Ipv4 datagram format

32 bits

Version (4)	Header length (4)	Type of Service (8)	Datagram length (bytes) (16)
16-bit identifier (16)		flags (3)	Frag offset (13)
TTL (8)	Upper-layer Protocol (8)	Header checksum (16)	
32-bit Source IP (32)			
32-bit destination IP (32)			
Options (if any)			
Data			

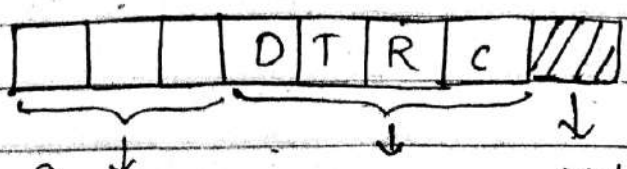
Version Number - 4 bits (4/6)

Header length → 4 bits (Header length = value × 4)
- → usually 20 bytes (without options)
- → value (5-16)

Type of Service → previously called service type now called differentiated service

Service type

D: Minimize delay
T: Maximize throughput
R: " Reliability
C: Minimize cost



Packets in IP layer are called datagram

Header (20-60 bytes) | Data

→ Precedence defines the priority of datagram in issues such as congestion.
→ In case a router is congested lower priority packets are dropped
→ Range 000 - 111

→ TOS (4 bit) but one and only one of the bits can have the value 1 in each datagram

Types of service

TOS bits	Description
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	" throughput
1000	Minimize delay

Protcol	TOS bits	Description
ICMP	0000	Normal
BOOTP	0000	"
NNTP	0001	Minimize cost
IGP	0010	Maximize reliability
SNMP	0010	" "
TELNET	1000	Minize delay
FTP(data)	0100	Maximize throughput
FTP(control)	1000	Minimize delay

TFTP	1000	Minimize delay
SMTP (command)	1000	" "
SMTP (data)	0100	Maximize throughput
DNS (UDP query)	1000	Minize delay
DNS (TCP query)	0000	Normal
DNS (zone)	0100	Maximize throughput

Those protocol that send data in bulk
↓
Maximum throughput

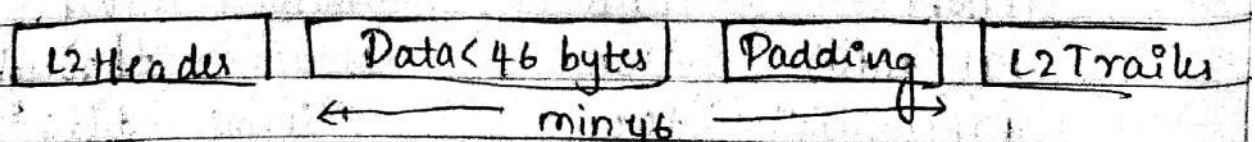
Those protocol that require immediate reply
↓
Minimize Delay

Total length (16 bits) (0-65535)
↳ data + heads
Data = Total - heads

Because some physical networks can't encapsulate large datagrams fragmentation is done.

Ethernet protocol (46 - 1500 bytes size of data)

Encapsulation of small datagram in Ethernet frame



After decapsulation they need to know how much is data & padding.

synchronous clocks

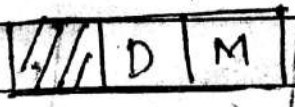
TTL - 8 bits
↳ timestamp / hops
↓
twice the max no. of routers b/w any 2 hosts
↳ It limits lifetime of datagram

Protocol → Datagram can have data from higher protocols

Value	Protocol
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

→ It defines the final destination protocol to which the datagram is delivered.
→ Helps receiving network layer know to which protocol the data belongs
→ Checksum
→ Source
→ Dest

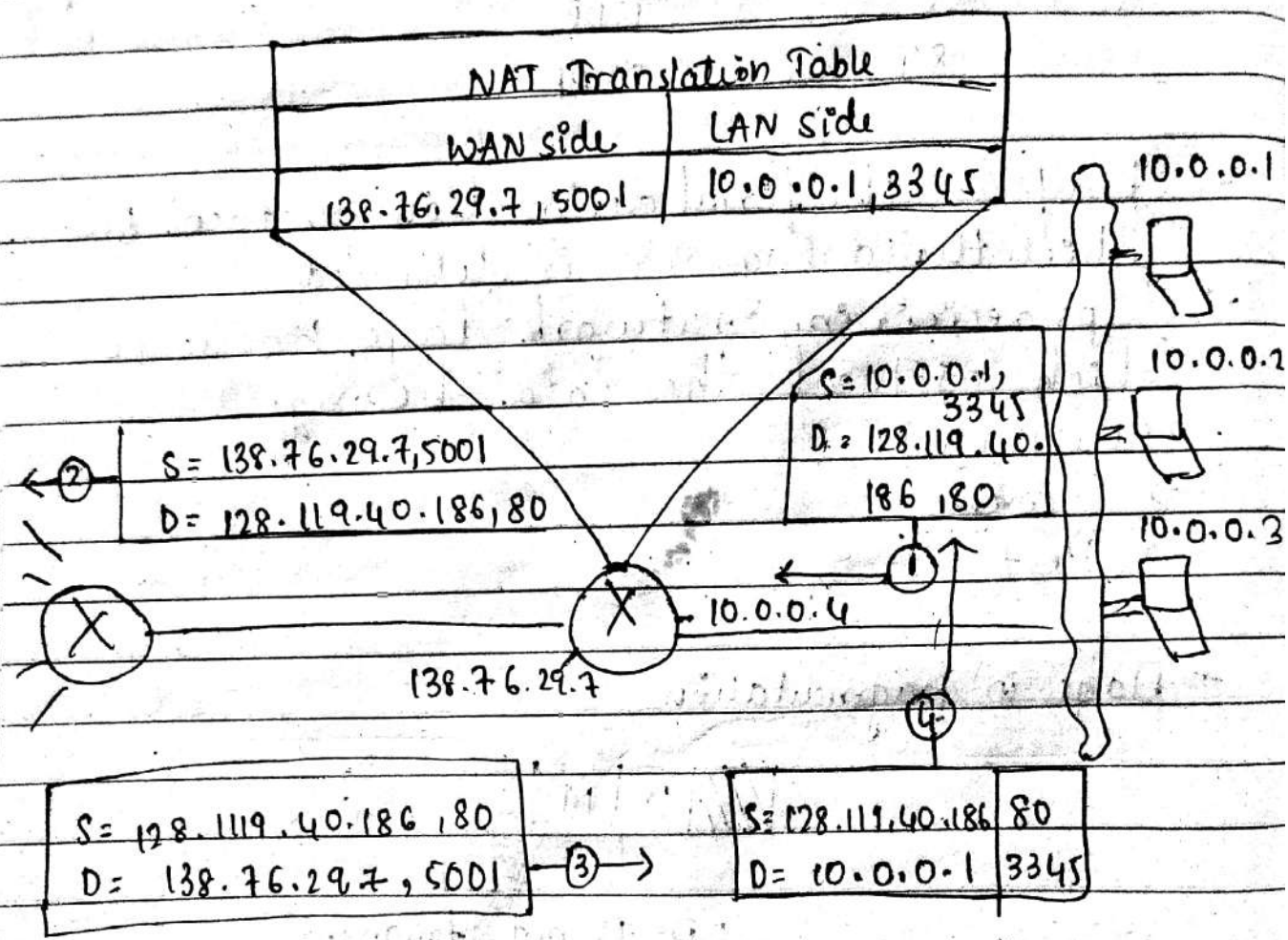
* Flags in fragmentation



D: Do not fragment
M: more fragments.

Protocol	MTU
Hyperchannel	65535
Token Ring (16 Mbps)	17,914
" " (4 Mbps)	4,464
FDDI	4,352
Ethernet	1500
X.25	576
PPP	296

NAT



→ In step 2, port no is translated
→ After step 2, table entry is made

NAT

→ NAT enabled router does not look like router to outside world it's just a single device with IP address

→ All packets leaving from LAN to large Internet has source IP of WAN side of router

→ " " entering LAN / home router must have destination IP as WAN side of router

→ The NAT enabled router has LAN and WAN sides

→ Private network is transparent to entire Internet, but rest of Internet only sees NAT router

→ The NAT router gets its IP address from ISP's-DHCP server

→ The Router runs a DHCP servers to provide addresses to computers within NAT - DHCP - router - controlled private network's address space

Why?

* Address Conservation
* Security :- NAT Routes hides details of home/private network from Internet

Benefits

*Enables to use large set of address internally and one or smaller set of addresses externally (conservation)

→ If NAT wasn't there the packets from LAN can't be forwarded beyond it's network.

NAT Translation Table

→ NAT router maintains it to know to which internal host the datagram must be forwarded
→ It includes port no's & IP addresses

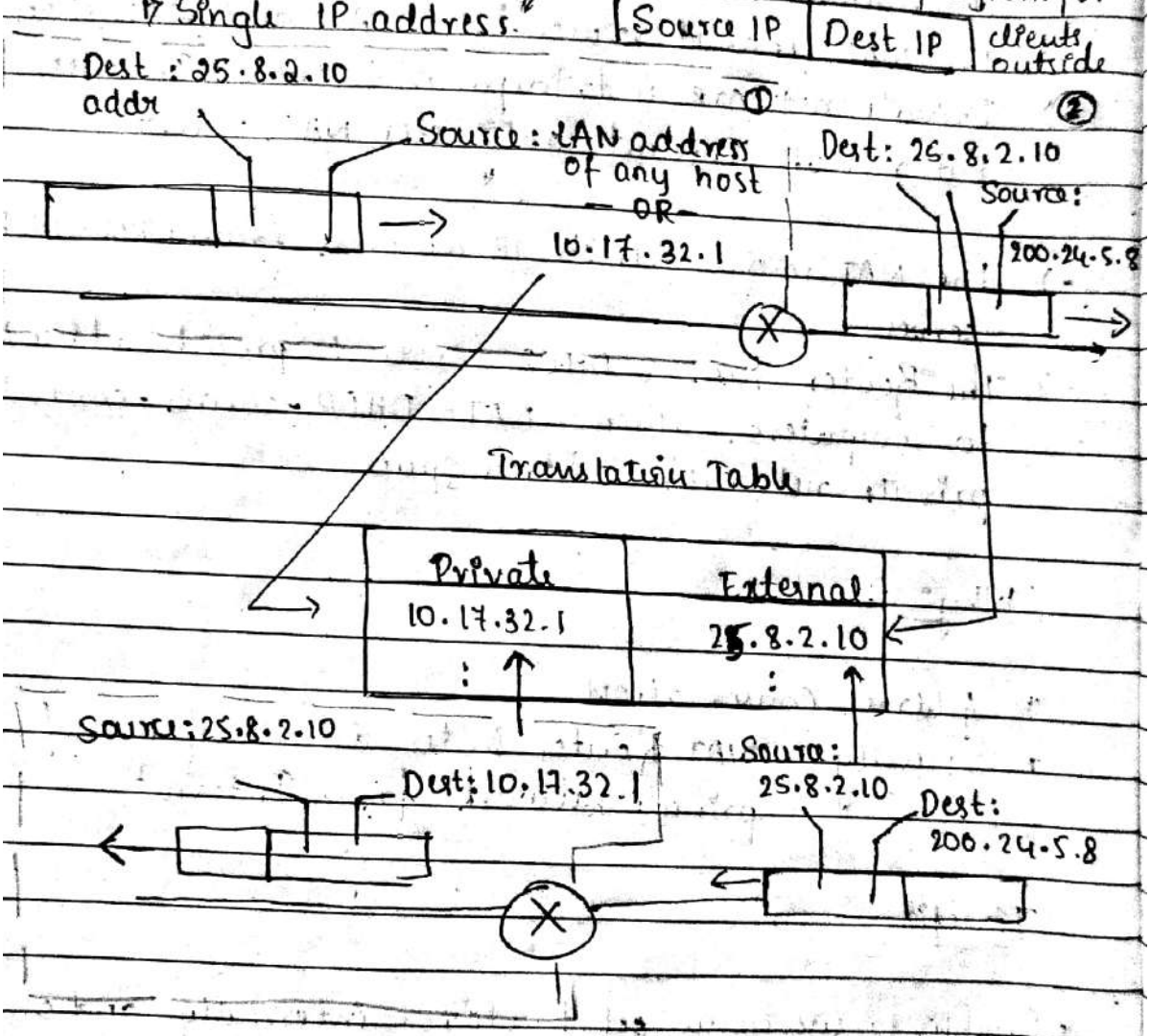
↓

Evolution

↗ Private network starts comm'n

↗ " " can't run server program for clients outside

↗ Single IP address



2) Using pool of IP addresses: (NAT has pool of global addresses) (So no. of global addresses = no of private network hosts that can communicate with some external host at same time) (But no more than no. of global IP's) (No private network host can access two external servers at same time)

3) Using both IP addresses & port no's

Security

* Firewalls : Install b/w private network & Internet
  → It inspects the incoming datagram, segment header fields and denies suspicious datagrams to internal network
  → Ex: Discarding ICMP echo request calls that does traditional port scan across private IP address range
  → Might block on basis of source & dest IP & port no's
  → Can allow only approved TCP connections

* Intrusion Detection System (IDS)
  → Installed at network boundary
  → Does "deep packet Inspection" (both headers & payloads)
  → It maintains a signature table / db of packet signatures that are known as part of attacks
  → DB gets updated with new attacks discovered.
  → Every datagram header & payload is matched with DB - if match found alert is created

* Intrusion Prevention Systems (IPS's)
  → Same as IDS but it discards packet in addition to creating alert

* IPv4 fragmentation

→ The maximum amount of data that a link-layer frame can carry is called MTU
→ Each IP datagram is encapsulated within link-layer frame for transportation from one router to another.

→ Fragment payload in the IP datagram into 2/more IP datagrams
→ Encapsulate these fragmented datagrams in separate link-layer frame
→ Send these frames to outgoing link
→ Reassembly of fragments happens in end system not in network cores
→ Identification number - tells source
→ flag - tells last fragment recieved or not
→ offset - tells where the fragment fits in orginal datagram.

IPv4 Addressing

→ Host is typically connected to singe link in network
→ " sends datagram on this link
→ Boundary b/w host & physical link - interface

→ Router has multiple links
→ " " " interfaces (for each of its link)
→ IP address technically associated with an interface rather than host / router containing the interface (∵ every host / router is capable of sending/recieving IP datagrams) IP requires each host & router interface to have its own address)

→ Network is interconnected by Ethernet LAN
→ Interfaces " " " " Switch or by wireless access point
→ IP terms, the network connecting these hosts interfaces & router interface forms subnet

→ The network portion of IP address is called prefix
→ The ability of using a single prefix to advertise multiple networks → address aggregation / route aggregation or route summarization

Ex:

prefix.../mask ← ISP → 8 organization → own subnets.

ICANN → allocates IP addresses
↘ manage DNS root servers.
↘ assigns domain name & resolve domain name disputes.

DHCP → address of local DNS server
↘ provides IP
↘ subnet mask
↘ first-hop router address // default gateway

plug & play

zeroconf (zero configuration)

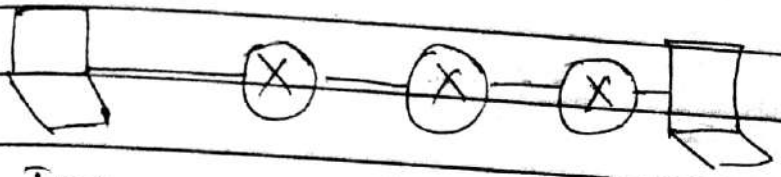
If DHCP server is not present in subnet then DHCP relay agent (typically router) that knows the address of DHCP server for that network is used.

Unit-3

Ipv4 v/s Ipv6

→ No of customers are more than available IP's in Ipv4
→ So we have moved from Ipv4 to Ipv6
→ Ipv4 is 32-bit whereas Ipv6 is 128 bits
→ Ipv4 address space is 2^{32} and for Ipv6 is 2^{128}

How communication is successful inspite of both Ipv4 and Ipv6 being used?



Ipv4 ——————→ Ipv6

Methods for Ipv4 to Ipv6 translation.

1) Dual Stack
→ Both src and dest maintains stack that has both IPv4 & IPv6 addresses
· AAAA- IPV6

2) Tunneling
↳ Separate tunnel b/w src & dest.
If tunnel ingress ends Ipv4 or Ipv6 then casting happen
Embeds Ipv6 into Ipv4

3) Header Translation / NAT
Header value for version is translated
Dest ingress Ipv6 & src is Ipv4. then Ipv4 to Ipv6 in header

SNMP Protocol

→ Version 1 & 2
→ Port 160 Request
→ Port 162 - Respond
→ Manages — Agent
↓
Device that controls entire setup. A system in which SNMP manage is loaded. It sends request. — Replies (Switch, Router)

NMS - Network Management System.
Whatsapp Gold
→ Used by network admins

Ipv4 - 32 bit address
- 2^{32} address space (4 billion +)
→ It defines the connection of a device to the Internet
→ They are unique
→ Binary and dotted decimal notations

→ First address of block make 32-n as zeros
→ last " " " make 32-n as ones
→ Block Size / No. of addresses in a block = 2^{32-n}

→ First address of a block AND MASK and address
→ Last address " " " OR address & complement of mask
→ No. of addresses / Block size = Compliment mask write it decimal and add 1

NAT → Address conservation
→ Security
↓ → enables user to have large set of addresses internally and one or small set of addresses externally
There will single connection to Internet through router that runs NAT software

Specials IP's local IP's

	Range	Total
10.0.0.0	to 10.255.255.255	2^{24}
172.16.0.0	to 172.31.255.255	2^{20}
192.168.0.0	to 192.168.255.255	2^{16}

No routes will forward a packet with these destination addresses.

Private network is transparent to rest of Internet; the rest of the Internet sees only the NAT router

20.3 IPv6

The network layer protocol in the TCP/IP protocol suite is currently IPv4 (Internetworking Protocol, version 4). IPv4 provides the host-to-host communication between systems in the Internet. Although IPv4 is well designed, data communication has evolved since the inception of IPv4 in the 1970s. IPv4 has some deficiencies (listed below) that make it unsuitable for the fast-growing Internet.

- Despite all short-term solutions, such as subnetting, classless addressing, and NAT, address depletion is still a long-term problem in the Internet.
- The Internet must accommodate real-time audio and video transmission. This type of transmission requires minimum delay strategies and reservation of resources not provided in the IPv4 design.
- The Internet must accommodate encryption and authentication of data for some applications. No encryption or authentication is provided by IPv4.

To overcome these deficiencies, **IPv6 (Internetworking Protocol, version 6)**, also known as **IPng (Internetworking Protocol, next generation)**, was proposed and is now a standard. In IPv6, the Internet protocol was extensively modified to accommodate the unforeseen growth of the Internet. The format and the length of the IP address were changed along with the packet format. Related protocols, such as ICMP, were also modified. Other protocols in the network layer, such as ARP, RARP, and IGMP, were

either deleted or included in the ICMPv6 protocol (see Chapter 21). Routing protocols, such as RIP and OSPF (see Chapter 22), were also slightly modified to accommodate these changes. Communications experts predict that IPv6 and its related protocols will soon replace the current IP version. In this section first we discuss IPv6. Then we explore the strategies used for the transition from version 4 to version 6.

The adoption of IPv6 has been slow. The reason is that the original motivation for its development, depletion of IPv4 addresses, has been remedied by short-term strategies such as classless addressing and NAT. However, the fast-spreading use of the Internet, and new services such as mobile IP, IP telephony, and IP-capable mobile telephony, may eventually require the total replacement of IPv4 with IPv6.

Advantages

The next-generation IP, or IPv6, has some advantages over IPv4 that can be summarized as follows:

- **Larger address space.** An IPv6 address is 128 bits long, as we discussed in Chapter 19. Compared with the 32-bit address of IPv4, this is a huge (2^{96}) increase in the address space.
- **Better header format.** IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the upper-layer data. This simplifies and speeds up the routing process because most of the options do not need to be checked by routers.
- **New options.** IPv6 has new options to allow for additional functionalities.
- **Allowance for extension.** IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.
- **Support for resource allocation.** In IPv6, the type-of-service field has been removed, but a mechanism (called *flow label*) has been added to enable the source to request special handling of the packet. This mechanism can be used to support traffic such as real-time audio and video.
- **Support for more security.** The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.

Packet Format

The IPv6 packet is shown in Figure 20.15. Each packet is composed of a mandatory base header followed by the payload. The payload consists of two parts: optional extension headers and data from an upper layer. The base header occupies 40 bytes, whereas the extension headers and data from the upper layer contain up to 65,535 bytes of information.

Base Header

Figure 20.16 shows the **base header** with its eight fields.

These fields are as follows:

- **Version.** This 4-bit field defines the version number of the IP. For IPv6, the value is 6.
- **Priority.** The 4-bit priority field defines the priority of the packet with respect to traffic congestion. We will discuss this field later.

Figure 20.15 *IPv6 datagram header and payload*

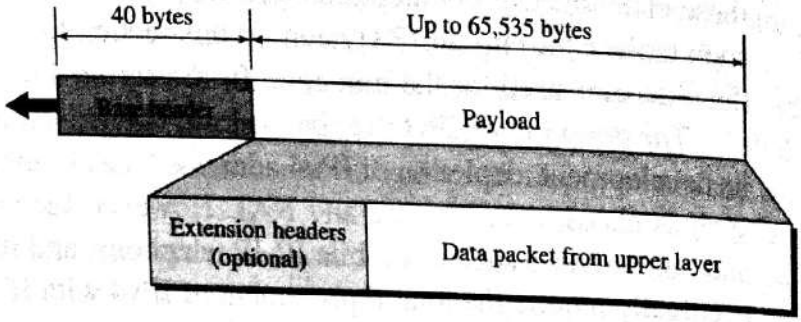
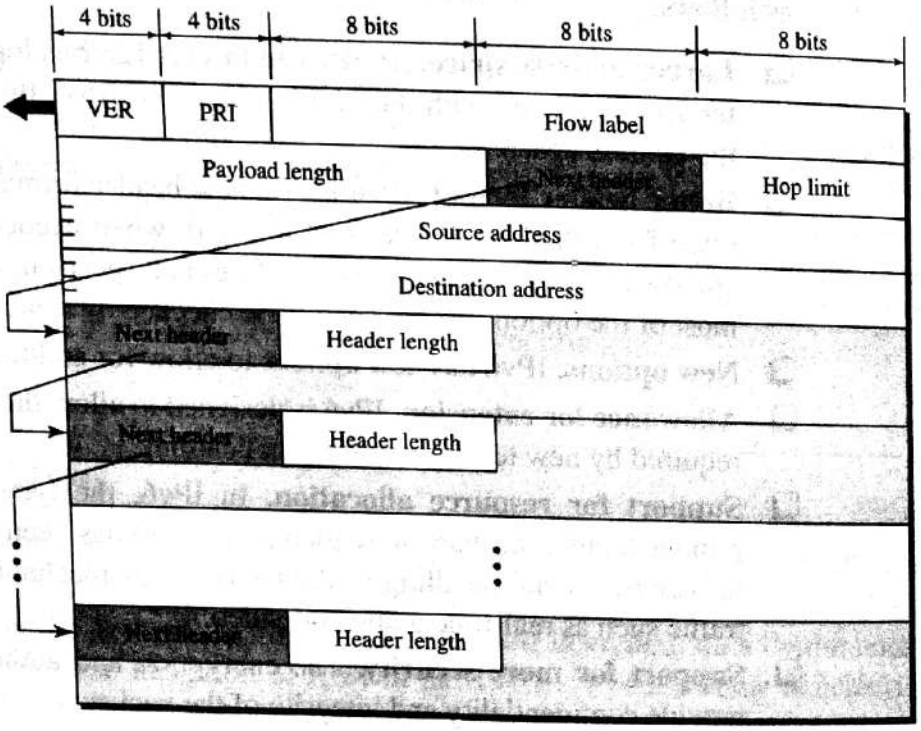


Figure 20.16 *Format of an IPv6 datagram*



- **Flow label.** The **flow label** is a 3-byte (24-bit) field that is designed to provide special handling for a particular flow of data. We will discuss this field later.
- **Payload length.** The 2-byte payload length field defines the length of the IP datagram excluding the base header.
- **Next header.** The **next header** is an 8-bit field defining the header that follows the base header in the datagram. The next header is either one of the optional extension headers used by IP or the header of an encapsulated packet such as UDP or TCP. Each extension header also contains this field. Table 20.6 shows the values of next headers. Note that this field in version 4 is called the *protocol*.
- **Hop limit.** This 8-bit **hop limit** field serves the same purpose as the TTL field in IPv4.
- **Source address.** The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram.

Table 20.6 *Next header codes for IPv6*

Code	*Next Header*
0	Hop-by-hop option
2	ICMP
6	TCP
17	UDP
43	Source routing
44	Fragmentation
50	Encrypted security payload
51	Authentication
59	Null (no next header)
60	Destination option

- **Destination address.** The destination address field is a 16-byte (128-bit) Internet address that usually identifies the final destination of the datagram. However, if source routing is used, this field contains the address of the next router.

Priority

The priority field of the IPv6 packet defines the priority of each packet with respect to other packets from the same source. For example, if one of two consecutive datagrams must be discarded due to congestion, the datagram with the lower **packet priority** will be discarded. IPv6 divides traffic into two broad categories: congestion-controlled and noncongestion-controlled.

Congestion-Controlled Traffic If a source adapts itself to traffic slowdown when there is congestion, the traffic is referred to as **congestion-controlled traffic.** For example, TCP, which uses the sliding window protocol, can easily respond to traffic. In congestion-controlled traffic, it is understood that packets may arrive delayed, lost, or out of order. Congestion-controlled data are assigned priorities from 0 to 7, as listed in Table 20.7. A priority of 0 is the lowest; a priority of 7 is the highest.

Table 20.7 *Priorities for congestion-controlled traffic*

Priority	*Meaning*
0	No specific traffic
1	Background data
2	Unattended data traffic
3	Reserved
4	Attended bulk data traffic
5	Reserved
6	Interactive traffic
7	Control traffic

The priority descriptions are as follows:

- **No specific traffic.** A priority of 0 is assigned to a packet when the process does not define a priority.
- **Background data.** This group (priority 1) defines data that are usually delivered in the background. Delivery of the news is a good example.
- **Unattended data traffic.** If the user is not waiting (attending) for the data to be received, the packet will be given a priority of 2. E-mail belongs to this group. The recipient of an e-mail does not know when a message has arrived. In addition, an e-mail is usually stored before it is forwarded. A little bit of delay is of little consequence.
- **Attended bulk data traffic.** A protocol that transfers data while the user is waiting (attending) to receive the data (possibly with delay) is given a priority of 4. FTP and HTTP belong to this group.
- **Interactive traffic.** Protocols such as TELNET that need user interaction are assigned the second-highest priority (6) in this group.
- **Control traffic.** Control traffic is given the highest priority (7). Routing protocols such as OSPF and RIP and management protocols such as SNMP have this priority.

Noncongestion-Controlled Traffic This refers to a type of traffic that expects minimum delay. Discarding of packets is not desirable. Retransmission in most cases is impossible. In other words, the source does not adapt itself to congestion. Real-time audio and video are examples of this type of traffic.

Priority numbers from 8 to 15 are assigned to **noncongestion-controlled traffic.** Although there are not yet any particular standard assignments for this type of data, the priorities are usually based on how much the quality of received data is affected by the discarding of packets. Data containing less redundancy (such as low-fidelity audio or video) can be given a higher priority (15). Data containing more redundancy (such as high-fidelity audio or video) are given a lower priority (8). See Table 20.8.

Table 20.8 *Priorities for noncongestion-controlled traffic*

Priority	*Meaning*
8	Data with greatest redundancy
...	...
15	Data with least redundancy

Flow Label

A sequence of packets, sent from a particular source to a particular destination, that needs special handling by routers is called a *flow* of packets. The combination of the source address and the value of the *flow label* uniquely defines a flow of packets.

To a router, a flow is a sequence of packets that share the same characteristics, such as traveling the same path, using the same resources, having the same kind of security, and so on. A router that supports the handling of flow labels has a flow label table. The table has an entry for each active flow label; each entry defines the services required by

the corresponding flow label. When the router receives a packet, it consults its flow label table to find the corresponding entry for the flow label value defined in the packet. It then provides the packet with the services mentioned in the entry. However, note that the flow label itself does not provide the information for the entries of the flow label table; the information is provided by other means such as the hop-by-hop options or other protocols.

In its simplest form, a flow label can be used to speed up the processing of a packet by a router. When a router receives a packet, instead of consulting the routing table and going through a routing algorithm to define the address of the next hop, it can easily look in a flow label table for the next hop.

In its more sophisticated form, a flow label can be used to support the transmission of real-time audio and video. Real-time audio or video, particularly in digital form, requires resources such as high bandwidth, large buffers, long processing time, and so on. A process can make a reservation for these resources beforehand to guarantee that real-time data will not be delayed due to a lack of resources. The use of real-time data and the reservation of these resources require other protocols such as Real-Time Protocol (RTP) and Resource Reservation Protocol (RSVP) in addition to IPv6.

To allow the effective use of flow labels, three rules have been defined:

1. The flow label is assigned to a packet by the source host. The label is a random number between 1 and $2^{24} - 1$. A source must not reuse a flow label for a new flow while the existing flow is still active.
2. If a host does not support the flow label, it sets this field to zero. If a router does not support the flow label, it simply ignores it.
3. All packets belonging to the same flow have the same source, same destination, same priority, and same options.

Comparison Between IPv4 and IPv6 Headers

Table 20.9 compares IPv4 and IPv6 headers.

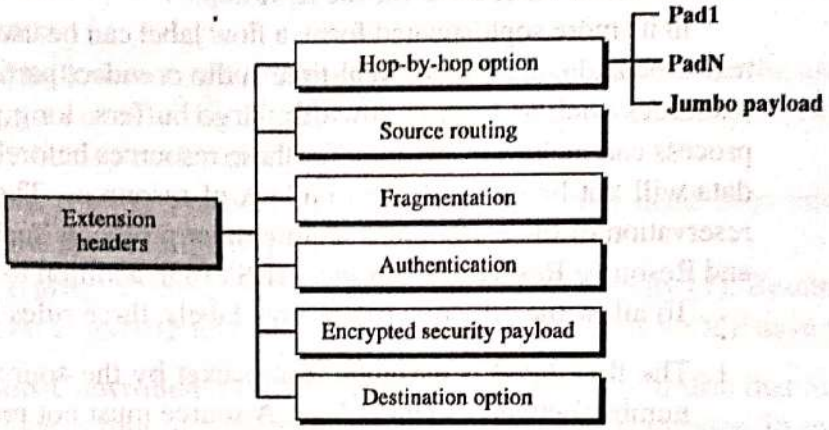
Table 20.9 *Comparison between IPv4 and IPv6 packet headers*

Comparison
1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
5. The TTL field is called hop limit in IPv6.
6. The protocol field is replaced by the next header field.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

Extension Headers

The length of the base header is fixed at 40 bytes. However, to give greater functionality to the IP datagram, the base header can be followed by up to six **extension headers.** Many of these headers are options in IPv4. Six types of extension headers have been defined, as shown in Figure 20.17.

Figure 20.17 *Extension header types*



Hop-by-Hop Option

The **hop-by-hop option** is used when the source needs to pass information to all routers visited by the datagram. So far, only three options have been defined: **Pad1, PadN,** and **jumbo payload.** The Pad1 option is 1 byte long and is designed for alignment purposes. PadN is similar in concept to Pad1. The difference is that PadN is used when 2 or more bytes is needed for alignment. The jumbo payload option is used to define a payload longer than 65,535 bytes.

Source Routing The source routing extension header combines the concepts of the strict source route and the loose source route options of IPv4.

Fragmentation

The concept of **fragmentation** is the same as that in IPv4. However, the place where fragmentation occurs differs. In IPv4, the source or a router is required to fragment if the size of the datagram is larger than the MTU of the network over which the datagram travels. In IPv6, only the original source can fragment. A source must use a **path MTU discovery technique** to find the smallest MTU supported by any network on the path. The source then fragments using this knowledge.

Authentication

The **authentication** extension header has a dual purpose: it validates the message sender and ensures the integrity of data. We discuss this extension header when we discuss network security in Chapter 31.

Encrypted Security Payload

The **encrypted security payload (ESP)** is an extension that provides confidentiality and guards against eavesdropping. We discuss this extension header in Chapter 31.

Destination Option The **destination option** is used when the source needs to pass information to the destination only. Intermediate routers are not permitted access to this information.

Comparison Between IPv4 Options and IPv6 Extension Headers

Table 20.10 compares the options in IPv4 with the extension headers in IPv6.

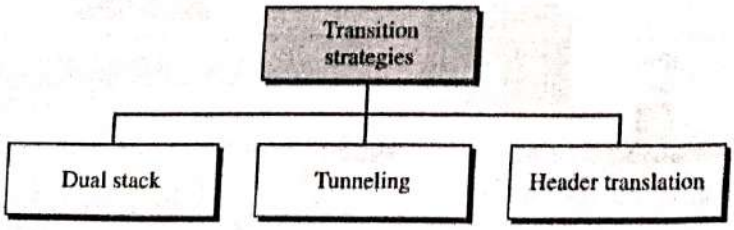
Table 20.10 *Comparison between IPv4 options and IPv6 extension headers*

Comparison
1. The no-operation and end-of-option options in IPv4 are replaced by Pad1 and PadN options in IPv6.
2. The record route option is not implemented in IPv6 because it was not used.
3. The timestamp option is not implemented because it was not used.
4. The source route option is called the source route extension header in IPv6.
5. The fragmentation fields in the base header section of IPv4 have moved to the fragmentation extension header in IPv6.
6. The authentication extension header is new in IPv6.
7. The encrypted security payload extension header is new in IPv6.

20.4 TRANSITION FROM IPv4 TO IPv6

Because of the huge number of systems on the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. It takes a considerable amount of time before every system in the Internet can move from IPv4 to IPv6. The transition must be smooth to prevent any problems between IPv4 and IPv6 systems. Three strategies have been devised by the IETF to help the transition (see Figure 20.18).

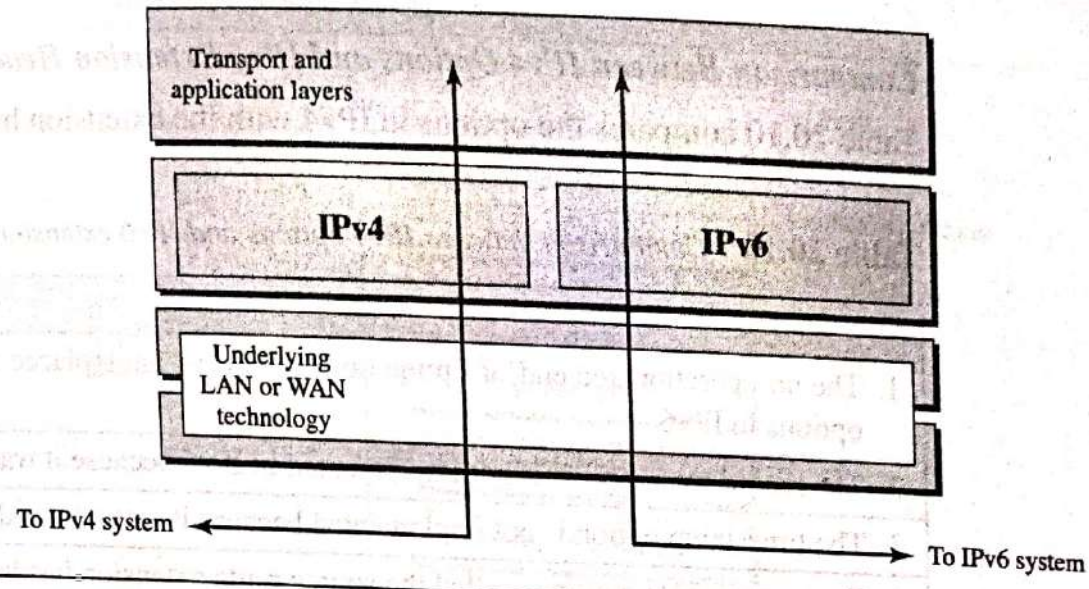
Figure 20.18 *Three transition strategies*



Dual Stack

It is recommended that all hosts, before migrating completely to version 6, have a **dual stack** of protocols. In other words, a station must run IPv4 and IPv6 simultaneously until all the Internet uses IPv6. See Figure 20.19 for the layout of a dual-stack configuration.

Figure 20.19 *Dual stack*

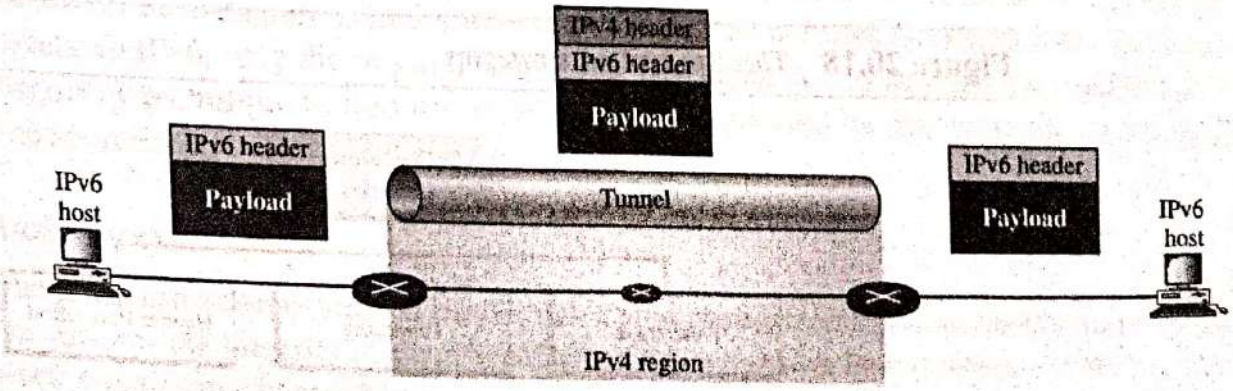


To determine which version to use when sending a packet to a destination, the source host queries the DNS. If the DNS returns an IPv4 address, the source host sends an IPv4 packet. If the DNS returns an IPv6 address, the source host sends an IPv6 packet.

Tunneling

Tunneling is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4. To pass through this region, the packet must have an IPv4 address. So the IPv6 packet is encapsulated in an IPv4 packet when it enters the region, and it leaves its capsule when it exits the region. It seems as if the IPv6 packet goes through a tunnel at one end and emerges at the other end. To make it clear that the IPv4 packet is carrying an IPv6 packet as data, the protocol value is set to 41. Tunneling is shown in Figure 20.20.

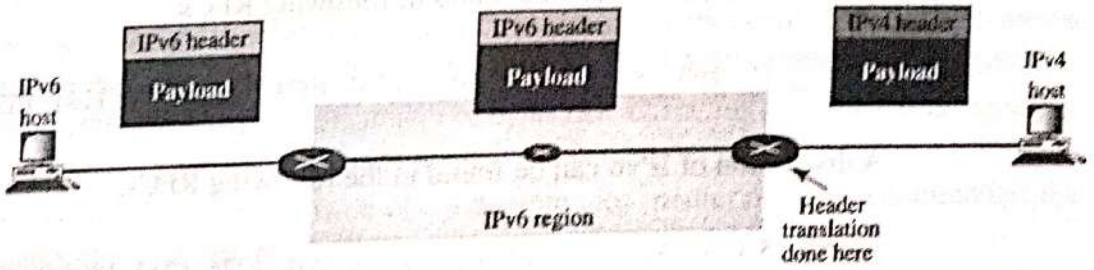
Figure 20.20 *Tunneling strategy*



Header Translation

Header translation is necessary when the majority of the Internet has moved to IPv6 but some systems still use IPv4. The sender wants to use IPv6, but the receiver does not understand IPv6. Tunneling does not work in this situation because the packet must be in the IPv4 format to be understood by the receiver. In this case, the header format must be totally changed through header translation. The header of the IPv6 packet is converted to an IPv4 header (see Figure 20.21).

Figure 20.21 *Header translation strategy*



Header translation uses the mapped address to translate an IPv6 address to an IPv4 address. Table 20.11 lists some rules used in transforming an IPv6 packet header to an IPv4 packet header.

Table 20.11 *Header translation*

Header Translation Procedure
1. The IPv6 mapped address is changed to an IPv4 address by extracting the rightmost 32 bits.
2. The value of the IPv6 priority field is discarded.
3. The type of service field in IPv4 is set to zero.
4. The checksum for IPv4 is calculated and inserted in the corresponding field.
5. The IPv6 flow label is ignored.
6. Compatible extension headers are converted to options and inserted in the IPv4 header. Some may have to be dropped.
7. The length of IPv4 header is calculated and inserted into the corresponding field.
8. The total length of the IPv4 packet is calculated and inserted in the corresponding field.

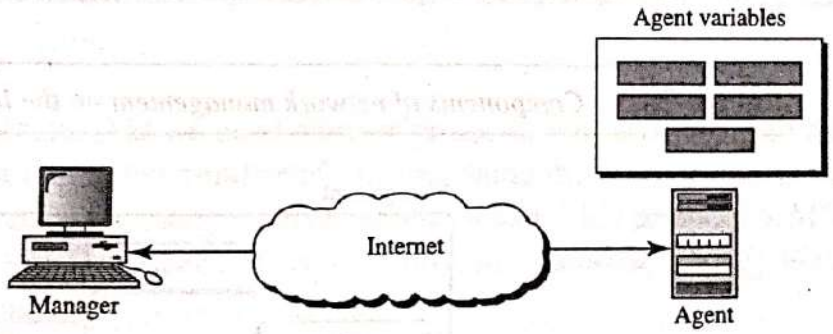
28.2 SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

The **Simple Network Management Protocol (SNMP)** is a framework for managing devices in an internet using the TCP/IP protocol suite. It provides a set of fundamental operations for monitoring and maintaining an internet.

Concept

SNMP uses the concept of manager and agent. That is, a manager, usually a host, controls and monitors a set of agents, usually routers (see Figure 28.2).

Figure 28.2 *SNMP concept*



SNMP is an application-level protocol in which a few manager stations control a set of agents. The protocol is designed at the application level so that it can monitor devices made by different manufacturers and installed on different physical networks. In other words, SNMP frees management tasks from both the physical characteristics of the managed devices and the underlying networking technology. It can be used in a heterogeneous internet made of different LANs and WANs connected by routers made by different manufacturers.

Managers and Agents

A management station, called a **manager,** is a host that runs the SNMP client program. A managed station, called an **agent,** is a router (or a host) that runs the SNMP server program. Management is achieved through simple interaction between a manager and an agent.

The agent keeps performance information in a database. The manager has access to the values in the database. For example, a router can store in appropriate variables the number of packets received and forwarded. The manager can fetch and compare the values of these two variables to see if the router is congested or not.

The manager can also make the router perform certain actions. For example, a router periodically checks the value of a reboot counter to see when it should reboot itself. It reboots itself, for example, if the value of the counter is 0. The manager can use this feature to reboot the agent remotely at any time. It simply sends a packet to force a 0 value in the counter.

Agents can also contribute to the management process. The server program running on the agent can check the environment, and if it notices something unusual, it can send a warning message, called a **trap,** to the manager.

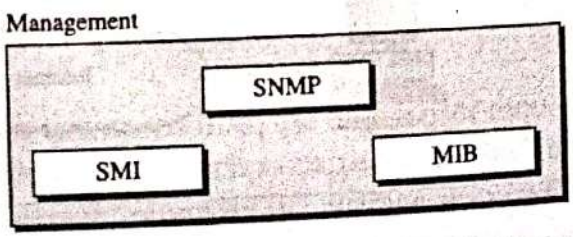
In other words, management with SNMP is based on three basic ideas:

1. A manager checks an agent by requesting information that reflects the behavior of the agent.
2. A manager forces an agent to perform a task by resetting values in the agent database.
3. An agent contributes to the management process by warning the manager of an unusual situation.

Management Components

To do management tasks, SNMP uses two other protocols: **Structure of Management Information (SMI)** and **Management Information Base (MIB).** In other words, management on the Internet is done through the cooperation of the three protocols SNMP, SMI, and MIB, as shown in Figure 28.3.

Figure 28.3 *Components of network management on the Internet*



Let us elaborate on the interactions between these protocols.

Role of SNMP

SNMP has some very specific roles in network management. It defines the format of the packet to be sent from a manager to an agent and vice versa. It also interprets the

result and creates statistics (often with the help of other management software). The packets exchanged contain the object (variable) names and their status (values). SNMP is responsible for reading and changing these values.

SNMP defines the format of packets exchanged between a manager and an agent. It reads and changes the status (values) of objects (variables) in SNMP packets.

Role of SMI

To use SNMP, we need rules. We need rules for naming objects. This is particularly important because the objects in SNMP form a hierarchical structure (an object may have a parent object and some children objects). Part of a name can be inherited from the parent. We also need rules to define the type of the objects. What types of objects are handled by SNMP? Can SNMP handle simple types or structured types? How many simple types are available? What are the sizes of these types? What is the range of these types? In addition, how are each of these types encoded?

We need these universal rules because we do not know the architecture of the computers that send, receive, or store these values. The sender may be a powerful computer in which an integer is stored as 8-byte data; the receiver may be a small computer that stores an integer as 4-byte data.

SMI is a protocol that defines these rules. However, we must understand that SMI only defines the rules; it does not define how many objects are managed in an entity or which object uses which type. SMI is a collection of general rules to name objects and to list their types. The association of an object with the type is not done by SMI.

SMI defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values.

SMI does not define the number of objects an entity should manage or name the objects to be managed or define the association between the objects and their values.

Role of MIB

We hope it is clear that we need another protocol. For each entity to be managed, this protocol must define the number of objects, name them according to the rules defined by SMI, and associate a type to each named object. This protocol is MIB. MIB creates a set of objects defined for each entity similar to a database (mostly metadata in a database, names and types without values).

MIB creates a collection of named objects, their types, and their relationships to each other in an entity to be managed.

An Analogy

Before discussing each of these protocols in greater detail, we give an analogy. The three network management components are similar to what we need when we write a program in a computer language to solve a problem.

The **lexicographic ordering** enables a manager to access a set of variables one after another by defining the first variable, as we will see in the GetNextRequest command in the next section.

SNMP

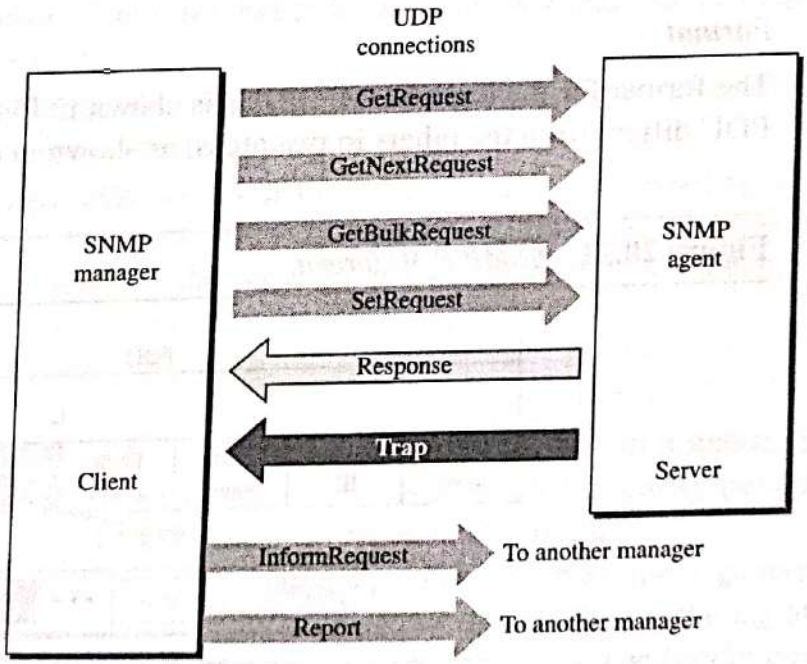
SNMP uses both SMI and MIB in Internet network management. It is an application program that allows

1. A manager to retrieve the value of an object defined in an agent
2. A manager to store a value in an object defined in an agent
3. An agent to send an alarm message about an abnormal situation to the manager

PDUs

SNMPv3 defines eight types of packets (or PDUs): GetRequest, GetNextRequest, GetBulkRequest, SetRequest, Response, Trap, InformRequest, and Report (see Figure 28.20).

Figure 28.20 *SNMP PDUs*



GetRequest The GetRequest PDU is sent from the manager (client) to the agent (server) to retrieve the value of a variable or a set of variables.

GetNextRequest The GetNextRequest PDU is sent from the manager to the agent to retrieve the value of a variable. The retrieved value is the value of the object following the defined ObjectId in the PDU. It is mostly used to retrieve the values of the entries in a table. If the manager does not know the indexes of the entries, it cannot retrieve the values. However, it can use GetNextRequest and define the ObjectId of the table. Because the first entry has the ObjectId immediately after the ObjectId of the table, the value of the first entry is returned. The manager can use this ObjectId to get the value of the next one, and so on.