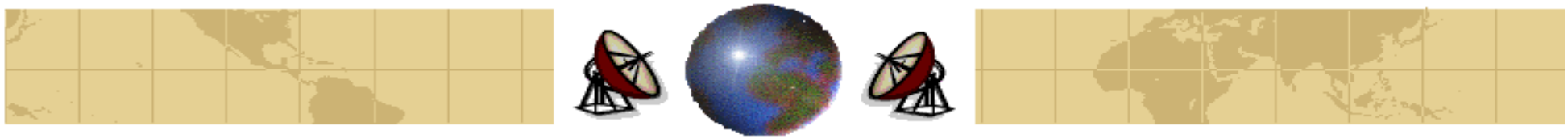
## IHL Field (4-bit):

- It tells how long the header is, in 32-bit words.
- The minimum value of this 4-bit field is 5, and the maximum value is 15, which limits the header to 60 bytes. Thus the option field is limited to 40 bytes.

## Type of Service (8-bit):

- It allows the host to tell the subnet what kind of service it wants.
- Various combinations of reliability and speed are possible.
- For digitized voice, fast delivery beats accurate delivery. For file transfer, error-free transmission is more important than fast transmission.
- The field itself is divided as:





# Format of IP Datagram

## Total Length Field (16-bit):

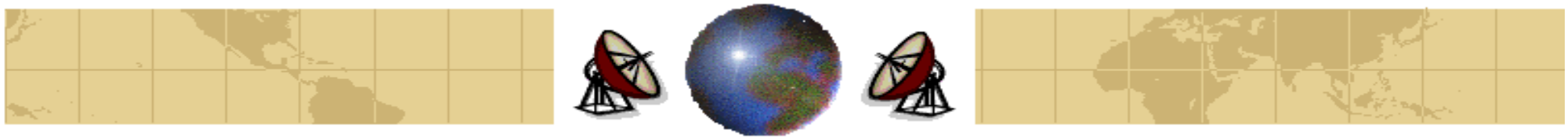
- The maximum length of both header and data is 65,535 bytes.
- In practice, the datagram length is around 1500 bytes, but for future gigabit networks, larger datagrams are needed.

## Identification Field (16-bit):

- It allows the destination host to determine which datagram a newly arrived fragment belongs to.
- All the fragments of a datagram contain the same *Identification* value.

## DF Field (1-bit):

- It stands for *Don't Fragment*.
- The destination is incapable of putting the pieces back together.
- To avoid fragmentation, the datagram must avoid all small-packet networks.



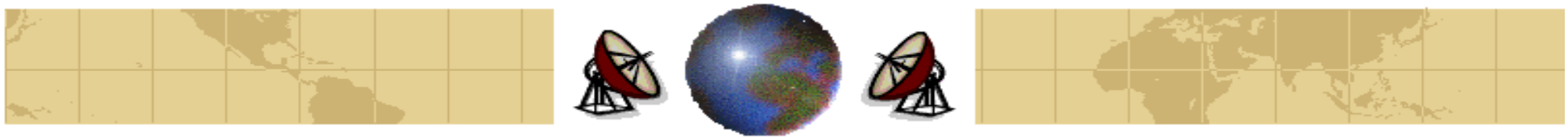
# Format of IP Datagram

**MF Field (1-bit):**

- It stands for *More Fragments*.
- All fragments except the last one have this bit set.
- It is needed to know when all fragments of a datagram have arrived.

**Fragment Offset Field (13-bit):**

- It tells where in the current datagram this fragment belongs.
- All fragments except the last one in a datagram must be a multiple of 8 bytes, the elementary fragment unit.
- Since 13 bits are provided, there is a maximum of 8192 fragments per datagram.



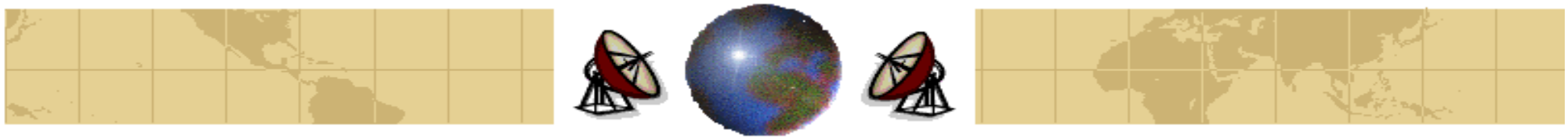
# Format of IP Datagram

## Time to Live Field (8-bit):

- It is a counter to limit packet lifetimes.
- It is supposed to count time in second, allowing maximum lifetime of 255 sec.
- In practice, it just counts hops. When it hit zero, the packet is discarded and a warning packet is sent back to the source host.
- This feature prevents datagrams from wandering around forever if the routing tables ever become corrupted.

## Protocol Field:

- When the network layer has assembled a complete datagram, it needs to know what to do with it.
- The *protocol field* tells the network layer which transport process to give it to, **TCP** is one possibility, but so are **UDP** and some others.
- The numbering of protocols is global across the entire Internet.



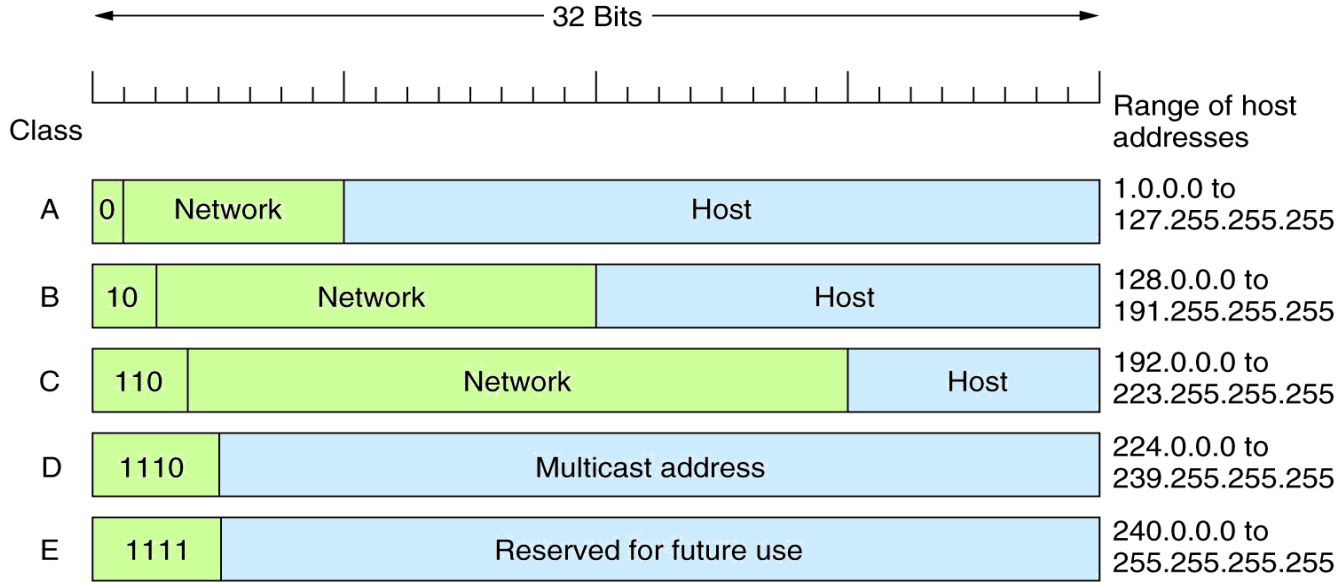
# Format of IP Datagram

| Option | Description |
|---|---|
| Security | Specifies how secret the datagram is |
| Strict source routing | Gives the complete path to be followed |
| Loose source routing | Gives a list of routers not to be missed |
| Record route | Makes each router append its IP address |
| Timestamp | Makes each router append its address and timestamp |

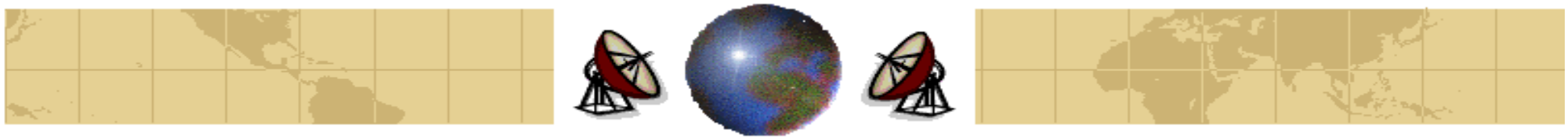
Some of the IP options

# IP Addresses

- Every host and router on the Internet has an **IP address**, which encodes its **network number** and **host number**. The combination is unique: no two machines have the same IP address.
- All IP addresses are 32 bits long and are used in the *Source Address* and *Destination Address* fields of IP packets.
- The formats used for IP addresses are:




IP address formats



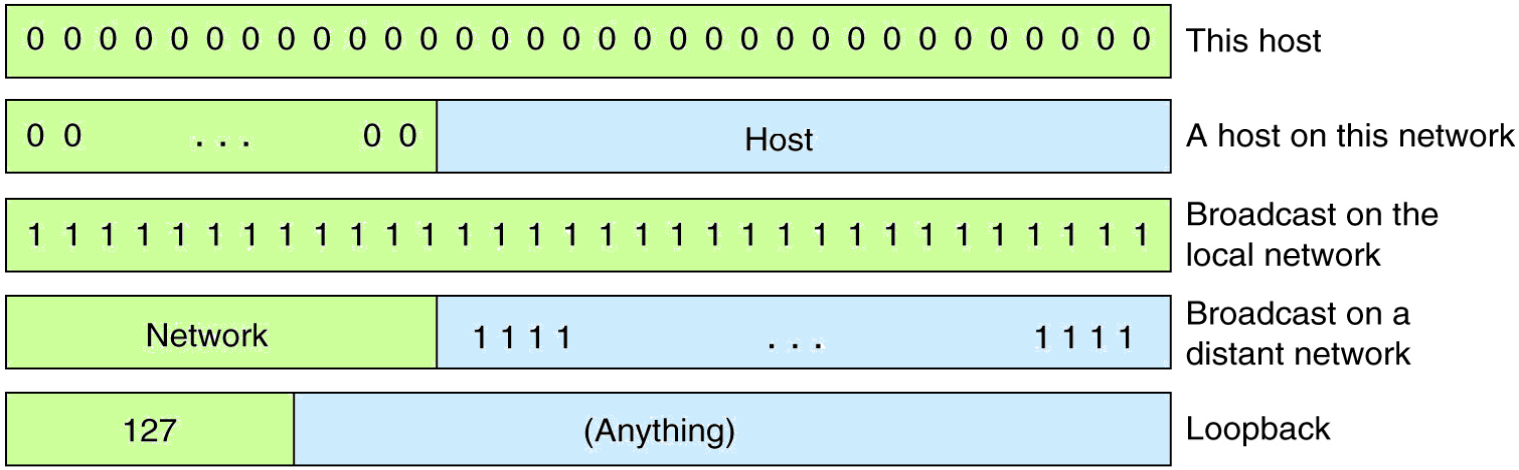
# IP Addresses

- The class A, B, C, D formats allow for up to 126 networks with 16 million host each, 16,382 networks with up to 64K hosts, 2 million networks, (e.g., LANs), with up to 254 hosts each, and multicast.
- Tens of thousands of networks are now connected to the Internet, and the number doubles every year.
- Network numbers are assigned by the **ICANN (Internet Corporation for Assigned Names and Numbers)** to avoid conflicts.
- 32-bit addresses are usually written in **dotted decimal notation**. Each of the 4 bytes is written in decimal, from 0 to 255.
- For example, the hexadecimal address C0290614 is written as 192.41.6.20.
- The *lowest IP address* is 0.0.0.0 and the *highest* is 255.255.255.255.

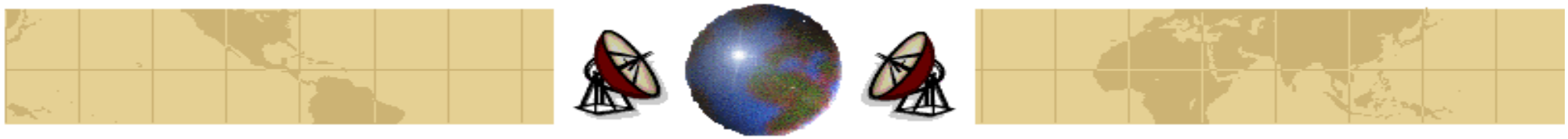
# IP Addresses

- The value 0 and -1 have special meanings. The value 0 (**0.0.0.0**) means *this network or this host*. The value -1 (**255.255.255.255**) is used as a *broadcast* address on the local network.

- All addresses of the form **127.xx.yy.zz** are reserved for *loopback testing*. Packets sent to that address are not put out onto the wire; they are processed locally and treated as incoming packets. This feature is used for debugging.




Special IP addresses



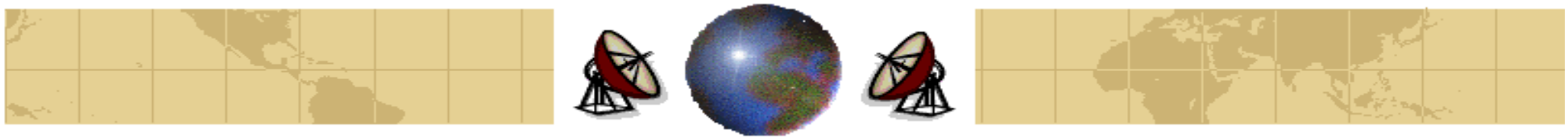
# *Subnet Mask*

- **Subnet Mask** is used to extract the network number from the IP address.

**Example:**

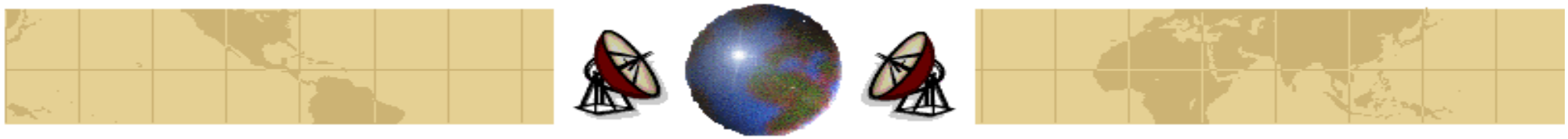
- If the IP address is 200.200.200.5, we can conclude that it is a class C and the subnet mask is 255.255.255.0. Therefore, by performing AND operation bitwise, we can obtain the network number which is 200.200.200.0.

| | | | | |
|---|---|---|---|---|
| **IP Address:** | 11001000 | 11001000 | 11001000 | 00000101 |
| **Subnet Mask:** | 11111111 | 11111111 | 11111111 | 00000000 |
| **Network Number** | 11001000 | 11001000 | 11001000 | 00000000 |



# Subnet Mask

| Class | IP Address Range | Subnet Mask |
|---|---|---|
| A | 1.0.0.0 to 126.255.255.255 | 255.0.0.0 |
| B | 128.0.0.0 to 191.255.255.255 | 255.255.0.0 |
| C | 192.0.0.0 to 223.255.255.255 | 255.255.255.0 |



# Subnet Mask

**Example 1:**

| | |
|---|---|
| **IP Address** | 200.200.10.50 |
| **Class** | **C** |
| **Subnet Mask** | 255.255.255.0 |
| **Network Number** | 200.200.10.0 |
| **Host Number** | 50 |

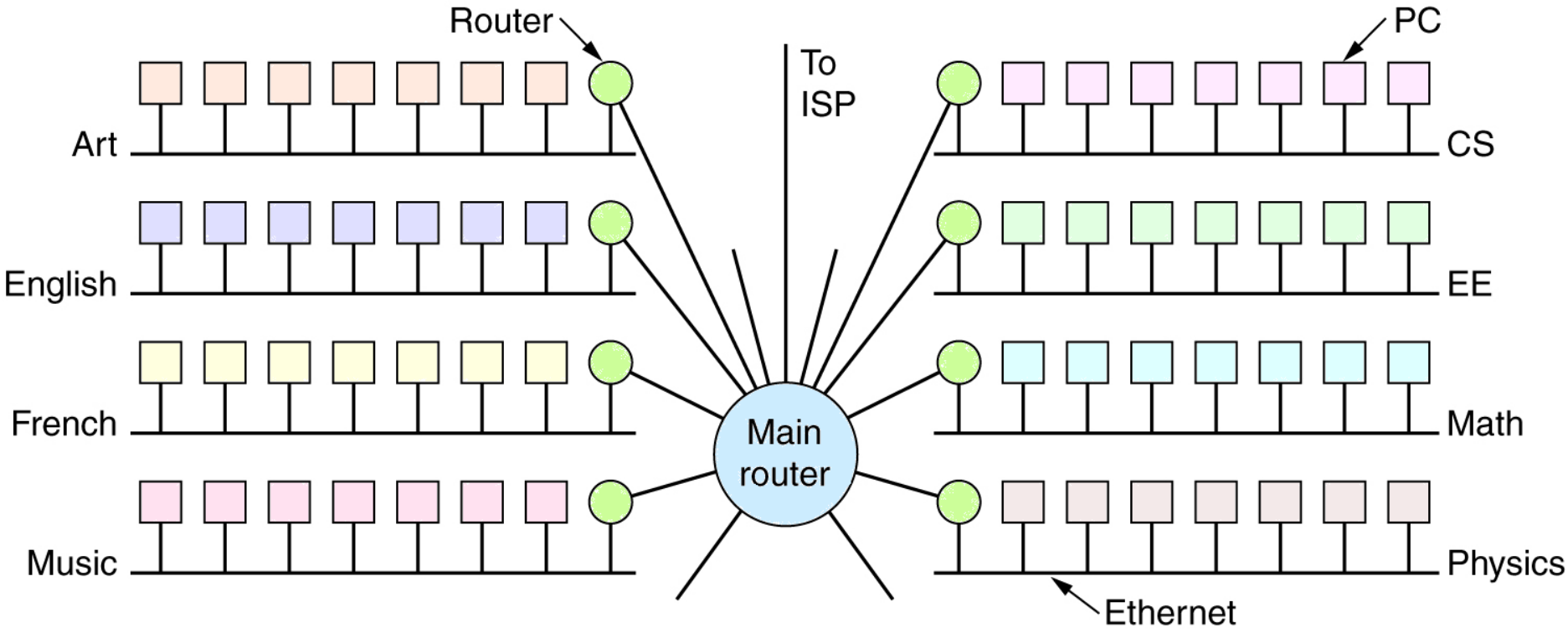
**Example 2:**

| | |
|---|---|
| **IP Address** | 150.50.10.20 |
| **Class** | **B** |
| **Subnet Mask** | 255.255.0.0 |
| **Network Number** | 150.50.0.0 |
| **Host Number** | 10.20 |

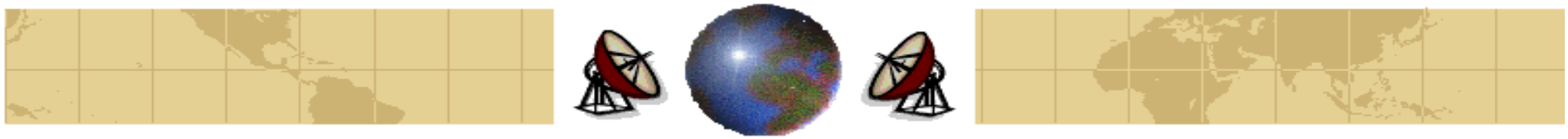
**Example 3:**

| | |
|---|---|
| **IP Address** | 125.30.20.1 |
| **Class** | **A** |
| **Subnet Mask** | 255.0.0.0 |
| **Network Number** | 125.0.0.0 |
| **Host Number** | 30.20.1 |

# Subnets




A campus network consisting of LANs for various departments

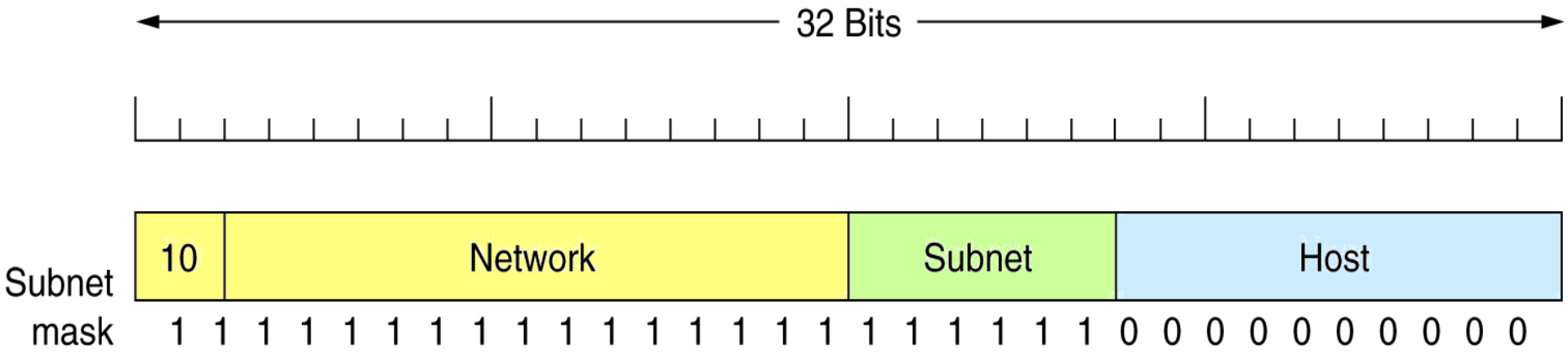


# Subnets

- IP address can cause problems as networks grow.
- For Example, consider a company that starts out with one class C LAN on the Internet. As time goes on, it might have more than 254 machines, and thus need a second class C address. Eventually, it might end up with many LANs, each with its own routers and each with its own class C network number.
- As the number of distinct local networks grows, managing them can become a serious headache such as:
  - Every time a new network is installed the system administrator has to contact ICANN to get a new network number. Then this number must be announced worldwide.
  - Moving a machine from one LAN to another requires it to change its IP address, which in turn may mean modifying its configuration files and also announcing the new IP address to the world.

# Subnets

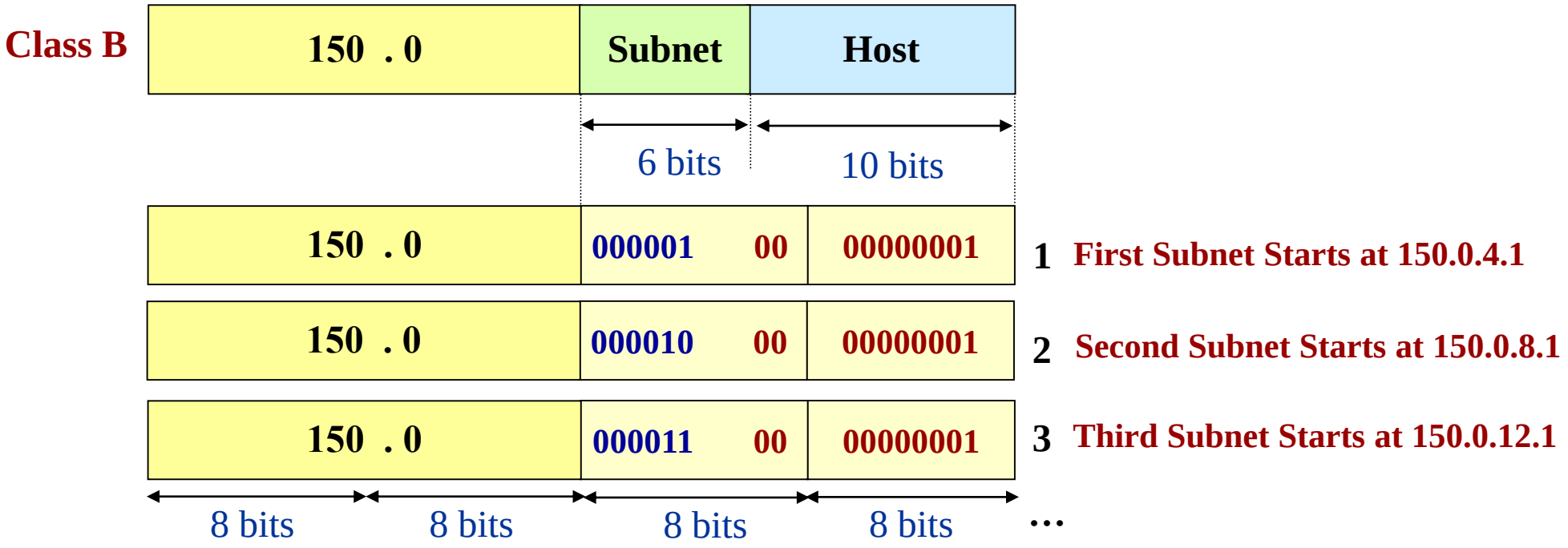
- The solution to these problems is to allow the network to split into several **subnets** for internal use but act like a single network to the outside world.
- If a growing company started up with a class B address instead of a class C address, the 16-bit host number is divided into a 6-bit subnet number and 10-bit host number. This split allows 62 LANs, each with up to 1022 hosts.




A class B network subnetted into 64 subnets

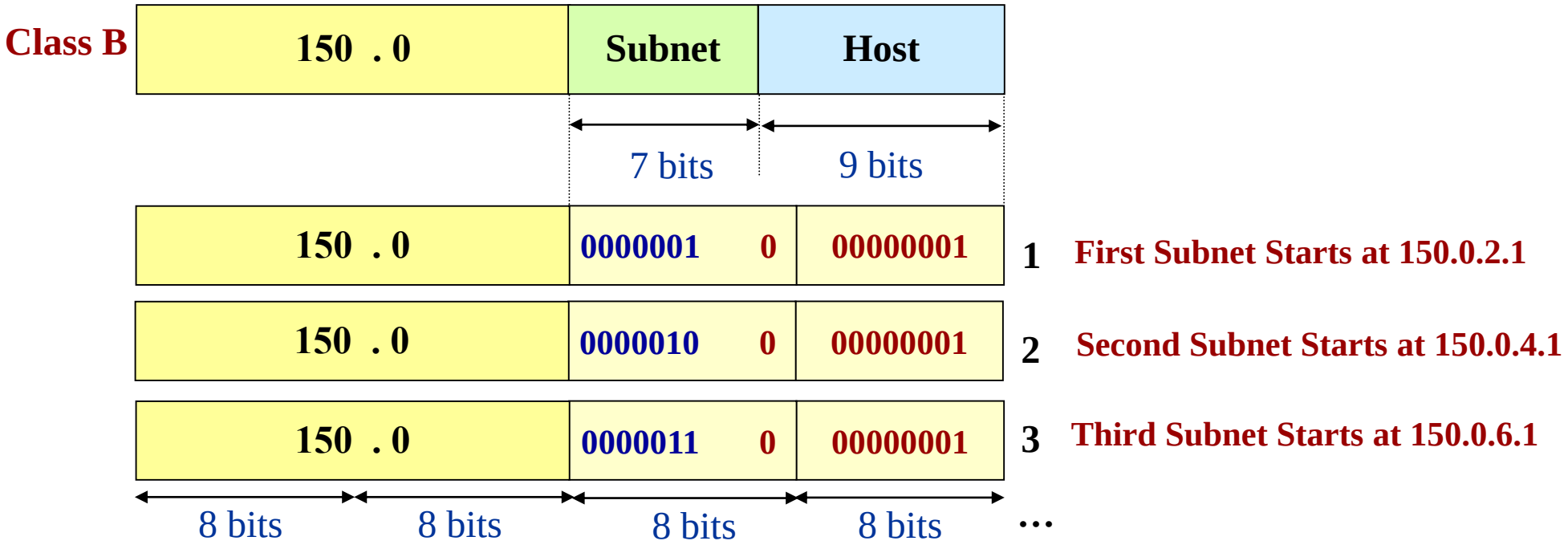
# Subnets

## Example 1:



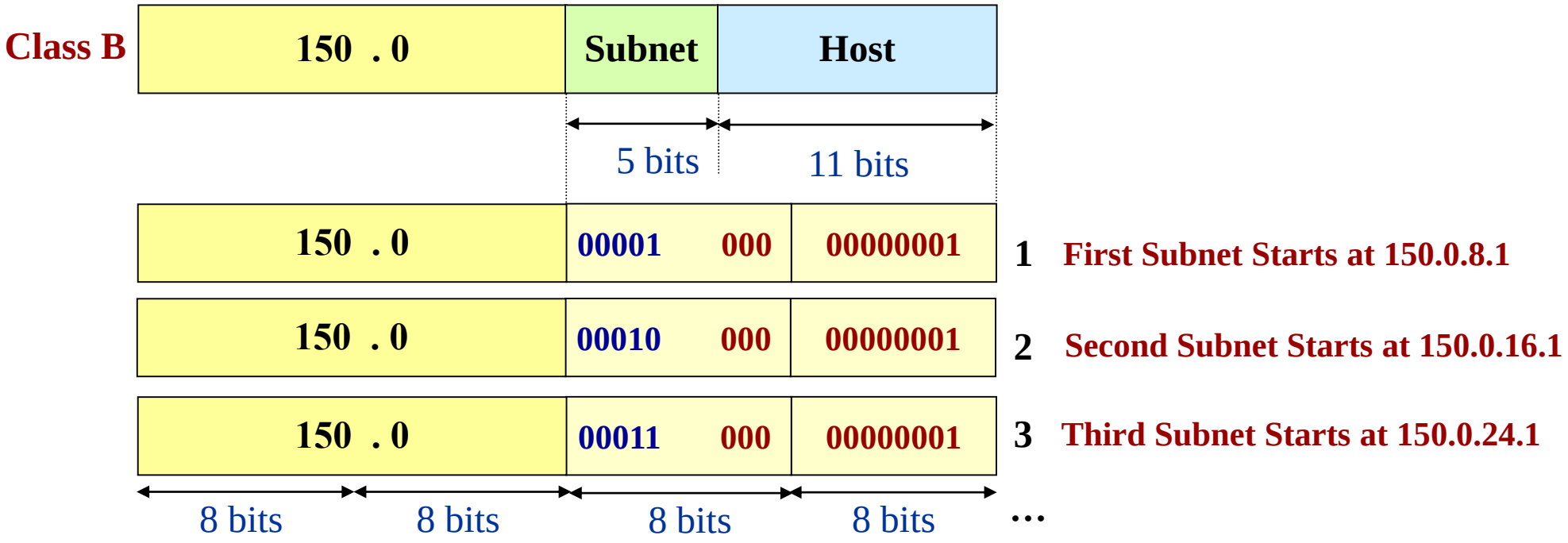
# Subnets

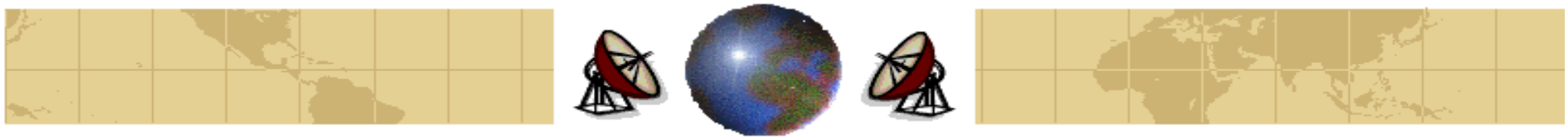
## Example 2:



# Subnets

## Example 3:



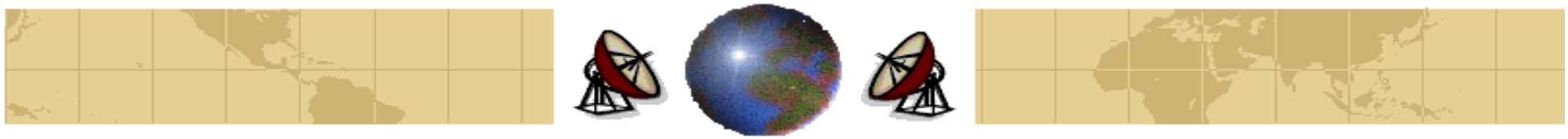


# Subnets

- In example 1, the first subnet might use IP addresses starting at 150.0.4.1, the second subnet might start at 150.0.8.1, and so on.
- Outside the network, the subnetting is not visible, so allocating a new subnet does not require contacting NIC or changing any external database.

**Processing Procedure of the IP Packets in the Router:**

- Each router has a table listing some number of **(network, 0)** IP addresses and some numbers of **(this network, host)** IP addresses. The first kind tells how to get to distant networks. The second kind tells how to get to local hosts.
- When an IP packet arrives, its destination address is looked up in the routing table. If the packet is for a distant network, it is forwarded to the next router on the interface given in the table. If it is local host (e.g., on the router's LAN), it is sent directly to destination.



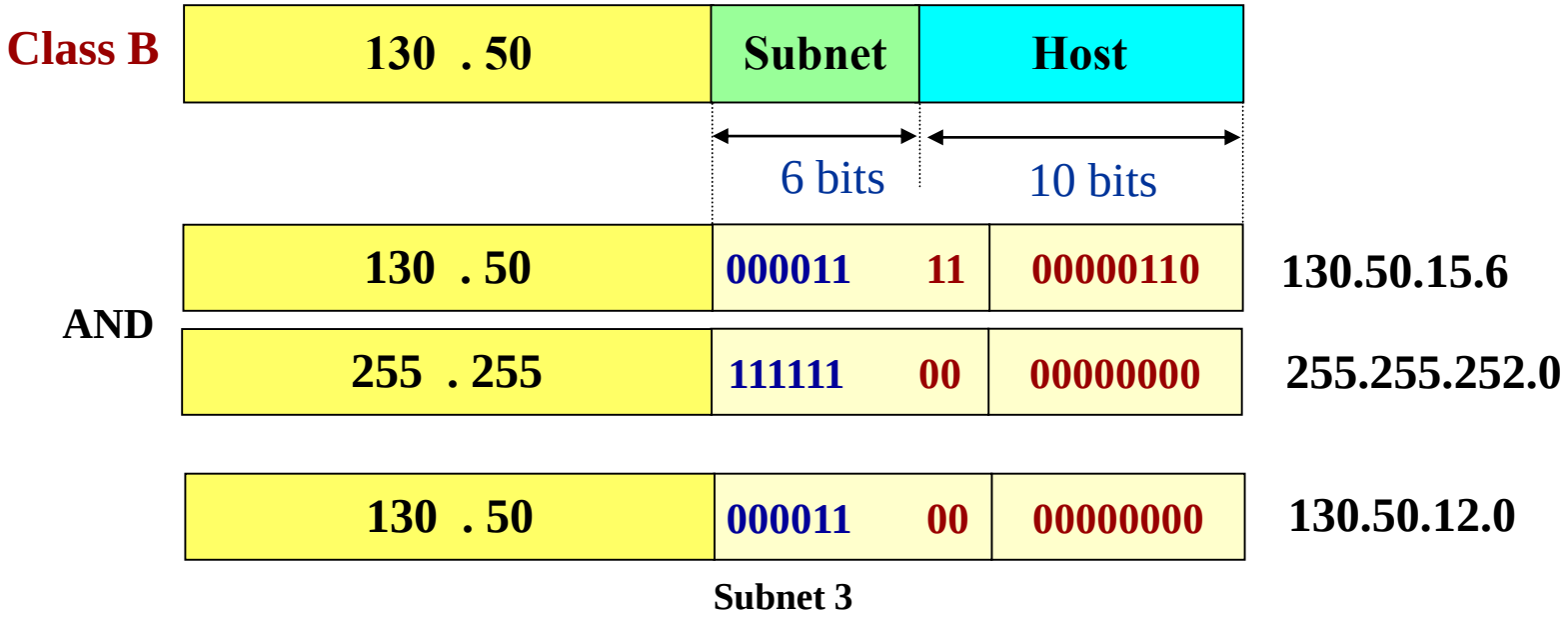
# Subnets

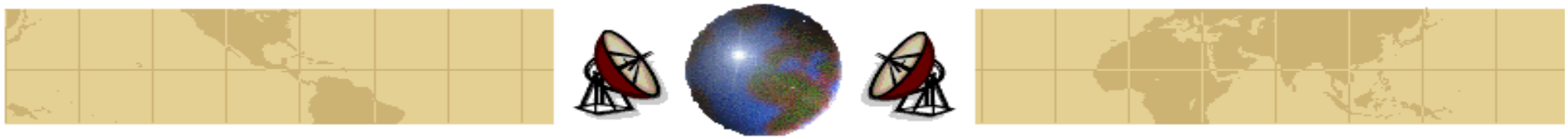
- The previous algorithm means that each router has to keep track of other networks and local hosts, not **(network, host)** pairs, greatly reducing the size of routing tables.
- When subnetting is introduced, the router tables are changed, adding entries of the form **(this-network, subnet, 0)** and **(this-network, this-subnet, host)**.
- Thus a router on subnet *k* knows how to get to all the other subnets and also how to get to all the hosts on subnet *k*. Each router does not know the details about the hosts on the other subnets.
- Each router performs a Boolean AND with the network's **subnet mask** to get rid of the host number and look up the resulting address in its tables.
- Subnetting reduces router table space by creating a three-level hierarchy.

# Subnets

## Example 4:

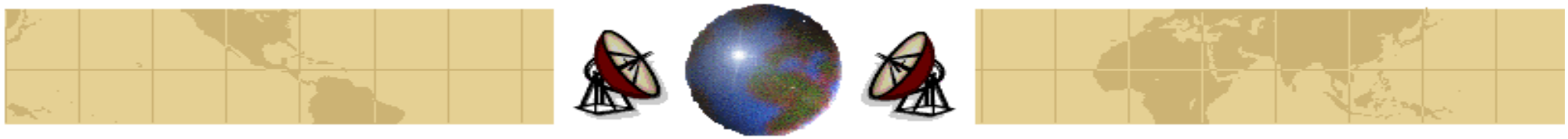
- A packet addressed to 130.50.15.6 and arriving at a router of subnet 5 is ANDed with the subnet mask (255.255.252.0) to give the address 130.50.12.0. This address is looked up in the routing tables to find out how to get to hosts on subnet 3.





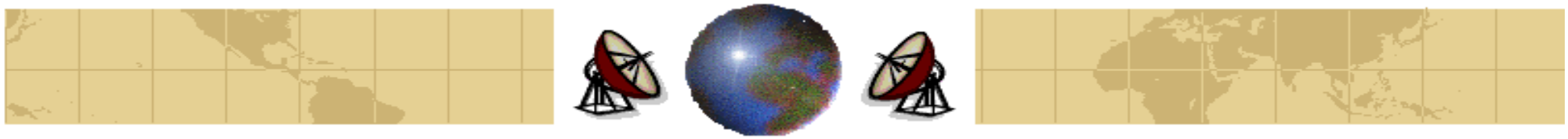
# CIDR – Classless InterDomain Routing

- The Internet is rapidly running out of IP addresses.
- For most organizations, a class A, with 16 million addresses is too big, and a class C network, with 254 addresses is too small. A class B network, with 65,536 is just right.
- In reality, class B address is far too large for most organizations. Studies have shown that more than half of all class B networks have fewer than 50 hosts. A class C network would have done the job.
- It might have been better to have had class C networks use 10 bits instead of 8 for the host number, allowing 1022 hosts per network. Had this been the case, most organization would have settled for a class C network, and there would have been half a million of them (versus only 16,384 class B networks).



# CIDR – Classless InterDomain Routing

- This causes routing table explosion. Routers have to know about all networks. If half a million class C networks were in use, every router in the entire Internet would need a table with half a million entries. In addition, various routing algorithms require each router to transmit its table periodically.
- One solution that is being implemented is **CIDR (Classless InterDomain Routing)**.
- The **basic idea** behind CIDR is to allocate the remaining class C networks in variable-sized blocks. If a site needs, say, 2000 addresses, it gives a block of 2048 addresses (eight contiguous class C networks), and not a full class B address.

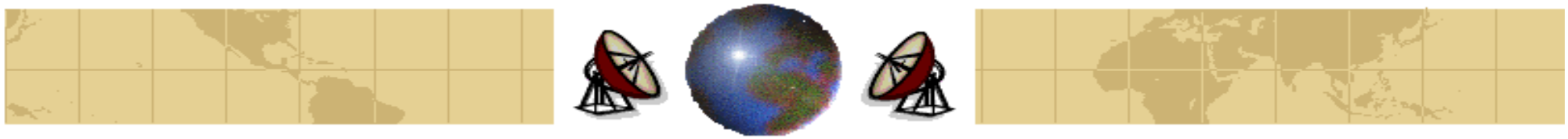


# CIDR – Classless InterDomain Routing

- The world was partitioned into four zones, and each one given a portion of the class C address space. The allocation was as follows:
  - Address 194.0.0.0 to 195.255.255.255 are for *Europe*.
  - Address 198.0.0.0 to 199.255.255.255 are for *North America*.
  - Address 200.0.0.0 to 201.255.255.255 are for *Central and South America*.
  - Address 202.0.0.0 to 203.255.255.255 are *Asia* and *Africa*.

- In this way, each region was given about 32 million addresses to allocate, with another 320 million class C addresses from 204.0.0.0 through 223.255.255.255 held in reserve for the future.

- The advantage of this allocation is that any router outside of Europe that gets a packet addresses to 194.xx.yy.zz or 195.xx.yy.zz can just send to its **standard European gateway**.



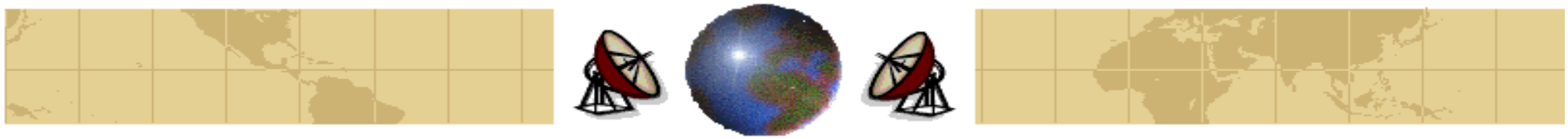
# CIDR – Classless InterDomain Routing

**Procedure**

- Each routing table entry is extended by a 32-bit mask.
- When a packet comes in, its destination address is first extracted.
- Then the routing table is scanned entry by entry, masking the destination address and comparing it to the table entry looking for a match.

**Example:**

- Suppose that Cambridge University needs 2048 addresses and is assigned the addresses 194.24.0.0 through 194.24.7.255 along with mask 255.255.248.0.
- Next, Oxford University asks for 4096 addresses, they get 194.24.16.0 through 194.24.31.255 along with mask 255.255.240.0.
- Now the University of Edinburgh asks for 1024 address and is assigned addresses 194.24.8.0 through 194.24.11.255 and mask 255.255.255.0.



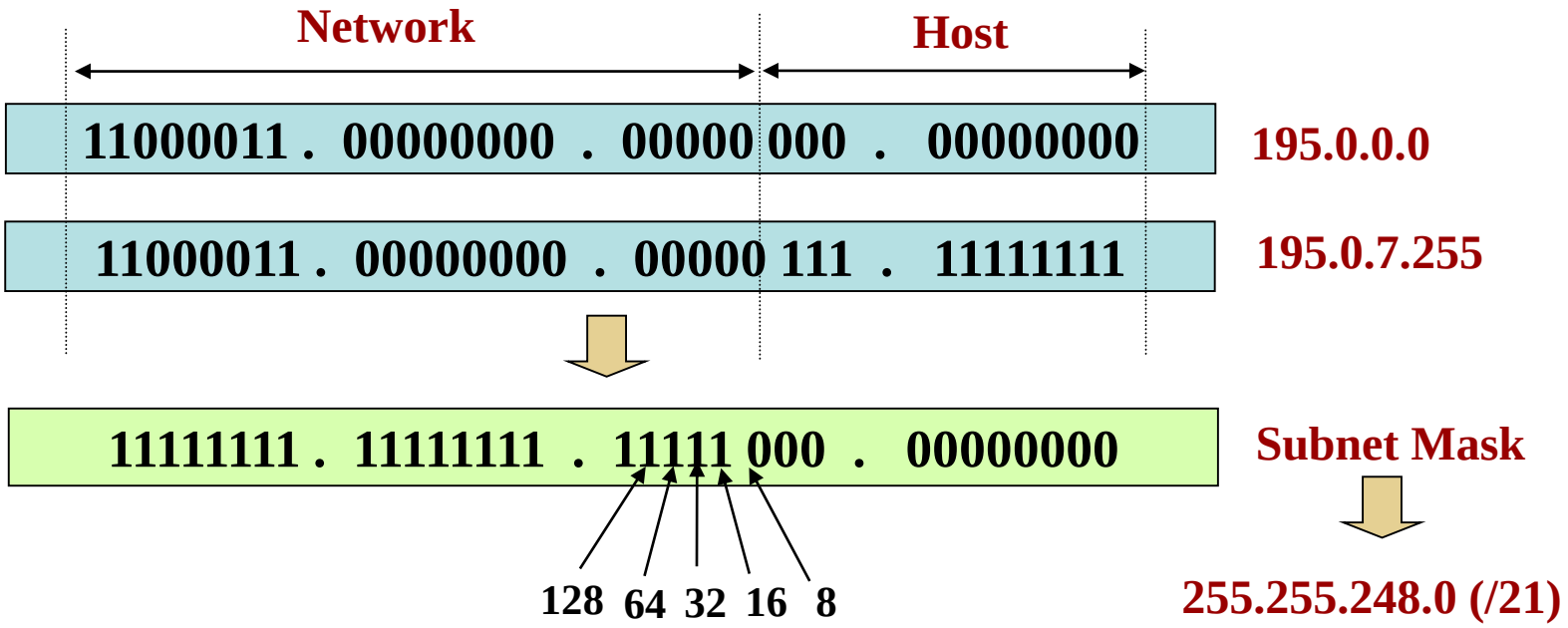
# CIDR – Classless InterDomain Routing

| University | First address | Last address | How many | Written as |
|---|---|---|---|---|
| Cambridge | 194.24.0.0 | 194.24.7.255 | 2048 | 194.24.0.0/21 |
| Edinburgh | 194.24.8.0 | 194.24.11.255 | 1024 | 194.24.8.0/22 |
| (Available) | 194.24.12.0 | 194.24.15.255 | 1024 | 194.24.12/22 |
| Oxford | 194.24.16.0 | 194.24.31.255 | 4096 | 194.24.16.0/20 |

A set of IP address assignments

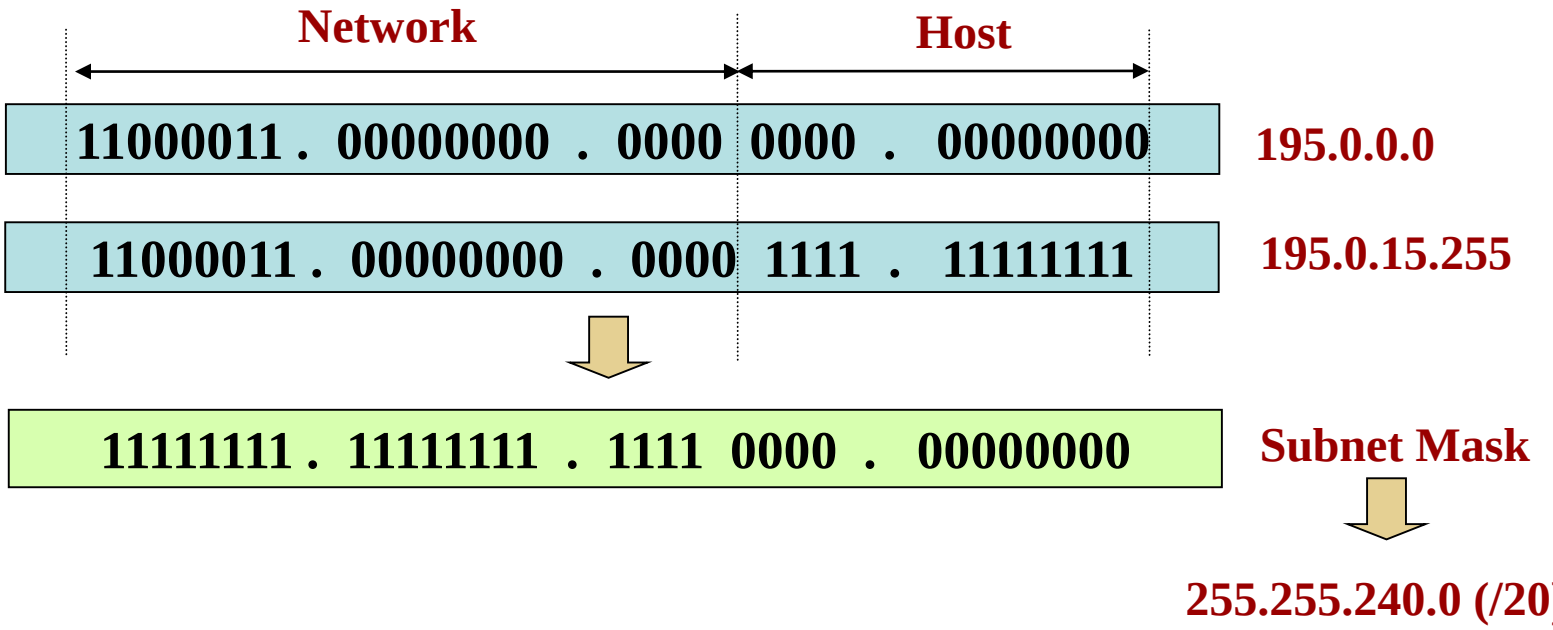
# CIDR – Classless InterDomain Routing

**Example 1:** 2046 Hosts are required, and starting IP address is 195.0.0.0.



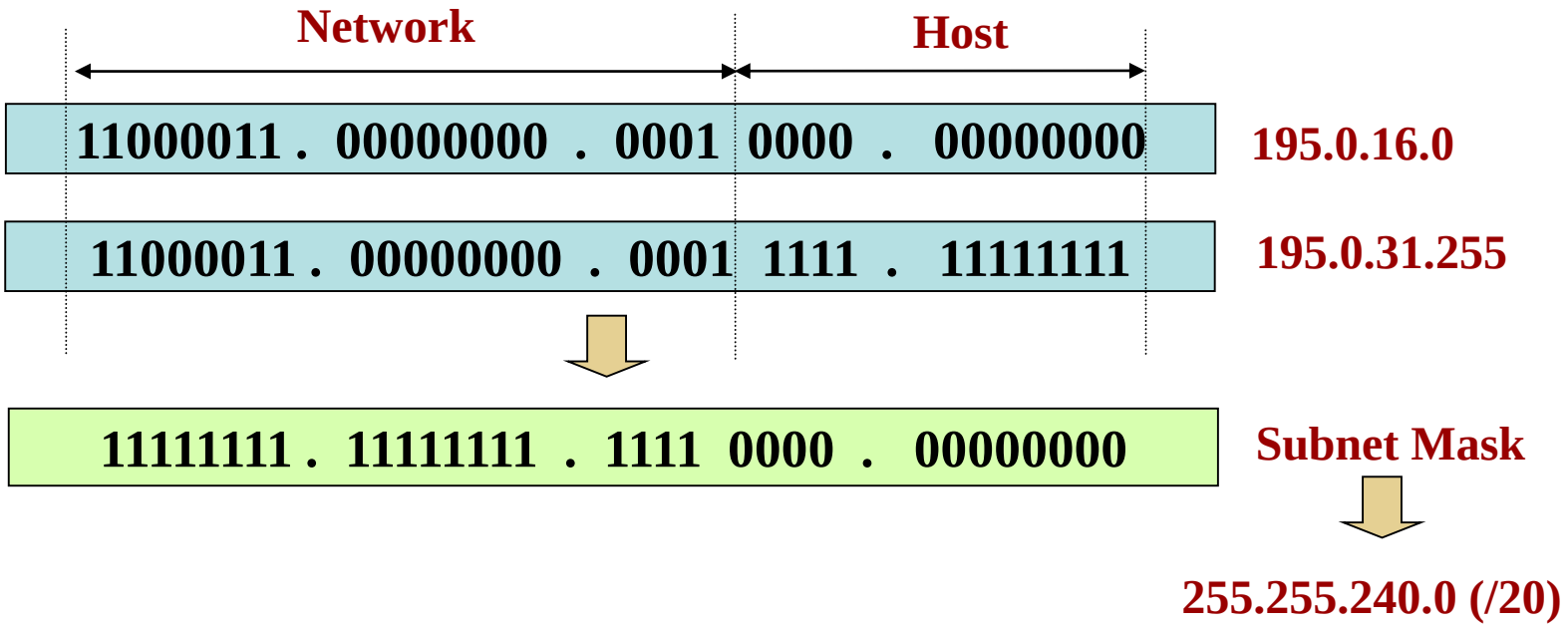
# CIDR – Classless InterDomain Routing

**Example 2:** 4094 Hosts are required, and starting IP address is 195.0.0.0.



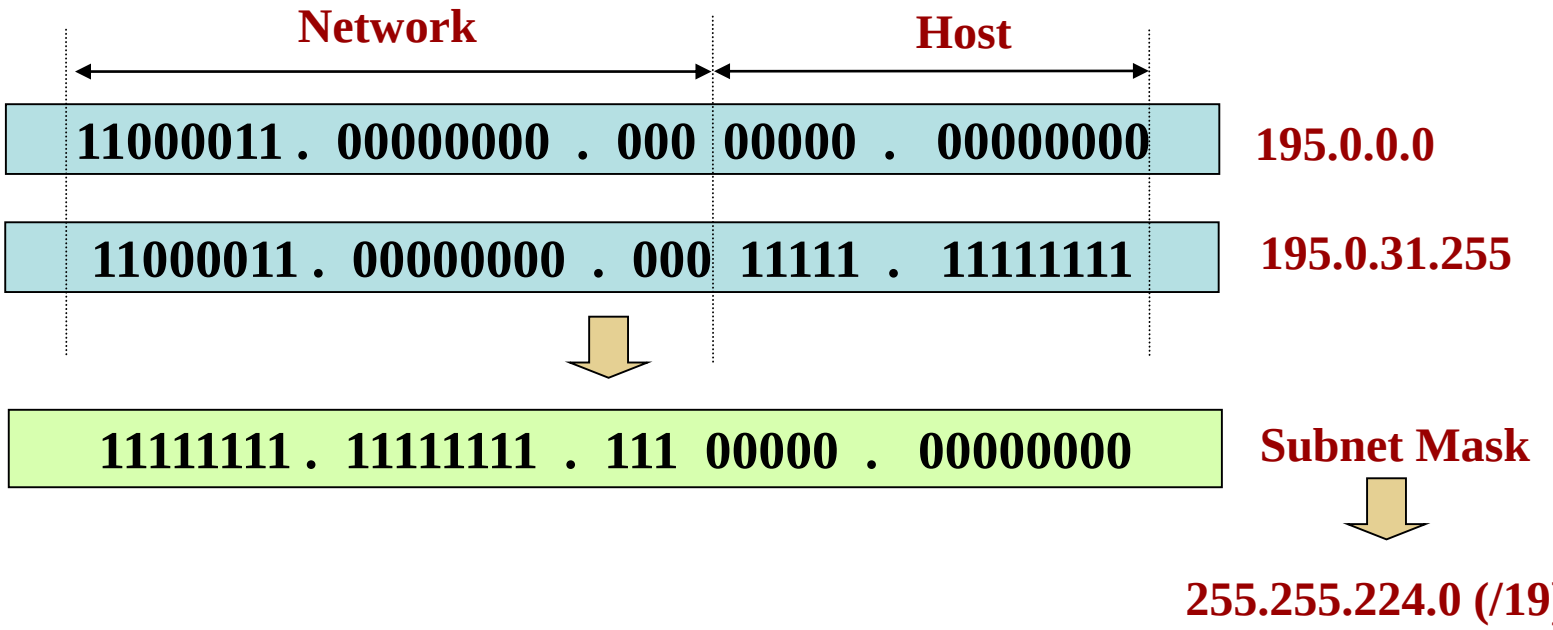
# CIDR – Classless InterDomain Routing

**Example 3:** 4094 Hosts are required, and starting IP address is 195.0.16.0.



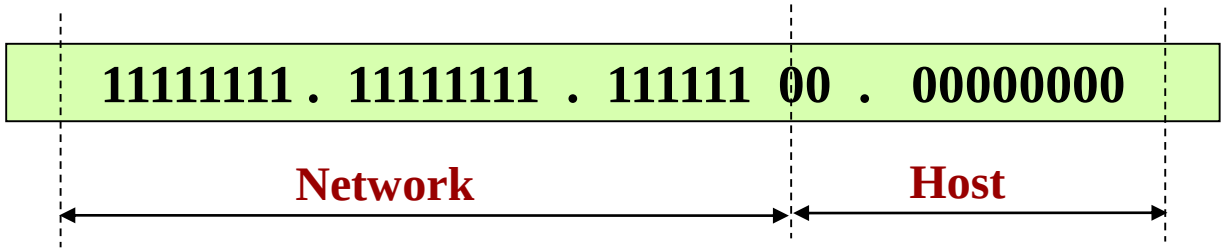
# CIDR – Classless InterDomain Routing

**Example 4:** 8190 Hosts are required, and starting IP address is 195.0.0.0.



# CIDR – Classless InterDomain Routing

**Example 5:** The *Subnet Mask* is /22.



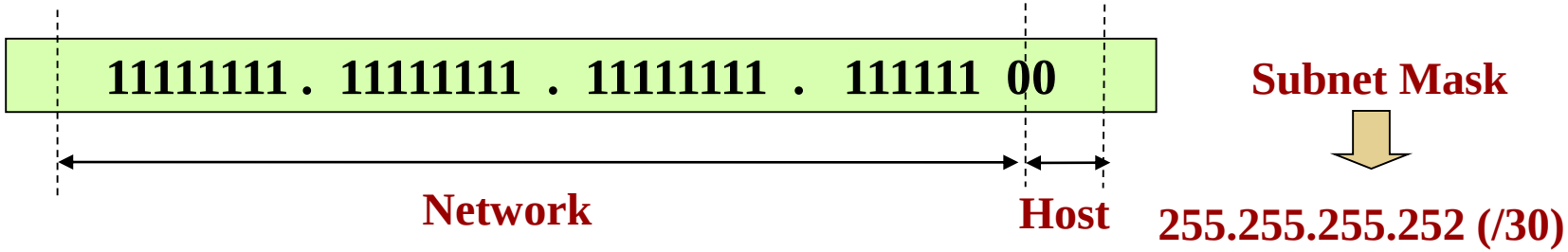

**Subnet Mask**

**255.255.252.0 (/22)**

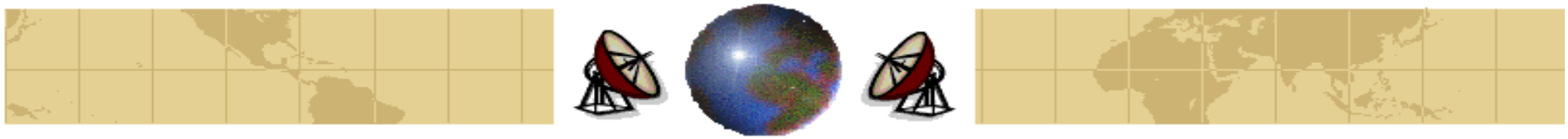
| | |
|---|---|
| **Subnet Mask** ***(Dotted Decimal Notation)*** | **255.255.252.0** |
| **Number of Hosts** | **$2^{10}$ - 2=1022 Hosts** |

# CIDR – Classless InterDomain Routing

**Example 6:** The *Subnet Mask* is /30.




| | |
|---|---|
| **Subnet Mask** ***(Dotted Decimal Notation)*** | **255.255.255.252** |
| **Number of Hosts** | **$2^2$ - 2=2 Hosts** |



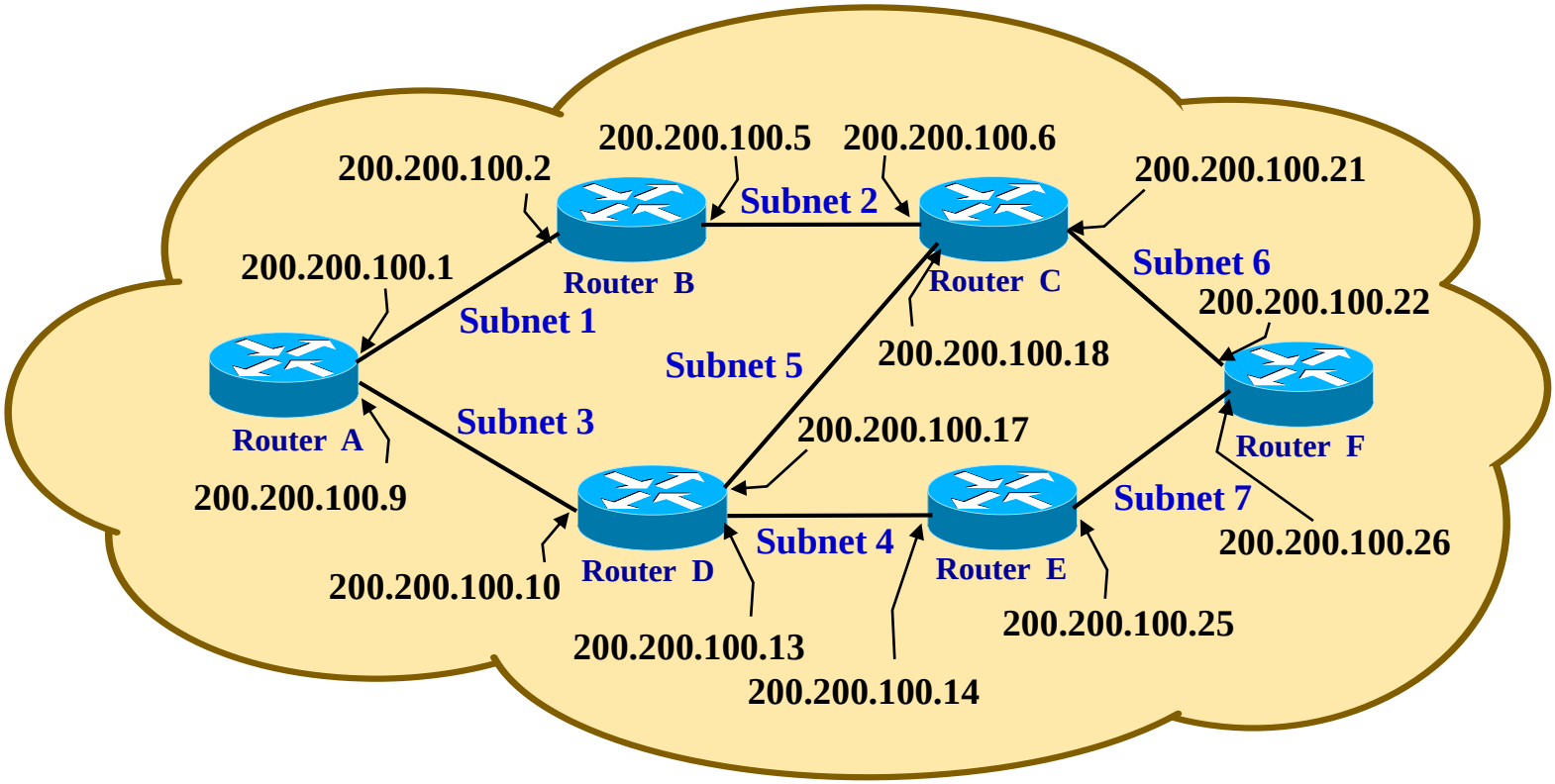
# CIDR – Classless InterDomain Routing

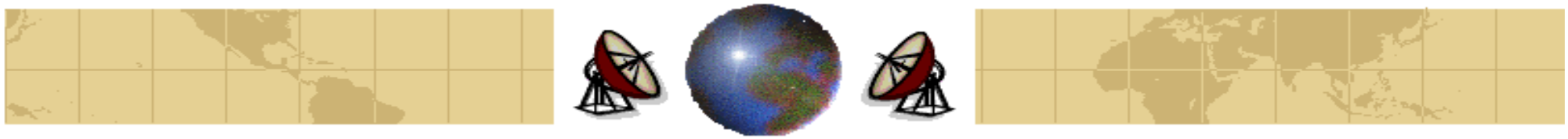
**Example 6:** *Continued.*

| | |
|---|---|
| **Address of the First Subnet** | **200.200.100.0** |
| **Address of the First Host Located in the First Subnet** | **200.200.100.1** |
| **Address of the Second Host Located in the First Subnet** | **200.200.100.2** |
| **-** | **200.200.100.3** |
| **Address of the Second Subnet** | **200.200.100.4** |
| **Address of the First Host Located in the Second Subnet** | **200.200.100.5** |
| **Address of the Second Host Located in the Second Subnet** | **200.200.100.6** |
| **-** | **200.200.100.7** |
| **Address of the Third Subnet** | **200.200.100.8** |
| **Address of the First Host Located in the Third Subnet** | **200.200.100.9** |
| **Address of the Second Host Located in the Third Subnet** | **200.200.100.10** |
| **-** | **200.200.100.11** |

# CIDR – Classless InterDomain Routing

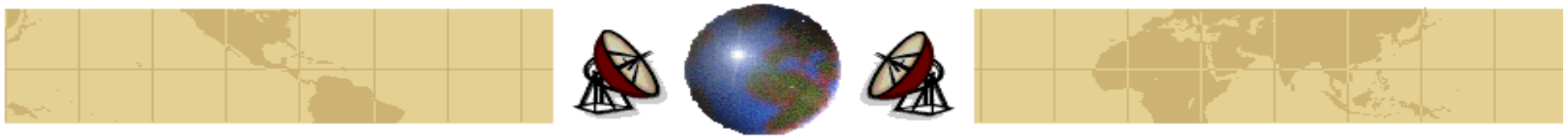
**Example 6:** *Continued.*





# *NAT – Network Address Translation*

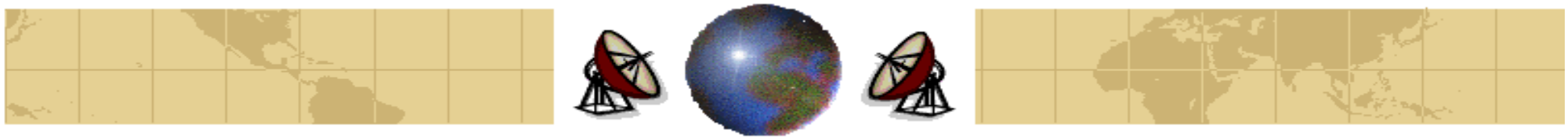
- IP addresses are scarce.
- An ISP might have a /16 address, giving it 65,534 host numbers. If it has more customers than that, it has a problem.
- For *home customers* with dial-up connections, the solution is to dynamically **assign** an IP address to a computer when it calls up and logs in, and **takes** the IP address back when the session ends. Therefore, a single /16 address can handle up to 65,534 *active* users.
- This strategy works well for an ISP with a moderate number of home users, but it fails for ISPs that serve *business customers*.



# *NAT – Network Address Translation*

- The *business customer* have multiple computers connected by a LAN. Generally, there is a router on the LAN that is connected to the ISP by a leased line to provide continuous connectivity. Therefore, each computer must have its own IP address all day long. In effect, *the total number of IP address owned by all it business customers combined cannot exceed the number of the IP addresses the ISP has*. This limits the total number of computers to 65,534.

- The problem of running out of IP addresses might occur in the near future. The solution is to migrate to **IPv6**, but this transition is slowly occurring. A quick fix is came in the form of **NAT (Network Address Translation)**.

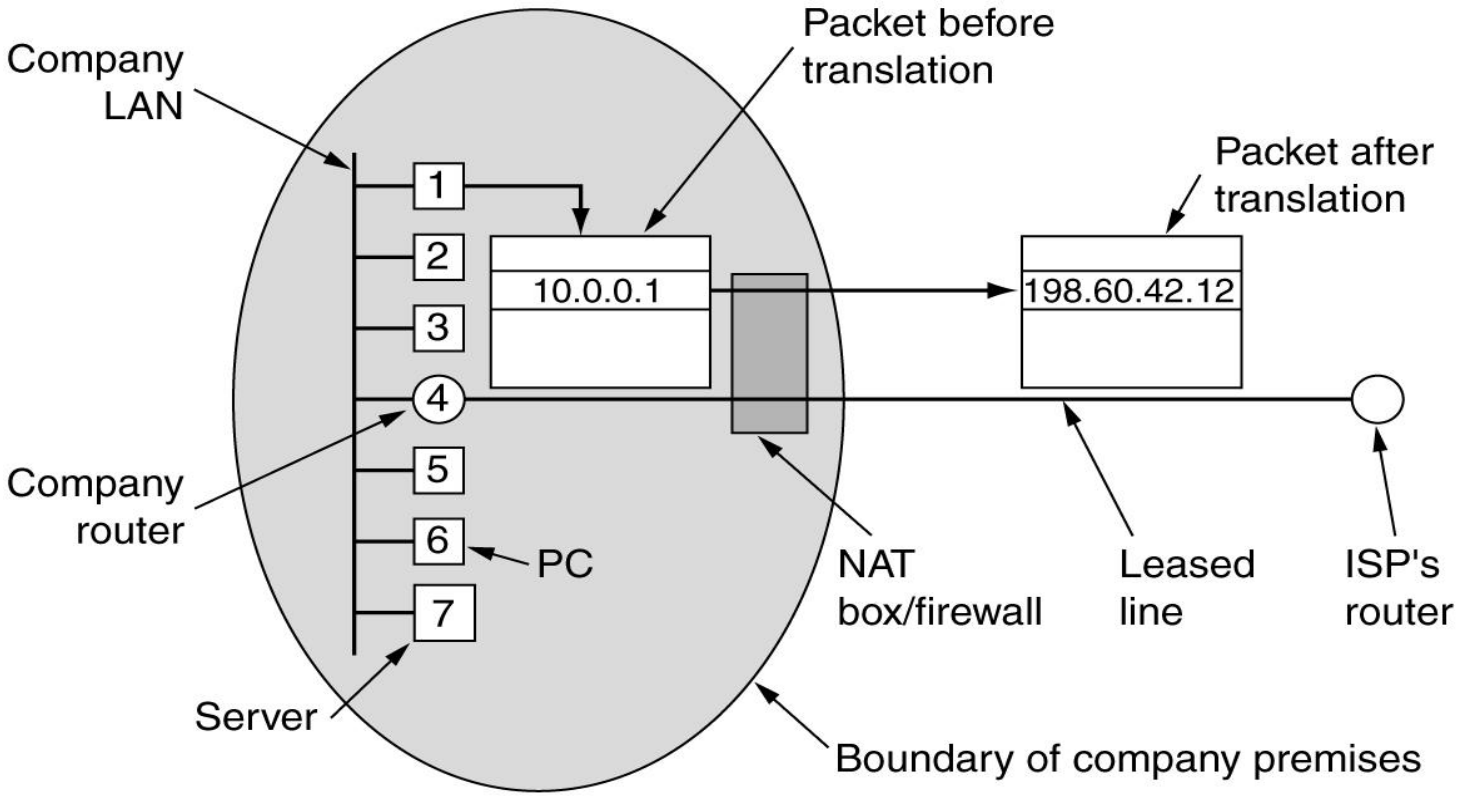


# *NAT – Network Address Translation*

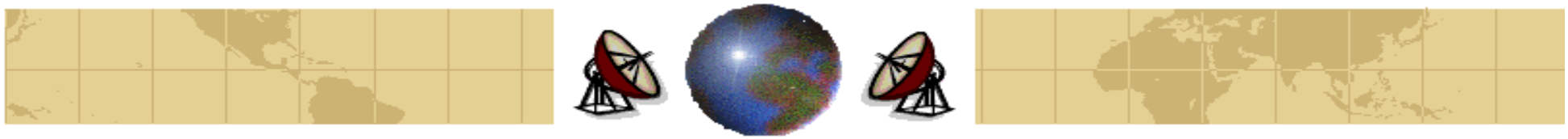
- The basic idea behind NAT is to assign each company *a single IP address or small number of IP addresses* for *Internet traffic*. Within the company, every computer gets a unique IP address, which is used for the *internal traffic*. However when a packet exits the company and goes to the ISP, an address translation takes place.

- To make this scheme possible, *three ranges* of IP addresses have been declared as **private**. Companies may use them internally as they which. The three reserved ranges are:
  - **10.0.0.0 – 10.255.255.255/8 (16,777,216 hosts).**
  - **172.16.0.0 – 172.31.255.255/16 (1,048,576 hosts).**
  - **192.168.0.0 – 192.168.255.255/24 (65,536 hosts).**

# NAT – Network Address Translation

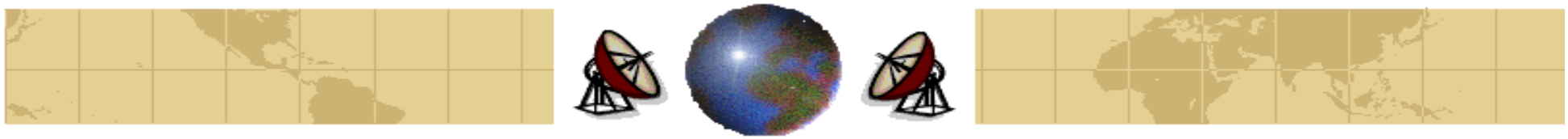



Placement and operation of a NAT box



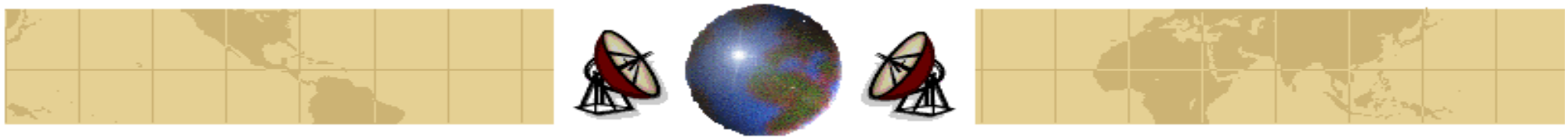
# NAT – Network Address Translation

- The *operation* of NAT is:
  - Within the company premises, every machine has a unique address of the form 10.*x.y.z*.
  - When a packet leaves a company premises, it passes through a **NAT box** that converts the internal source IP address, 10.0.0.1 in the figure, to the company's true IP address, 198.60.42.12.
- The NAT box is often *combined* with the **firewall**.



# NAT – Network Address Translation

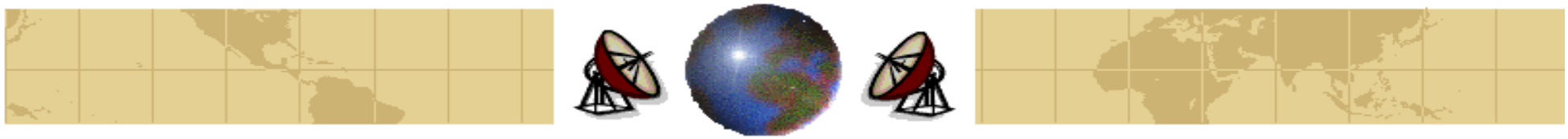
- When **reply** comes back (e.g., form a web server), it is addressed to 198.60.42.12, so *how does the NAT box know which address to replace it with?* Note that there is no spare field in the IP packet to keep track of who the real sender are. Remember, we need a quick fix of the problem of running out of IP addresses.
- The actual solution is related to the *source port* and the *destination port* of the TCP/UDP headers.
- The ports are 16-bit integers that indicate where the *TCP connection begins and ends*. These fields provide the field needed to make NAT works.



# NAT – Network Address Translation

## Procedure for a TCP Connection:

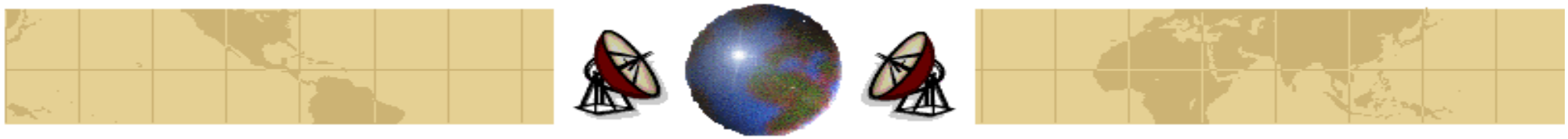
- When a process wants to establish a TCP connection with a remote process, it attaches itself to an unused port on its own machine. This is called the **source port** and *tells the TCP code where to send incoming packets belonging to this connection*.
- The process also supplies a **destination port** to *tell who to give this packets to on the remote side.*
- Together, a source port and a destination port serve to *identify the processes using the connection on both ends*.



# *NAT – Network Address Translation*

- Using the source port field, we can solve our mapping problem. Whenever an *outgoing packet* enters a NAT box, the 10.*x.y.z* source address is replaced by the company's true address (198.60.42.12). In addition, the TCP source port field is replaced by an *index* into the NAT box's 65,536-entry translation table. The table entry contains the **original IP address** (10.*x.y.z*) and the original the **original source port**.

- When a packet arrives at the NAT box from the ISP, the source port in the TCP header is extracted and used as *index* into the NAT box's mapping table. From the entry located, the **internal IP address** (10.*x.y.z*) and the **original TCP source port** are extracted and inserted in the packet.



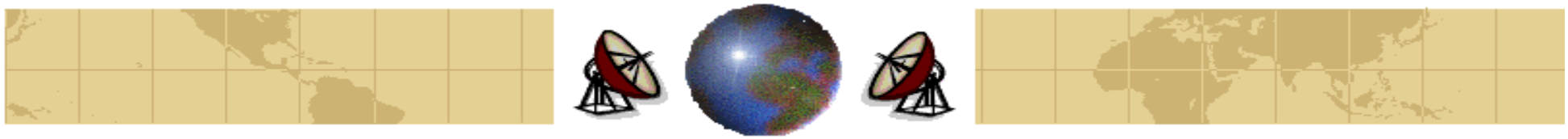
# NAT – Network Address Translation

**Objections of the IP Community against NAT:**

1. NAT violates the architectural model of IP, which *states that every IP address uniquely identifies a single machine worldwide*.

2. NAT changes the internet from a connectionless network to a kind of connection-oriented network. *The NAT box maintains information for each connection passing through it* (This is a property of connection oriented networks). If the NAT box crashes, all its TCP connections are destroyed.

3. NAT violates the protocol layering: *Layer k may not make any assumption about what layer k+1 has put into the payload field*. This is to keep the layers independent. If TCP is upgraded to TCP-2, with different header layout, NAT will fail. The whole idea of layered protocols is to *ensure that changes in one layer do not require changes in other layers*.



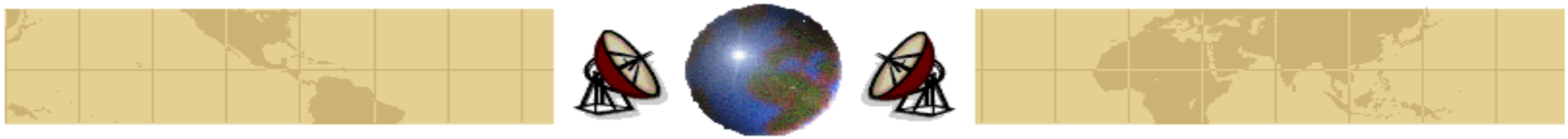
# NAT – Network Address Translation

**Objections of the IP Community against NAT:**

4. Processes on the Internet are not required to use TCP or UDP. Therefore, introduction of a NAT box will cause the application to fail because the NAT box will not be able to locate the *TCP Source Port* correctly.

5. Some applications insert IP addresses in the body of the text. The receiver then extracts these IP addresses and uses them. Since NAT cannot replace them, so any attempt to use them on the remote side will fail (e.g., FTP, H.323 Internet Telephony protocol, etc.) It is possible to patch NAT to work with an application, *but having to patch the code in the NAT box for every new application is not a good idea*.

6. Since the *TCP Source Port* is 16 bits, at most 65,536 machines can be mapped onto an IP address. Actually, the number is less because of the reserved ports.



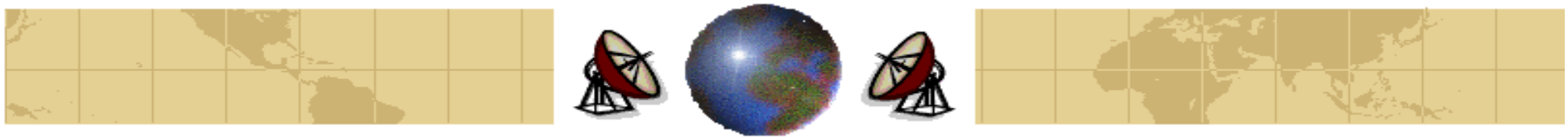
# Internet Control Message Protocol

## The Internet Control Message Protocol:

- When something unexpected occurs, the event is reported by the ICMP (Internet Control Message Control).
- Many ICMP messages are defined.
- Each ICMP message is encapsulated in an IP packet.

| Message type | Description |
|---|---|
| Destination unreachable | Packet could not be delivered |
| Time exceeded | Time to live field hit 0 |
| Parameter problem | Invalid header field |
| Source quench | Choke packet |
| Redirect | Teach a router about geography |
| Echo request | Ask a machine if it is alive |
| Echo reply | Yes, I am alive |
| Timestamp request | Same as Echo request, but with timestamp |
| Timestamp reply | Same as Echo reply, but with timestamp |

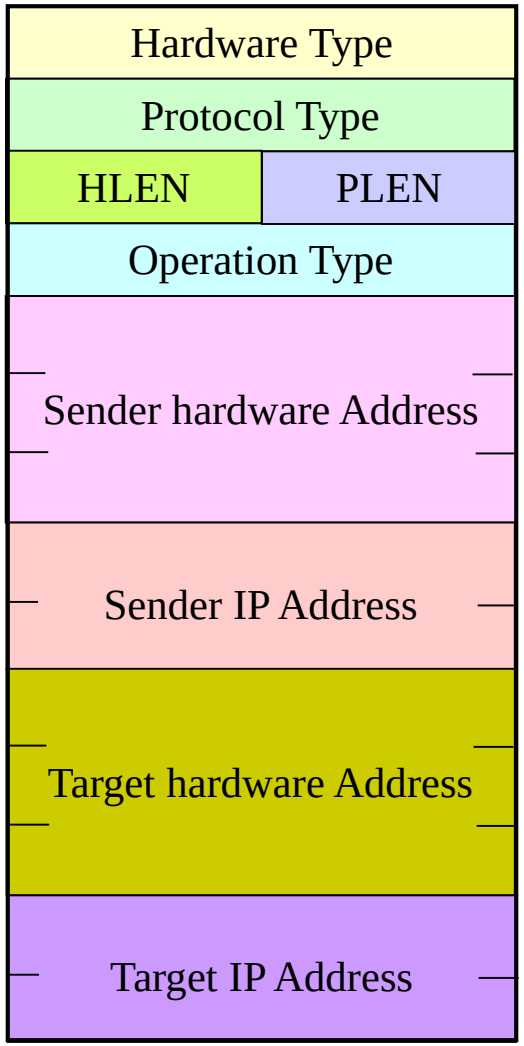
The principal ICMP message types



# ARP– The Address Resolution Protocol

- An IP address is assigned for every machine on the Internet.
- The IP addresses cannot be used for sending packets because the data link layer hardware does not understand Internet addresses.
- Most hosts are attached to a LAN by an interface board that only understands LAN addresses.
- For example, every Ethernet board is equipped with 48-bit Ethernet address. Manufacturers of Ethernet boards request a block of addresses from a central authority to ensure that *no two boards have the same address*. The reason of that is to avoid conflicts when two boards ever appear on the same LAN.
- The boards send and receive frames *based on 48-bit Ethernet addresses*. They *know nothing about 32-bit IP addresses*.

# ARP and RARP Message Format




**HLEN:** Hardware Address Length

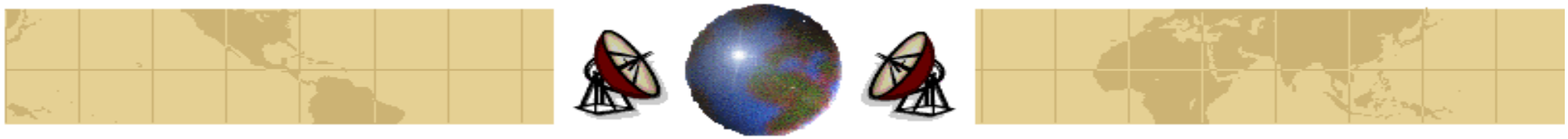
**PLEN:** IP Address Length

**Operation** = 1 ARP Request

= 2 ARP Response

= 3 RARP Request

= 4 RARP Response



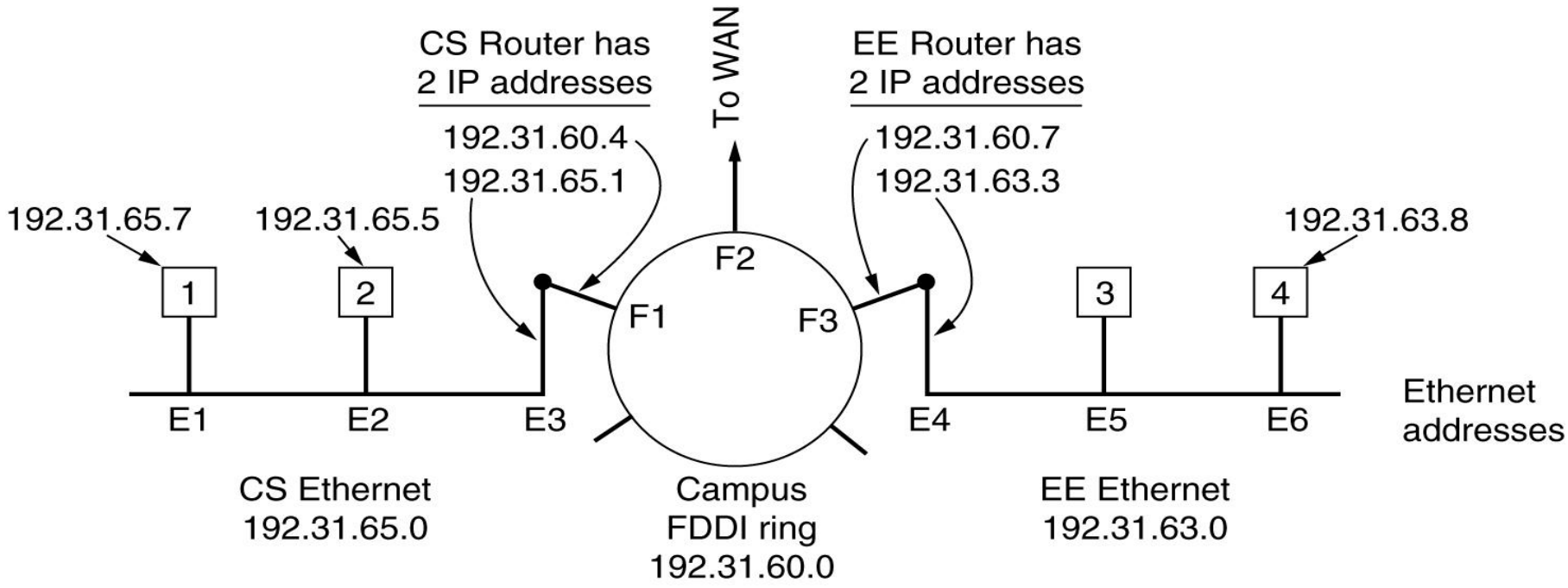
# ARP– The Address Resolution Protocol

**How do IP addresses get mapped onto data link layer addresses?**

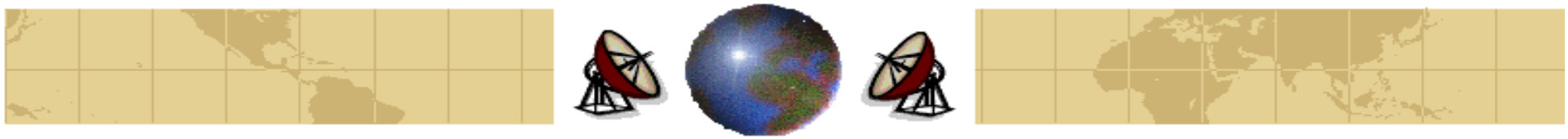
**Example:**

- Assume a small university has three class C networks:
  - The first one is Ethernet and used in the Computer Science department with IP address 192.31.65.0.
  - The second one is also Ethernet and used in the Electrical Engineering department with IP address 192.31.63.0.
  - The previous networks are connected by campus FDDI ring with IP address 192.31.60.0.
  - Each machine on an Ethernet has a unique Ethernet address labeled *E1* through *E6*, and each machine on the FDDI ring has an FDDI address, labeled *F1* through *F3*.

# ARP– The Address Resolution Protocol




Three interconnected /24 networks: two Ethernets and an FDDI ring



# ARP– The Address Resolution Protocol

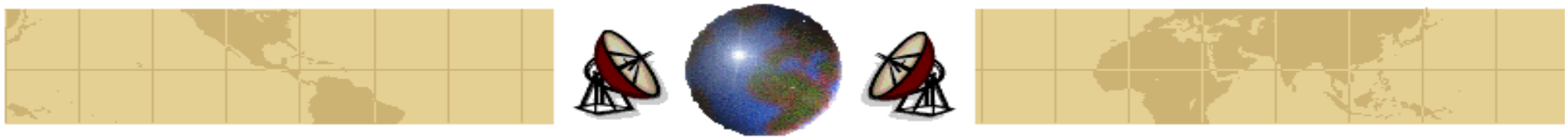
**First: Sending a Packet from a User on Host 1 to a User on Host 2.**

1. The sender knows the name of the intended receiver (e.g., *name@.ccis.ksu.edu.sa*).

2. Find the IP address for the host 2 (*ccis.ksu.edu.sa*). This lookup is performed by the **Domain Name System** (**DNS**) and returns the IP address for host 2 (192.31.65.5).

3. The upper layer software on host 1 now builds a packet with 192.31.65.5 in the *Destination Address* field and gives it to IP address to transmit.

4. The IP software can look at the address and see that the destination is on its network, but it needs to *find the destination's Ethernet address*.



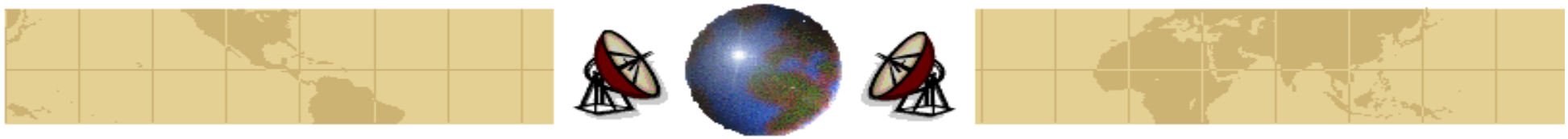
# ARP– The Address Resolution Protocol

5. One solution is to have a configuration file that maps IP address onto Ethernet addresses. This solution is possible, but no for organizations with thousands of machines.

6. The better solution is for host 1 to output a broadcast packet onto the Ethernet asking: "*Who owns IP address 192.31.65.5*?" The **broadcast** will arrive on every machine on Ethernet 192.31.65.5, and *each will check its IP address*. *Host 2 alone will respond with its Ethernet address E2*. The protocol for asking this question and getting the reply is called APR (Address Resolution Protocol).

7. At this point, the IP software in host 1 builds an Ethernet frame addressed to *E2*, puts IP packet addressed to192.31.65.5 in the payload field, and dumps it onto the Ethernet.

8. The Ethernet board on host 2 detects this frame. The Ethernet driver extracts the IP packet from the payload and passes it to the IP software.



# ARP– The Address Resolution Protocol

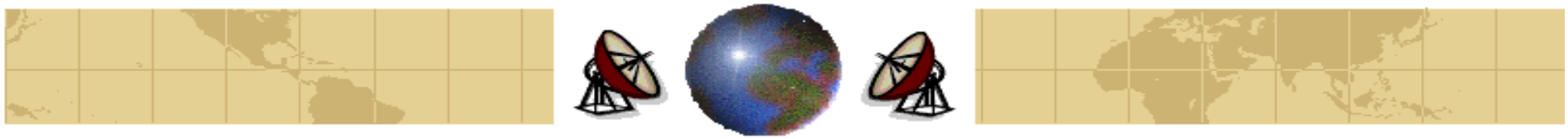
**Optimizations of ARP:**

1. *Caching ARP results in case the host needs to contact again the same host shortly*. Thus this optimization eliminates the need for a second broadcast.

2. In many cases, host 2 will need to send back a reply. Therefore, it needs to run ARP to determine the sender's Ethernet address. This ARP broadcast can be avoided by having host 1 includes its IP to Ethernet mapping in the ARP packet. *When ARP broadcast arrives at host 2, the pair (192.31.67.7, E1) is entered into host 2's ARP cache for future use*.

3. Every machine broadcasts its mapping (IP address, Ethernet address) when it boots. There should not be response, but the side effect of the broadcast is to make an entry in everyone's ARP cache. If a response arrives, two machines have been assigned the same IP address.

- To allow mappings to change, for example, when an Ethernet board breaks and is replaced by a new one, entries in the ARP cache should time out periodically.



# ARP– The Address Resolution Protocol

**Second: Sending a Packet from a User on Host 1 to a User on Host 6.**

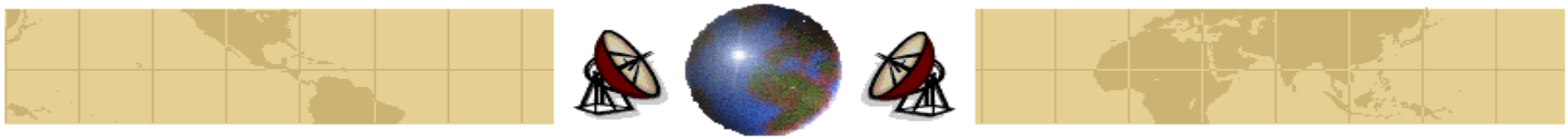
- Using *ARP will fail because routers do not forward Ethernet-level broadcasts*. There are two solutions:

**First Solution: Proxy ARP:**

- Routers are configured to respond to ARP requests for other local networks. For example, the CS router responds to ARP request for network 192.31.63.0.

**Second Solution:**

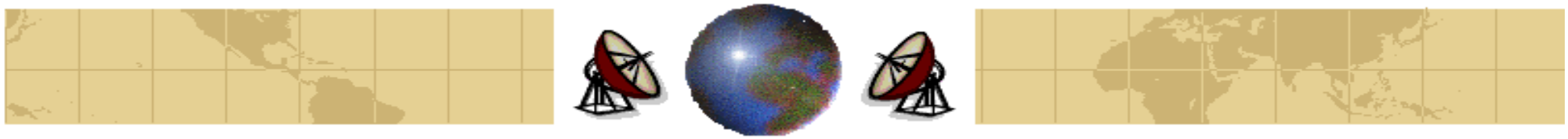
- When a source host sees that the destination is on a remote network, it sends all such traffic to a *default Ethernet address* that handles all remote traffic, in this case *E3*.



# ARP– The Address Resolution Protocol

**Procedure for Sending a Packet from a User on Host 1 to a User on Host 6:**

1. Host 1 packs the IP packet into the payload field of an Ethernet frame addressed to *E3*.
2. When the CS router gets the Ethernet frame, it removes the IP packet from the payload field and look up the IP address in it routing table.
3. It discovers that packets for network 192.31.63.0 are supposed to go to router 192.31.60.7. If it does not know the FDDI address of 192.31.60.7, it *broadcasts an ARP packet onto the ring and learns that its ring address is F3*.
4. It then inserts the packet in the payload field of an *FDDI frame* addresses to *F3* and puts it on the ring.



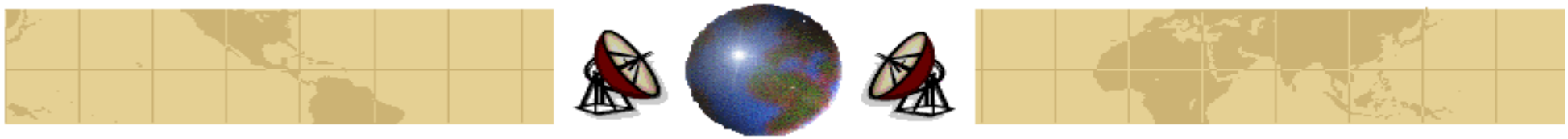
# ARP– The Address Resolution Protocol

5. At the EE router, the FDDI driver removes the packet form the payload field and gives it to IP software, which sees that it needs to send the packet to 192.31.63.8.

6. If this IP address is not in the *ARP cache*, the *EE router broadcasts an ARP request on the EE Ethernet and learns that the destination address is E6.*

7. The EE router builds an Ethernet frame addresses to *E6*, puts the packet in the payload field, and sends it over the Ethernet.

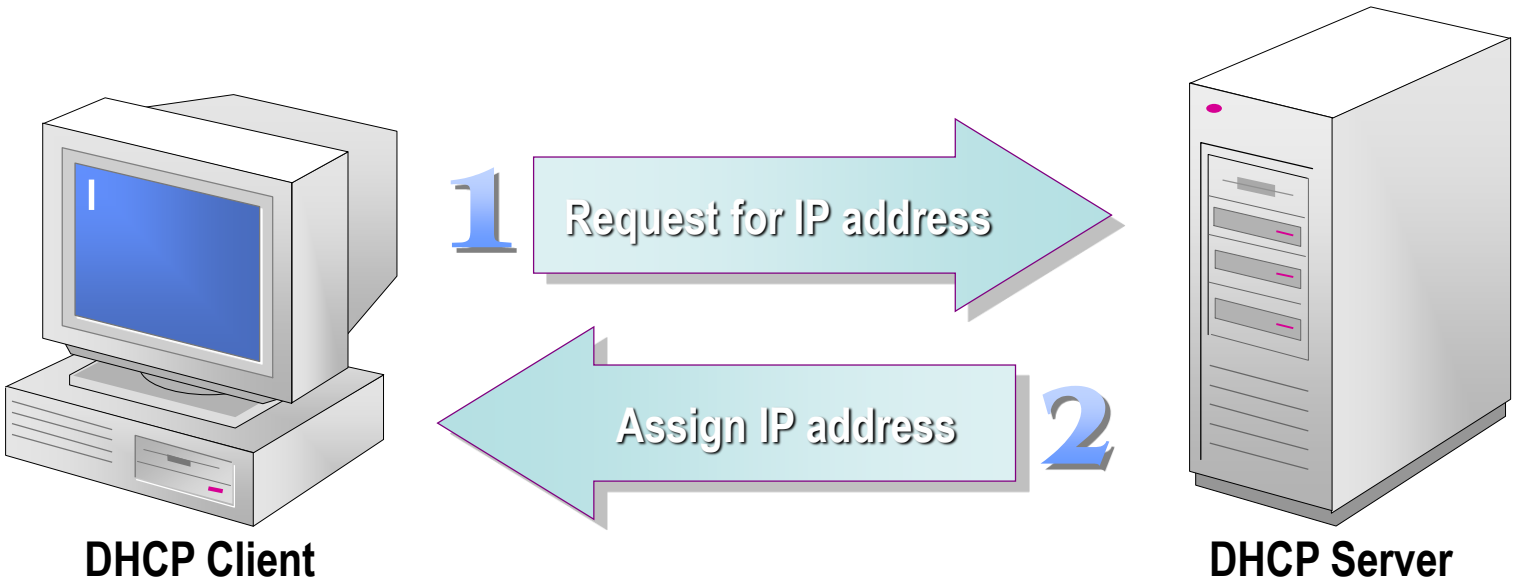
8. When Ethernet frame arrives at host 4, the packet is extracted from the frame and passed to the IP software for processing.

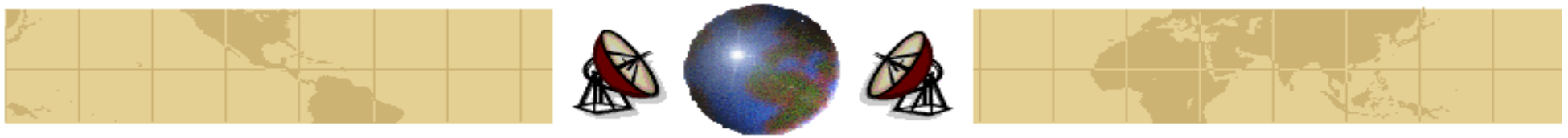


# *RARP– The Reverse Address Resolution Protocol*

- Sometimes the reverse problem has to be solved: Given an Ethernet address, what is the corresponding IP address.
- This problem occurs when booting a diskless workstation. Such a machine will get the *binary image of its operating system from a remote file server*. *But how does it learn its IP address*?
- The solution is to use the **RARP** (**Reverse Address Resolution Protocol**). This protocol allows a newly-booted workstation to broadcast its Ethernet address and say: "*My 48-bit Ethernet address is 14.04.05.18.01.25. Does anyone out there know my IP address*?“
- The **RARP server** sees this request, *looks up the Ethernet address in its configuration files, and sends back the corresponding IP address*.

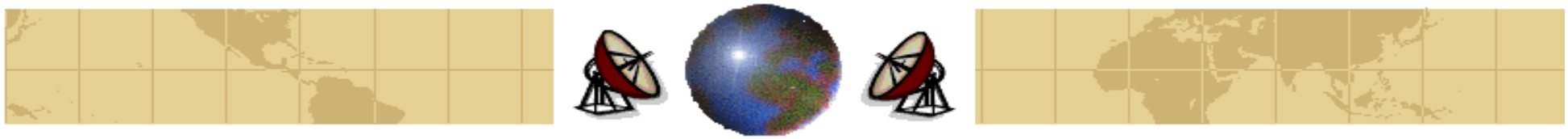
# Dynamic Host Configuration Protocol



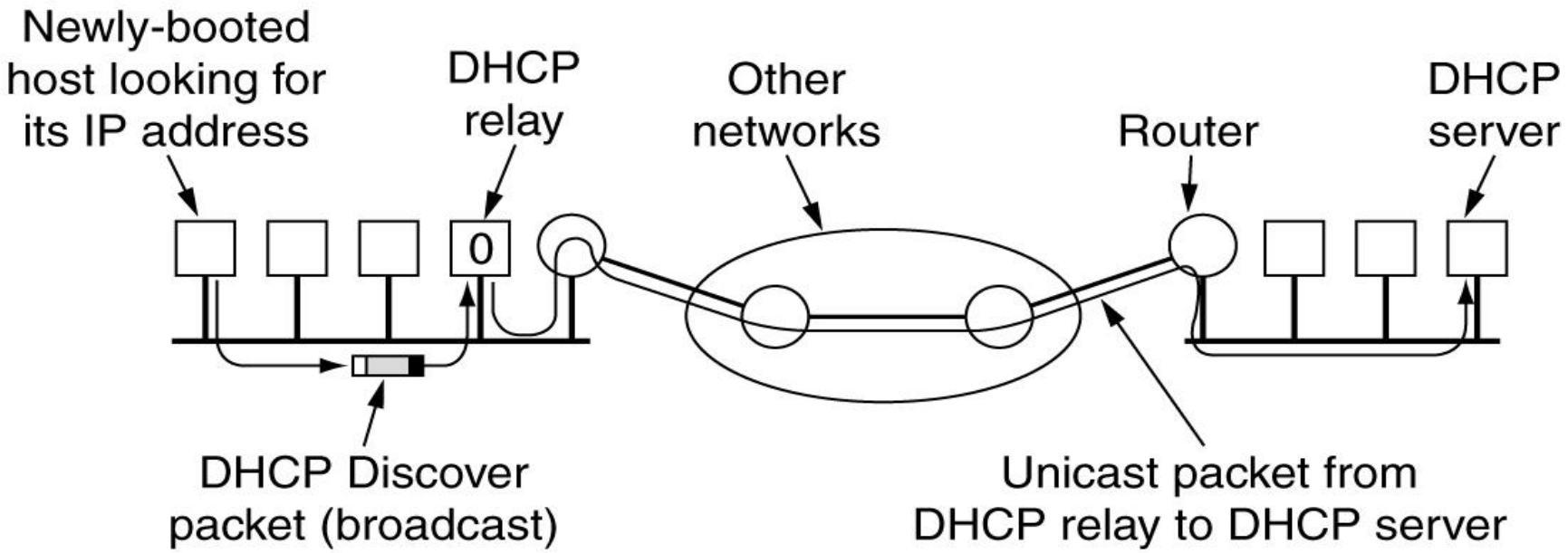


# Dynamic Host Configuration Protocol

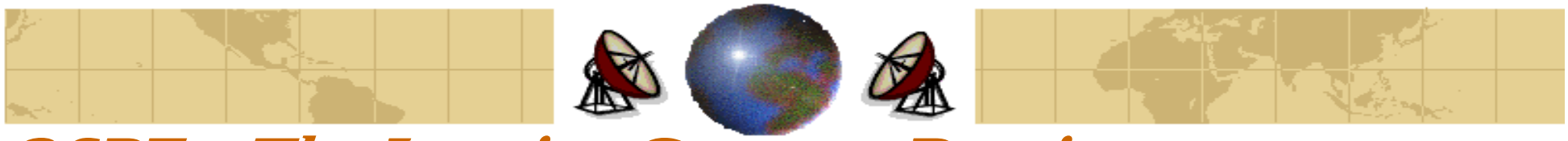
- It is based on a *special server that assigns IP addresses to hosts asking for one*.
- The server need not to be on the same LAN as the requesting host.
- Since the **DHCP server** may not be reachable by *broadcasting*, a **DHCP relay agent** is needed on each LAN.
- To get its IP address, a newly-booted machine broadcasts a **DHCP DISCOVER packet**. The DHCP relay agent on its LAN intercepts all DHCP broadcasts. When it gets a DHCP DISCOVER packet, it sends the packet as a *unicast* packet to the **DHCP server**, possibly on a distant network. The only information the relay agent needs is the *IP address of the DHCP server*.
- IP address assignment may be for a fixed period of time, a technique called **Leasing**. Just before the lease expires, the host must ask the DHCP server for a renewal. If it fails to make a request or the request is denied, the host may no longer use the IP address it was given earlier.



# Dynamic Host Configuration Protocol




Operation of DHCP



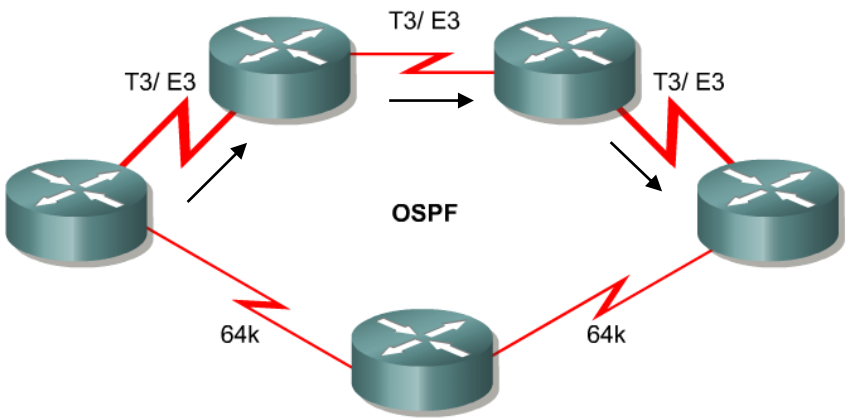
# OSPF – The Interior Gateway Routing Protocol

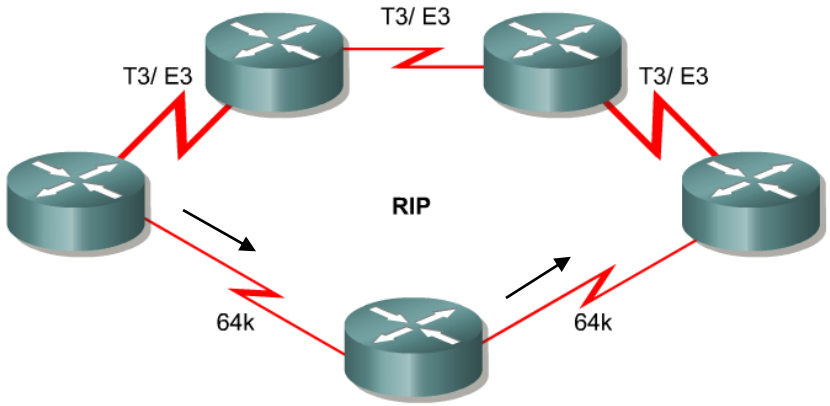
- A routing algorithm within an AS is called an **Interior Gateway Protocol**, and an algorithm for routing between ASes is called an **Exterior Gateway Protocol**.
- The original Internet interior gateway protocol was a distance vector algorithm (**DVA**) based on Bellman-Ford algorithm. However, it was replaced by a link state protocol in May 1979. In 1990, the successor, called **OSPF** (**Open Shortest Path First**) became a standard.

**Characteristics of OSPF:**

1. *Open*.
2. A *variety of metrics* is supported, including physical distance, delay, and so on.
3. *Dynamic*.
4. Routing based on *type of service* (Differentiating between real-time traffic and others).
5. *Load balancing* (Splitting the load over multiple lines).
6. *Hierarchical systems* are supported
7. *Security* is provided (Preventing from spoofing router by sending them false routing information).

# OSPF versus RIP Route Selection



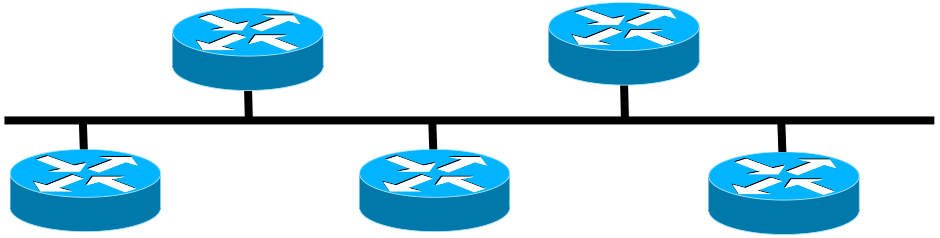



# OSPF – The Interior Gateway Routing Protocol

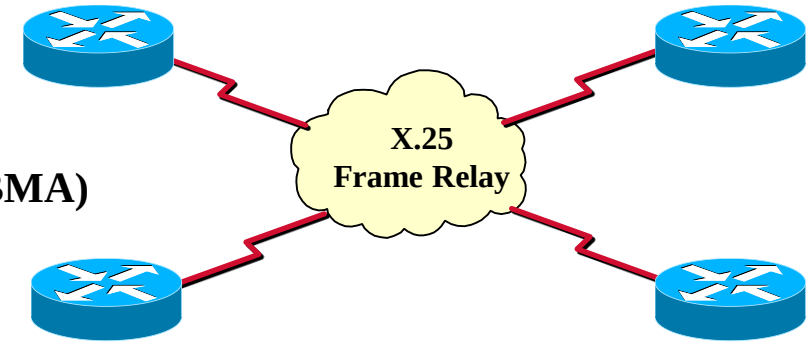
## OSPF Network Types:

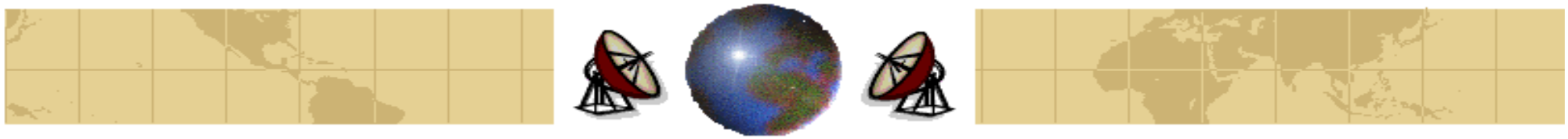
**Point-to-Point**

**Broadcast Multiaccess**



**Nonbroadcast Multiaccess (NBMA)**

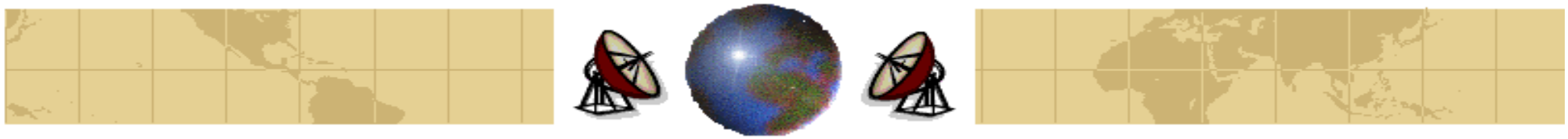




# OSPF – The Interior Gateway Routing Protocol

- OSPF supports **three** types of connections or networks:
  1. *Point-to-Point lines* between exactly two routers.
  2. *Multiaccess networks with broadcasting* (e.g., most LANs).
  3. *Multiaccess networks without broadcasting* (e.g., most packet-switched WAN).

- A **multiaccess** network is one that can *have multiple routers on it, each pair of routers on such a network is able to communicate directly.*

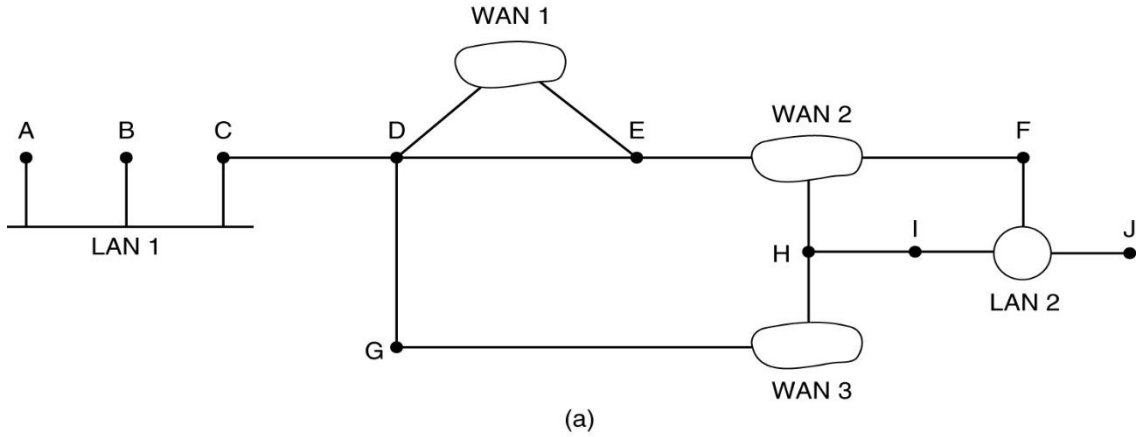


# OSPF – The Interior Gateway Routing Protocol

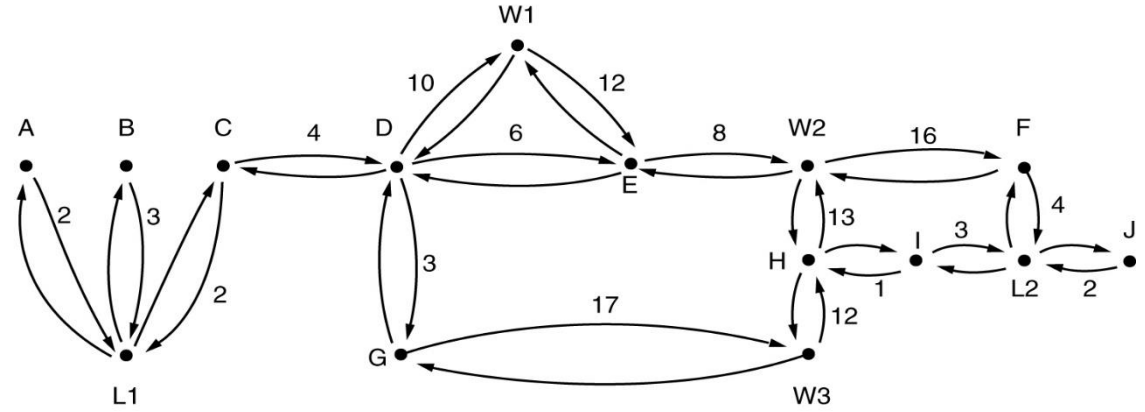
## First: A Simple Autonomous System:

- OSPF works by abstracting a collection of actual networks, routers, and lines into a *directed graph* in which:
  - Each arc is assigned a cost (distance, delay, reliability, etc.). Shortest path is computed based on these weights on the arcs.
  - A connection between two nodes is represented by a pair of arcs, one in each direction.
  - Each network is represented by a node.
  - Each router is also represented by a node.
- OSPF represents actual networks as a directed graph and computes the shortest path from every router to every other router.

# OSPF – The Interior Gateway Routing Protocol

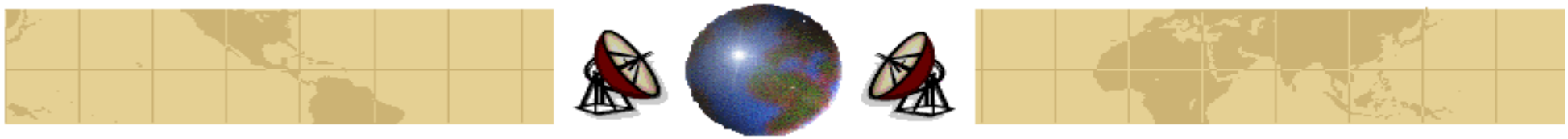



(a)




(b)

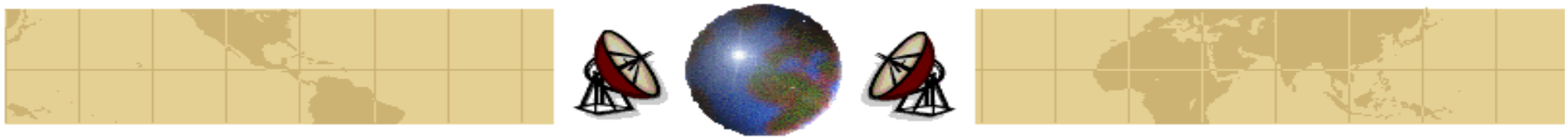
(a) An autonomous system (b) A graph representation of (a)



# OSPF – The Interior Gateway Routing Protocol

## Second: A Large Autonomous System:

- Many of the ASes in the Internet are themselves large and nontrivial to manage.
- OSPF allows them to be divided up into numbered **areas**, where an area is a network or a set of contiguous networks. Areas do not overlap but need not to be exhaustive, that is, some routers may belong to no area.
- Each AS has a **backbone area**, called **area 0**. All areas are connected to the backbone.
- *Each router that is connected to two or more areas is part of the backbone*.
- Within an area, each router has the *same* **link state database** and runs the *shortest path algorithm*. Its main job to calculate the shortest path from itself to every other router in the area, including the router that is connected to the backbone, of which there must be at least one.



# OSPF – The Interior Gateway Routing Protocol

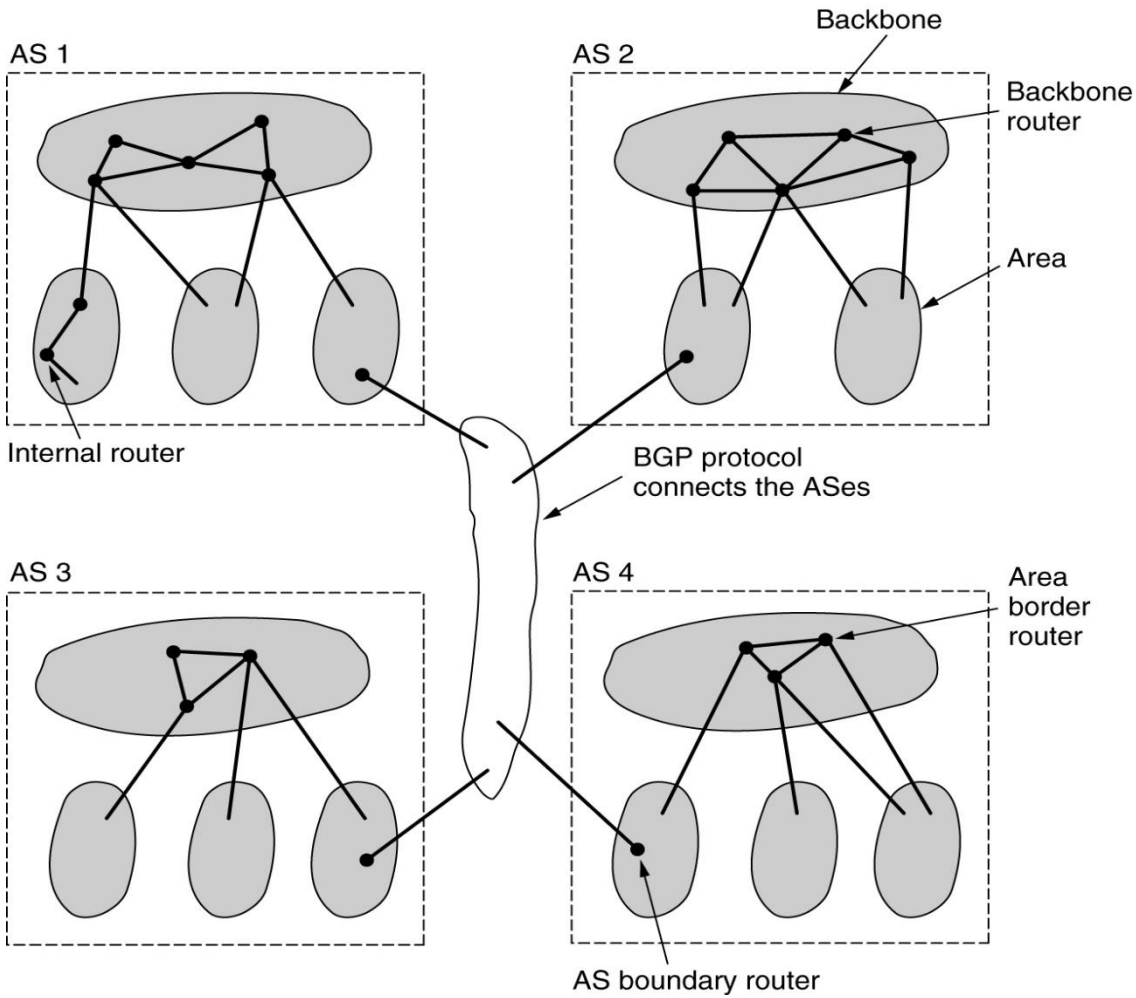
- The way OSPF handles **type of service** routing is to have multiple graphs. One uses **delay** as the metric, one uses **throughput** as the metric, and the other uses the **reliability** as the metric. Although this triples the computation needed, it allows separate routes for optimizing delay, throughput, and reliability.

- Three kinds of routes may be needed:
  1. **Intra-area**
  2. **Interarea**
  3. **InterAS**.

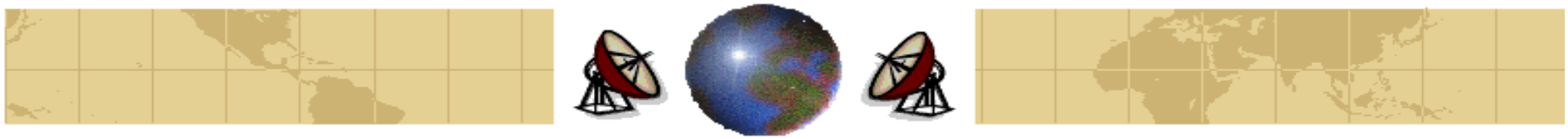
- **Intra-area** routes are the easiest, since the source router knows the shortest path to the destination router.

- **Interarea routing** always proceeds in **three steps**: *go from the source to the backbone; go across the backbone to the destination area; go to the destination*.

# OSPF



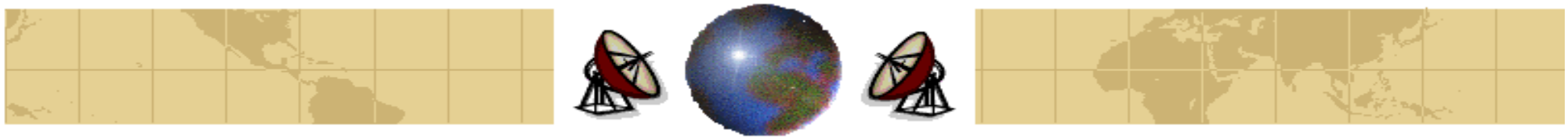

The relation between ASes, backbones, and areas in OSPF.



# OSPF – The Interior Gateway Routing Protocol

- OSPF distinguishes four classes of routers:
  - **Internal routers** are wholly within one area.
  - **Area border routers** connect one or more OSPF areas to the backbone.
  - **Backbone routers** are on the backbone.
  - **AS boundary routers** talk to routers in other ASes.

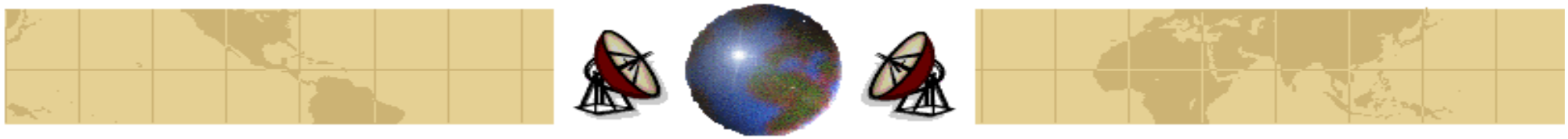
- Note that all the area border routers are part of the *backbone*. In addition, a router that is in the backbone but not part of any other area is also an *internal router*.



# OSPF – The Interior Gateway Routing Protocol

**Procedure:**

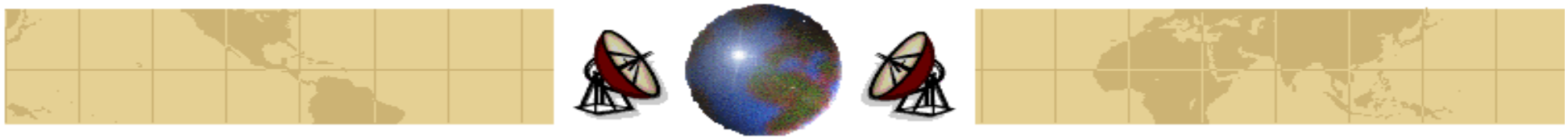
- When a router boots, it sends **HELLO** packets on all of its *point-to-point* lines and multicasts them on LANs. On WANs, it needs some configuration information to know *who to contact*. From the responses, each router learns who its neighbors are.
- The **Hello** carries information about which *all neighbors must agree to form an adjacency and exchange link-state information*.
- On **multi-access networks** the **Designated Router (DR)** and **Backup Designated Router (BDR)** maintain adjacencies with all other OSPF routers on the network.
- OSPF works by exchanging information between **adjacent routers**, which is not the same as between neighboring routers. For example, *it is not efficient to have every router on the LAN talk to every other router on the LAN*. To avoid this situation, on router is elected as the **DR**. It is adjacent to all the other routers on it LAN, and exchanges information with them.
- A **BDR** is always is *kept up* and *up-to-date* to ease the transition if the **DR** crashes.



# OSPF – The Interior Gateway Routing Protocol

**Procedure (*continued*):**

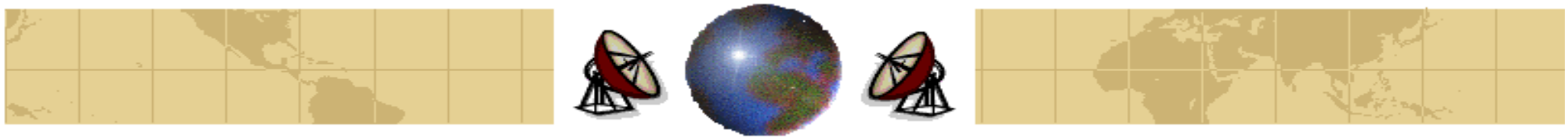
- **Adjacent routers** go through a *sequence of states*. *Adjacent routers must be in the full state before routing tables are created and traffic routed*.
- Each router sends *link-state advertisements (LSA) in link-state update (LSU)* packets. These LSAs *describe all of the routers links*. Each router that receives an LSA from its neighbor records the LSA in the link-state database. This process is repeated for all routers in the OSPF network.
- When a router *detects up or down link or its cost changes*, the router sends **LINK STATE UPDATE (LSU)** packets to each of its adjacent routers. Each **LSU** carries a collection of **link state advertisements (LSAs)** *one hop further from its origin*. Several link state advertisements may be included in a single packet. The **LSU** provides the *costs of the links used in the topological database.*
- The **flooding** packets are acknowledged, to make them reliable. Each **LSU** has a *sequence number*, so a router can see whether an incoming **LSU** is older or newer than what currently has.
- A **LINK STATE ACK (LSAck)** packet is used to acknowledge each **LSU**, to make them reliable.



# OSPF – The Interior Gateway Routing Protocol

**Procedure (*continued*):**

- **Database Description packets** announce the link state entries currently held by the sender. By comparing its own values with those of the sender, the receiver can determine who has the most recent values. *This packet is used when a line is brought up*.
- Either partnet can request link state information from the other one by using **Link State Request (LSR)** packets. The result is that each pair of adjacent routers check to see who has the most recent data, and new information is spread throughout the area.
- Finally, we can put all the pieces together. Using *flooding*, each router informs all the other routers in its area of its neighbors and costs. This information allows each router to construct a *graph* for its area and compute the shortest path (Dijkstra).
- Routing information is now maintained. When there is a change in a link-state, routers use a flooding process to notify other routers on the network about the change. The Hello protocol *dead interval* provides a simple mechanism for determining that an adjacent neighbor is down.

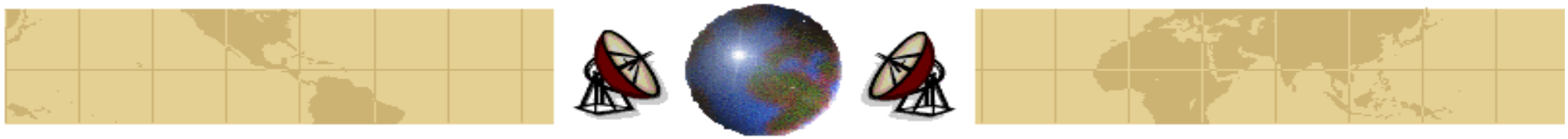


# OSPF – The Interior Gateway Routing Protocol

**Procedure (*continued*):**

- The backbone area does this too. In addition, the backbone routers accept information from the *area border routers* in order to *compute the best route from each backbone router to every other router*.

- This information is propagated back to the *area border routers*, which advertise it within their areas. Using this information, a route about to *send an interarea packet can select the best exit router to the backbone*.

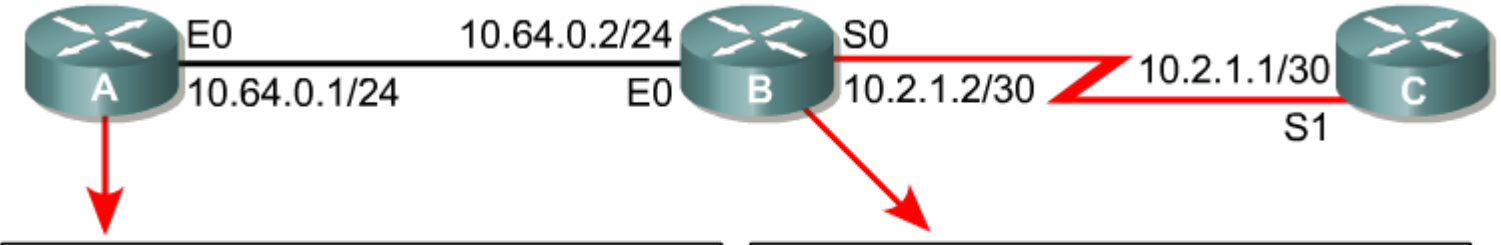


# *OSPF*

| Message type | Description |
|---|---|
| Hello | Used to discover who the neighbors are |
| Link state update | Provides the sender's costs to its neighbors |
| Link state ack | Acknowledges link state update |
| Database description | Announces which updates the sender has |
| Link state request | Requests information from the partner |

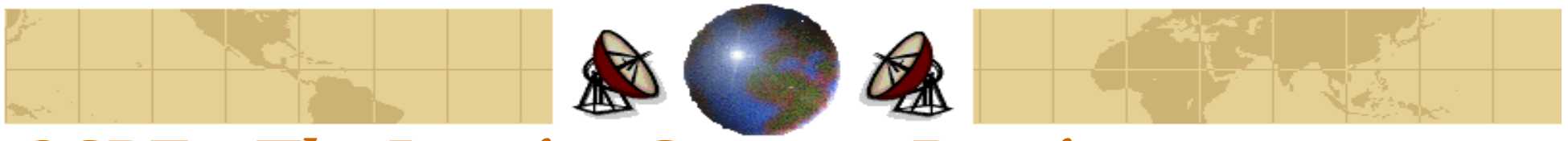
The five types of OSPF messages

# Basic OSPF Configuration




```
<Output Omitted>
interface Ethernet0
ip address 10.64.0.1 255.255.255.0
!
<Output Omitted>
router ospf 1
network 10.64.0.0 0.0.0.255 area 0
```

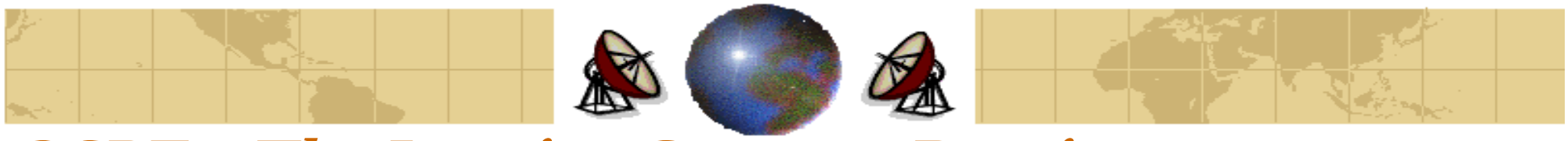
```
<Output Omitted>
interface Ethernet0
ip address 10.64.0.2 255.255.255.0
!
interface Serial0
ip address 10.2.1.2 255.255.255.252
<Output Omitted>
router ospf 1
network 10.2.1.0 0.0.0.3 area 0
network 10.64.0.0 0.0.0.255 area 0
```



# OSPF – The Interior Gateway Routing Protocol

## OSPF Databases:

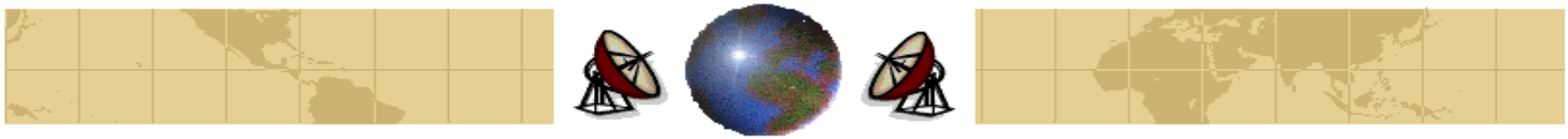
| Database | Description |
|---|---|
| **Adjacencies Database** | List all of the **neighbor routers** to which a router has established bidirectional communication. This is *unique* for each router. |
| **Link-State Database (Topological Database)** | List of information about **all other routers in the network**. This database shows the network topology. All routers within an area have *identical* link state databases. |
| **Forwarding Database (Routing Table)** | List of **routes generated when an algorithm (Dijkstra)** is run on the link-state database. Each router's routing table is *unique* and contains information on how and where to send packets to other routers. |



# OSPF – The Interior Gateway Routing Protocol

## OSPF Packet Types:

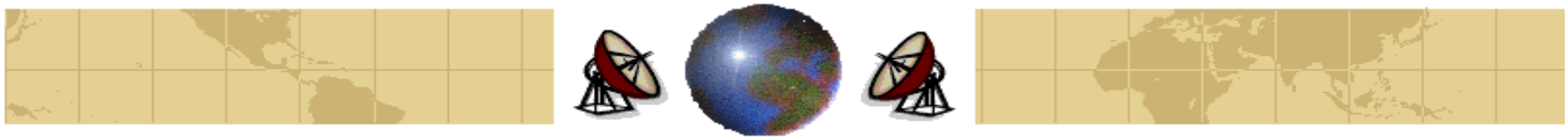
| Type | OSPF Packet Type | Description |
|---|---|---|
| 1 | **Hello** | Establishes and maintains *adjacency information* with neighbors |
| 2 | **Database Description Packet** | Describes the contents of an OSPF router's *link-state database* |
| 3 | **Link State Request** | Requests *specific pieces* of router's *link-state database* |
| 4 | **Link State Update (LSU)** | Transports *link-state* advertisements (LSAs) to neighbors routers |
| 5 | **Link State Acknowledgement (LSACK)** | Acknowledgement receipt of a neighbor's *LSA*. |



# OSPF – The Interior Gateway Routing Protocol

## OSPF States:

- OSPF neighbor relationships progress through the seven states, one at a time in the following order:
  1. **Down**
  2. **Init**
  3. **Two-way**
  4. **ExStart**
  5. **Exchange**
  6. **Loading**
  7. **Full Adjacency**



# OSPF – The Interior Gateway Routing Protocol

## 1. Down State:

- In the Down state, the OSPF process *has not yet exchange information with any neighbor*. OSPF is waiting to enter the Init State.

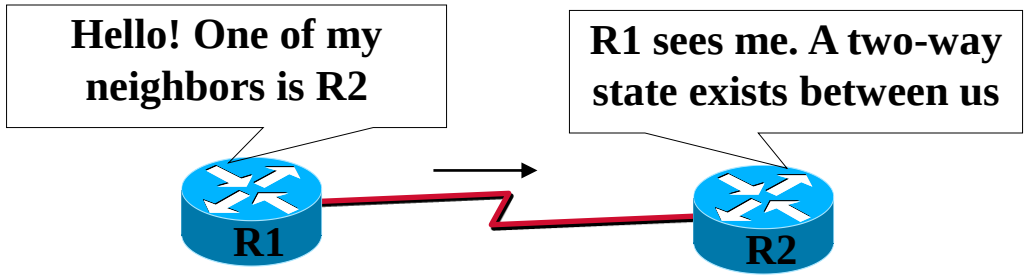
## 2. Init State:

- OSPF routers send Type 1 (Hello) packets at regular interval (usually 10 seconds) to establish relationship with neighbor routers. *When an interface receives its first Hello packet, the router enters the Init state*, which means the router knows a neighbor is out there and is waiting to take the relationship to the next step.

# OSPF – The Interior Gateway Routing Protocol

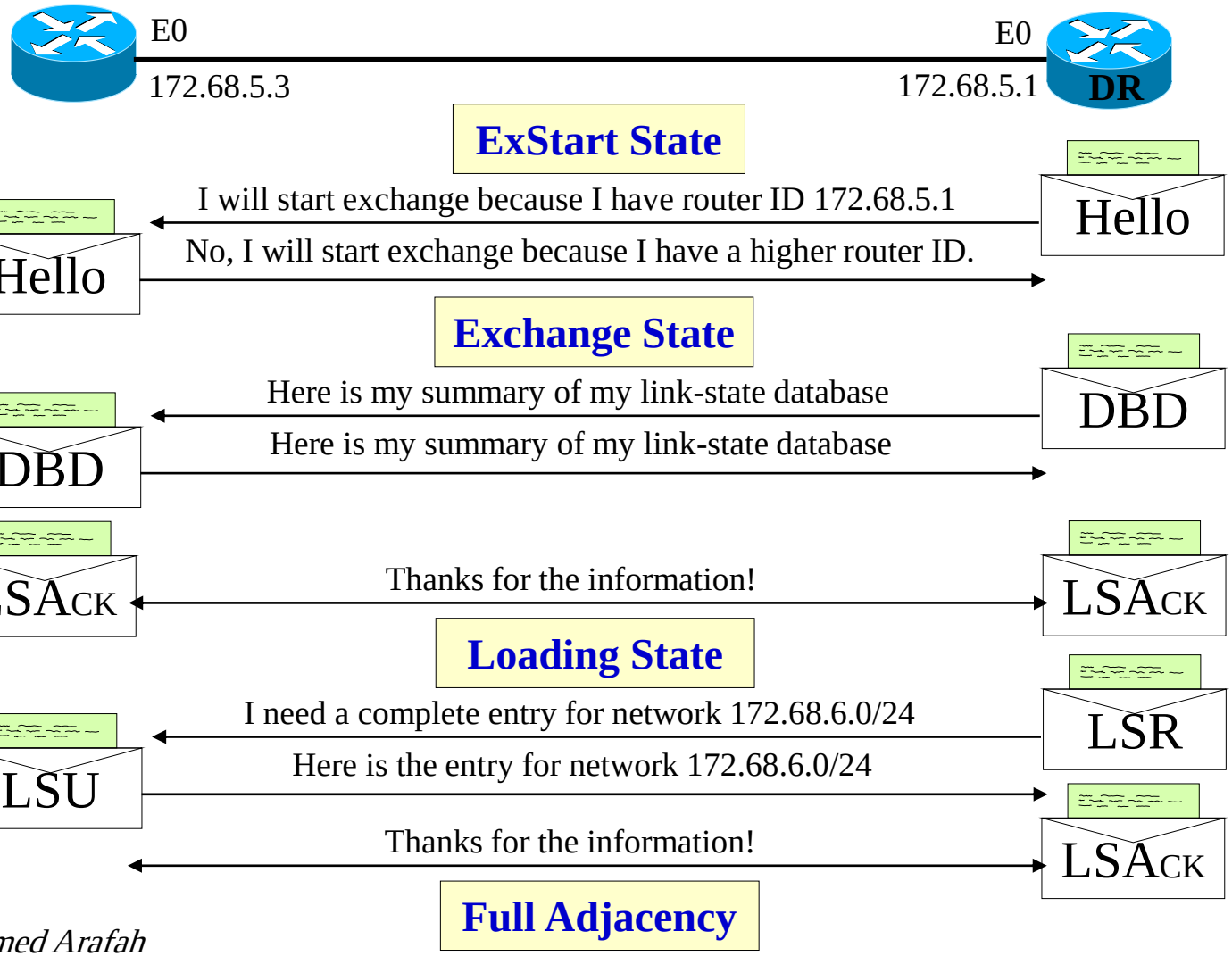
### 3. Two-Way State:

- Hello packets include a list of the sender's known OSPF neighbors. A router enters the two-way state when it sees itself in a neighbor's Hello.




- The routing information is not shared between routers in two-way state. To learn about other routers' link state, every OSPF router must form at least one *adjacency*. The first step toward full adjacency is the **ExStart State**.

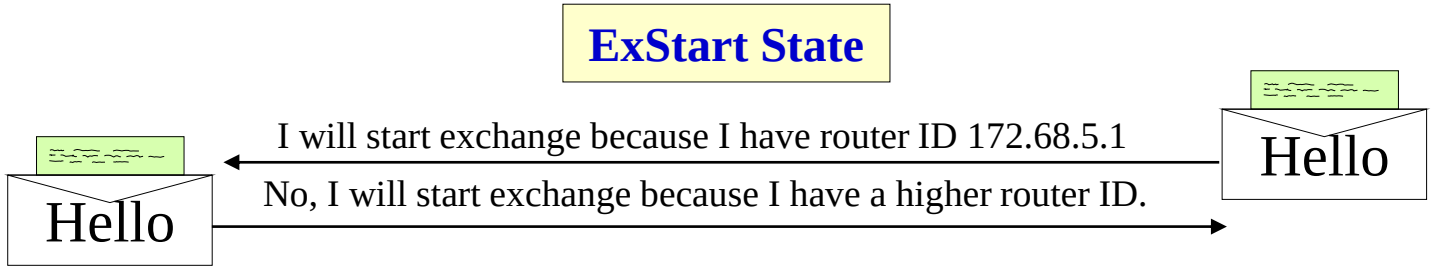
# OSPF – The Interior Gateway Routing Protocol



# OSPF – The Interior Gateway Routing Protocol

## 4. ExStart State:

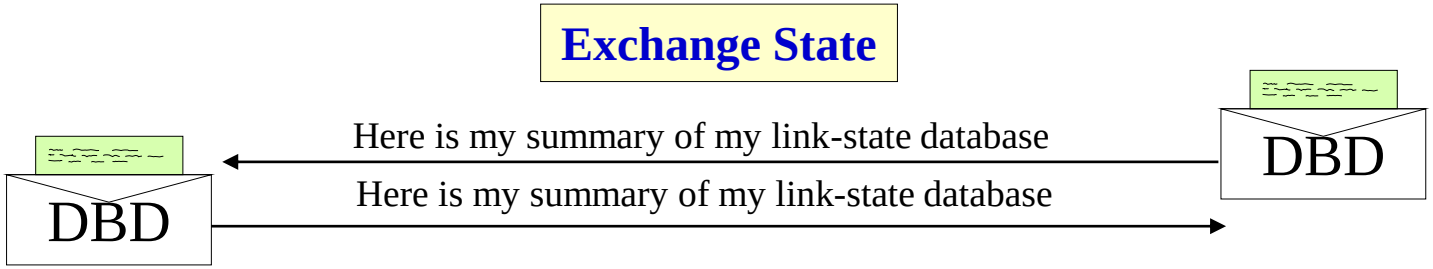
- The two neighbor routers use Hello packets to negotiate who is the *master* and who is the *slave* in their relationship.
- The router with the highest OSPF router ID becomes the master.
- When the neighbors establish their roles as master and slave, they enter the *Exchange state* and begin sending routing information.



# OSPF – The Interior Gateway Routing Protocol

## 5. Exchange State:

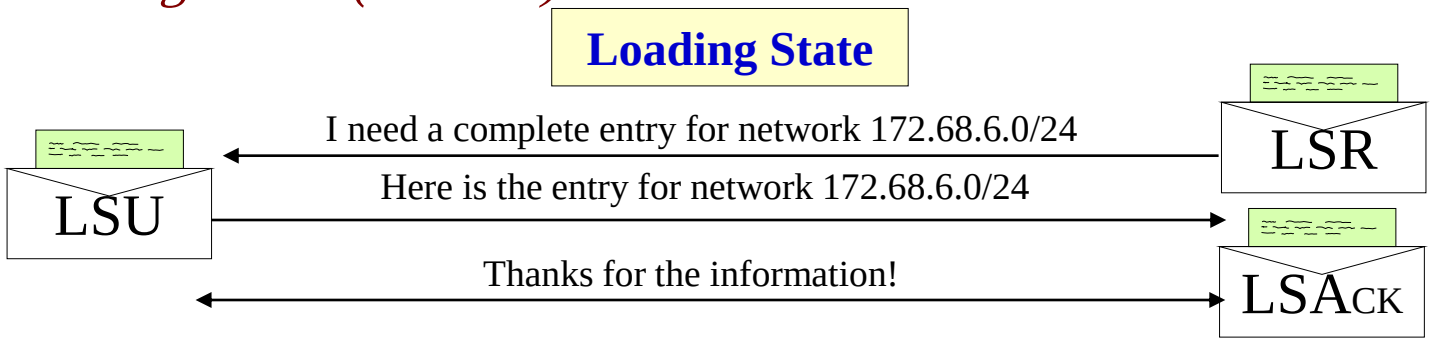
- Neighbor routers use Type 2 *database description (DBD) packets* to send each other their *link-state information*.
- The routers compare what they learn with their existing link-state databases.
- If either router of the router receives information about a link that is not already in its database, the router requests a complete update from its neighbor.

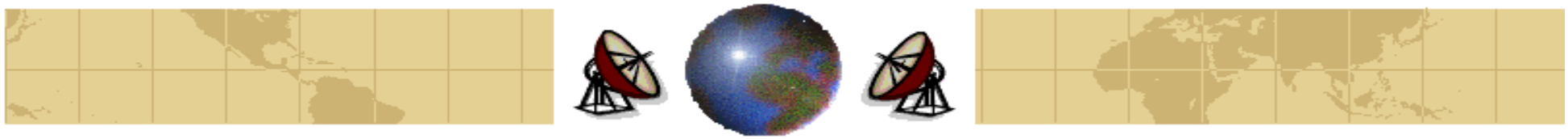


# OSPF – The Interior Gateway Routing Protocol

## 6. Loading State:

- After each router describes its database to the others, the router can request information that is more complete by using Type 3 packet, *Link State Requests (LSRs).*
- When a router receives an LSR, it responds with an update using Type 4 *Link State Update (LSU) packet*. These Type 4 LSU contain the actual *link-state advertisements (LSAs),* which is the heart of the link-state routing protocols.
- Type 4 LSUs are acknowledged using Type 5 packets, called *Link-State Acknowledgements (LSAcks).*

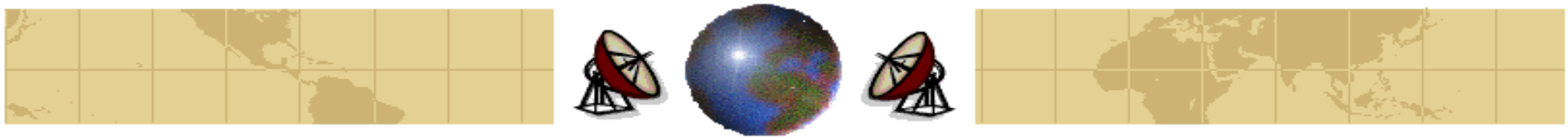




# OSPF – The Interior Gateway Routing Protocol

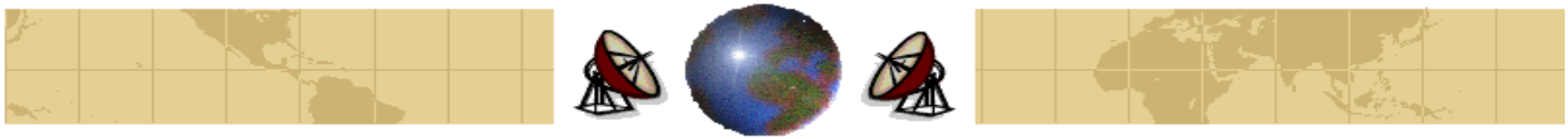
## 7. Full Adjacency:

- With Loading state complete, the routers are fully adjacent.
- Each router keeps a list of adjacent neighbors, called the *adjacency database*.



# BGP – The Exterior Gateway Routing Protocol

- Between ASes, a different routing protocol, **BGP (Border Gateway Protocol)**, is used.
- Exterior gateway protocol routers have to worry about politics a great deal.
- Policies are manually configured into each BGP router. They are not a part of the protocol itself.
- Two BGP routers are considered connected if they share a common network.
- BGP is fundamentally a distance vector protocol. Instead of periodically giving each neighbor its estimated cost to each possible destination, each BGP routers tells its neighbors the exact path it is using.
- After all the paths come in from the neighbors, the BGP router examines them to see which is the best.

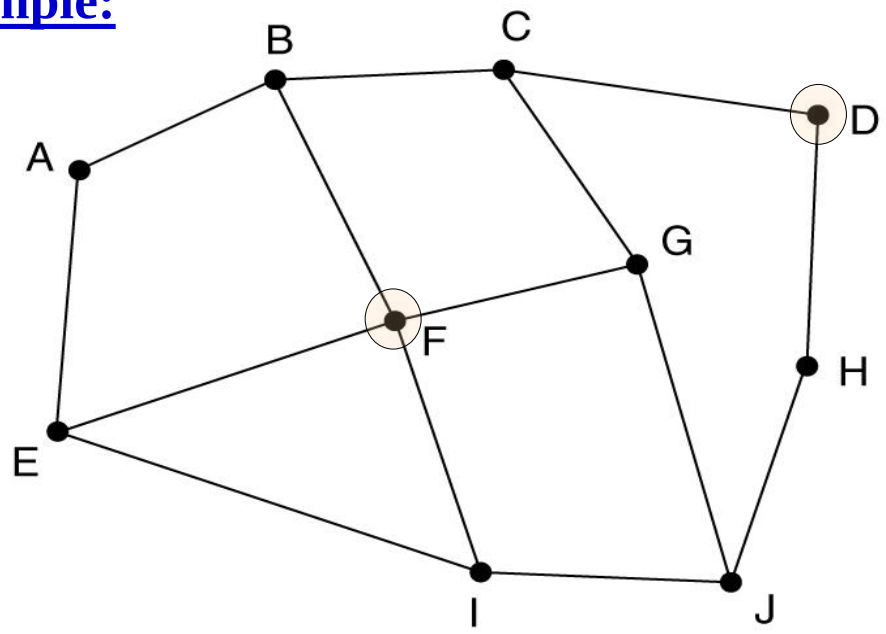


# BGP – The Exterior Gateway Routing Protocol

- Between ASes, a different routing protocol, **BGP (Border Gateway Protocol)**, is used.
- Exterior gateway protocol routers have to worry about politics a great deal.
- *Policies are manually configured* into each BGP router. They are not a part of the protocol itself.
- Two BGP routers are considered connected if they *share a common network*.
- BGP is fundamentally a distance vector protocol. *Instead of periodically giving each neighbor its estimated cost to each possible destination, each BGP routers tells its neighbors the exact path it is using*.
- After all the paths come in from the neighbors, the BGP router examines them to see which is the best.

# BGP – The Exterior Gateway Routing Protocol

**Example:**




(a)

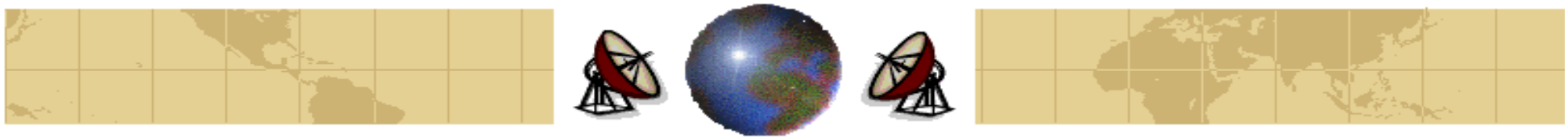
Information F receives
from its neighbors about D

From B: "I use BCD"
From G: "I use GCD"
From I: "I use IFGCD"
From E: "I use EFGCD"

(b)

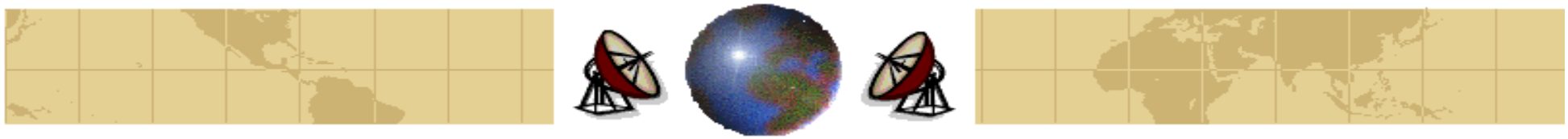
(a) A set of BGP routers

(b) Information sent to F



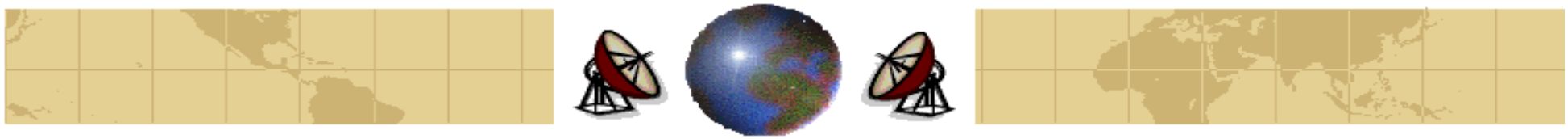
# *Internet Multicasting*

- IP supports multicasting, using **class D addresses**.
- *Twenty-eight bits are used for identifying groups*, so over 250 million groups can exist at the same time.
- Two kinds of group addresses are supported. A **permanent group** is always there and does not have to be set up, and a **temporary group** that must be created before they can be used.
- Multicasting is implemented by special multicast routers, about once a minute, each multicast router sends a hardware multicast to the hosts on its LAN (address 224.0.0.1) asking them to report back on the groups their processes currently belong to. Each host sends back responses for all the class D addresses it is interested in.
- These query and responses packets use a protocol called **IGMP (Internet Group Management Protocol)**.



# IPv6

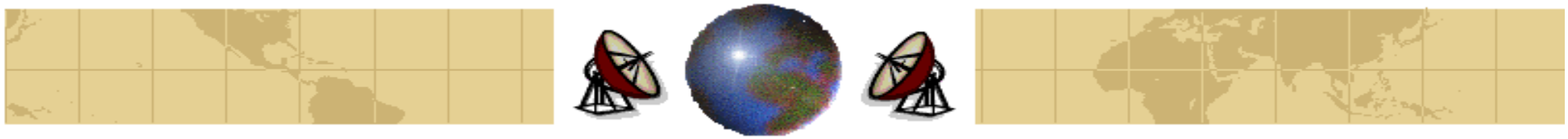
- While CIDR may buy a few more years, everyone realizes that the days of IP in its current form (IPv4) are numbered.
- With the explosion of interest in the Internet starting in mid 1990s, it will be used by much larger group of people, especially people with different requirements. Also, it may not be long before every television set in the world is an Internet node, producing a billion machine being used for video on demand.
- Seeing these problems, the work was started on a new version of IP, one which would never run out of addresses, would solve a variety of other problems, and be more flexible and efficient as well.



# IPv6

## Major goals of IPv6:

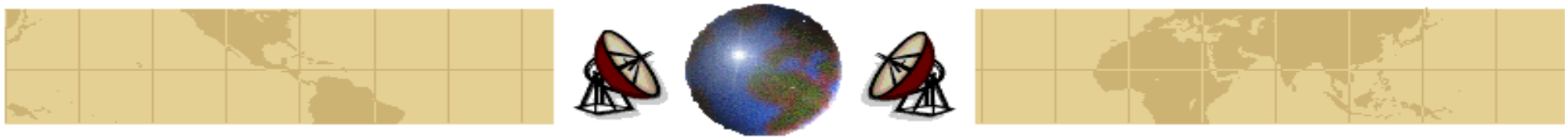
- Support billions of hosts.
- Reduce the size of routing tables.
- Simply the protocol, to allow routers to process the packet faster.
- Provide better security.
- Pay more attention to type of service, particularly for real-time data.
- Aid multicasting by allowing scopes to be specified (TV as Internet node).
- Make possible to host to change place without changing its address
- Allow the protocol to evolve in the future.
- Permit the old and new protocols to coexist for years.



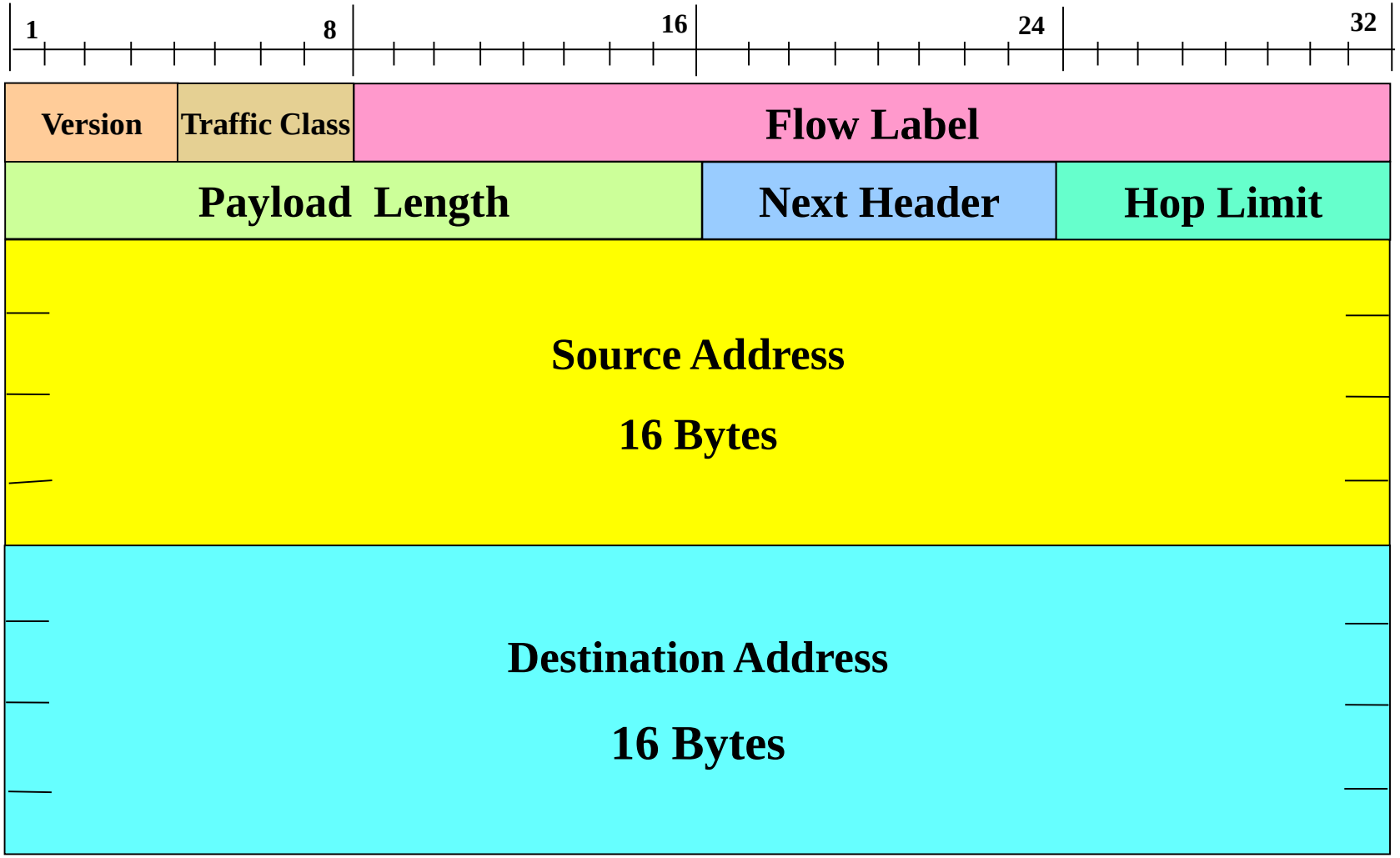
# IPv6

## Main Features of IPv6:

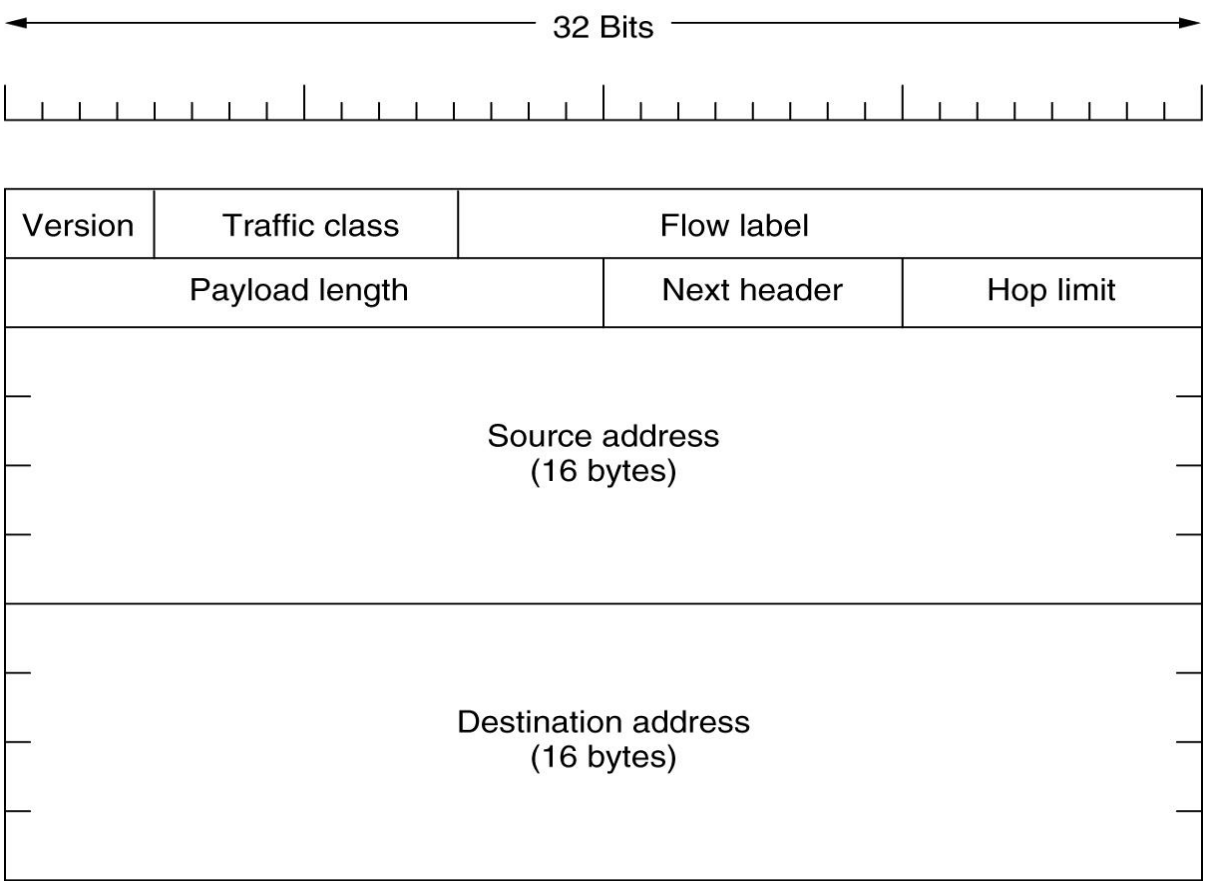
- *IPv6 has longer addresses than IPv4*. They are 16 bytes long, which provides effectively unlimited supply of Internet addresses.
- *IPv6 simplifies the header*. It contains only 7 fields (versus 13 in IPv4). This change allows the router to process packets faster.
- *IPv6 provides better support for options* (fields that were required in IPv4 are becoming optional).
- *Authentication and privacy* are key features in IPv6.
- More attention has been paid to *type of service* than in the past.



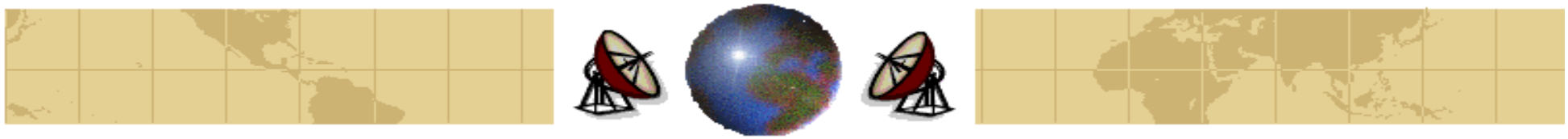
# The Main IPv6 Header



# The Main IPv6 Header




The IPv6 fixed header (required)



# The main IPv6 Header

## Version Field (4 bits):

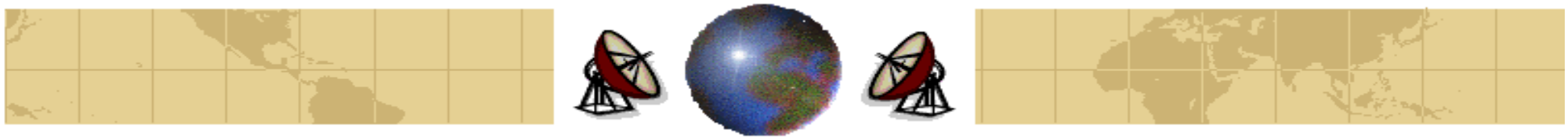
- The version field is always 6.

## Traffic Class Field (4 bits):

- Traffic Class field is used to distinguish among packets.
- Values 0 through 7 are for transmissions that are *capable of slowing down* in the event of congestion.
- *Values 8 through 15 are for real-time traffic*.
- The IPv6 standard suggests to use 1 for news, 4 for FTP, and 6 for Telnet connections.

## Flow Label Field (24 bits):

- The flow label field is used will be allow a source and destination to set up a connection with particular properties and requirements.



# The main IPv6 Header

## The Payload Length Field (16 bits):

- The payload field tells *how many bytes follow the 40-byte header*.

## The Next Header Field (8 bits):

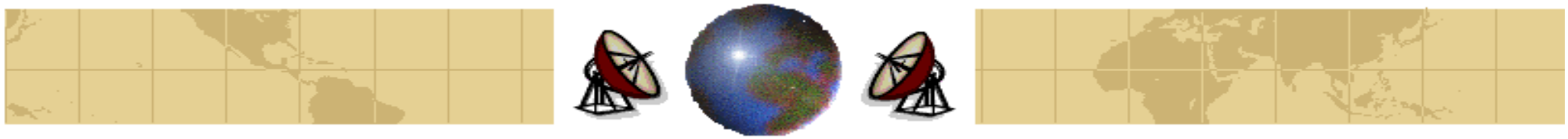
- The reason the header could be simplified is that there can be additional (optional) headers.
- *If this header is the last IP header, the next header field tells which transport protocol handler (e.g., TCP, UDP) to pass the packet to*.

## The Hop Limit Field (8 bits):

- The hop limit field is used to keep packets from living forever.

## The Source Address Field (16 bytes).

## The Destination Address Field (16 bytes).



# The main IPv6 Header

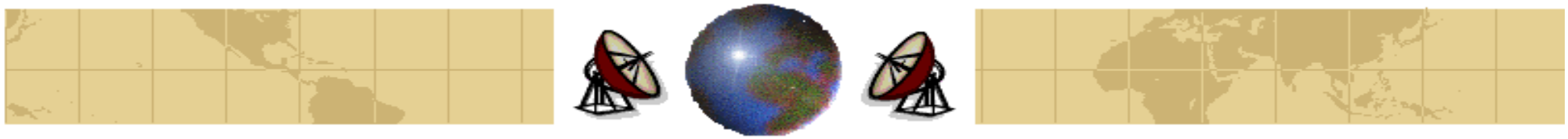
- The 16-byte addresses are written as eight groups of four hexadecimal digits with colon between the groups, like as:

  **8000:0000:0000:0000:0123: 4567:89AB:CDEF**

- Since many addresses will have many zeros inside them, three optimizations have been authorized:
  - Leading zeros within a group can be omitted.
  - One or more groups of 16 zeros can be replaced by a pair of colons. Thus the above address now becomes

    **8000::123: 4567:89AB:CDEF**
  - IPv4 addresses can be written as a pair of colons and an old dotted decimal number, for example:

    **:: 192.31.20.46**



# Extension Headers

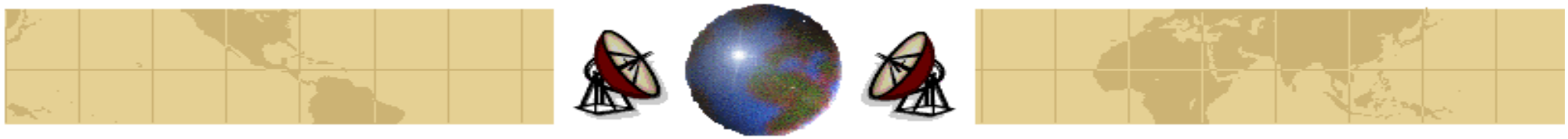
| Extension header | Description |
|---|---|
| Hop-by-hop options | Miscellaneous information for routers |
| Destination options | Additional information for the destination |
| Routing | Loose list of routers to visit |
| Fragmentation | Management of datagram fragments |
| Authentication | Verification of the sender's identity |
| Encrypted security payload | Information about the encrypted contents |

IPv6 extension headers

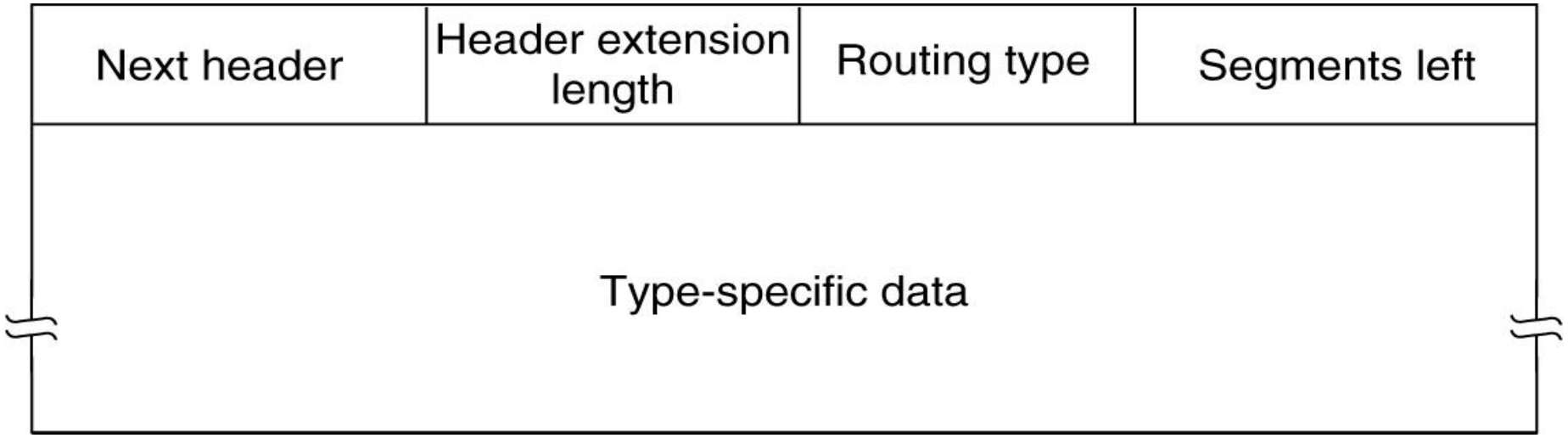
# *Extension Headers*

| Next header | 0 | 194 | 4 |
|---|---|---|---|
| Jumbo payload length | | | |

The hop-by-hop extension header for large datagrams (jumbograms)



# Extension Headers




The extension header for routing