Internet Protocol Suite

- Internet
- TCP/IP Suite: Protocol Model
- Network Layer: IP, ICMP
- Transport Layer: TCP, UDP, RTP
- Application Layer: Telnet, DNS, FTP, HTTP, SNMP

INTERNET

Internet is a network of networks. It includes public and private networks. It also includes wide area networks, campus area networks, as well as local area networks.

Examples of Networks:

ARPANET (Advanced Research Project Agency NETwork)

BITNET (Because It's Time NETwork)
A low cost, low speed network. It consists of over 200 universities and attaches to CERN in Europe.

CSNET (Computer Science NETwork)

MILNET (MILitary NETwork)
Originally part of the ARPANET, MILNET was partitioned in 1984.

NSFNET (National Science Foundation NETwork)
A network that provides the cross-country backbone networking connecting mainframes and supercomputers.

INTERNET STANDARDS BODIES

- Internet Architecture Board (IAB)

* Architectural oversight
* Selection of Internet Engineering Steering Group (IESG)
* Standard process oversight
* Editorial management of RFCs and Internet assigned numbers authority (IANA)
* External liaison

- Internet Engineering Steering Group (IESG)

* Technical management of IETF activities
* Technical management of standards process

- Internet Engineering Task Forces (IETF):

* A self-organized group of engineers who get together to solve Internet related problems
* 9 areas and more than 100 working groups
  - > Applications (APP)
    Protocols seen by user programs, such as e-mail and the Web
  - > General (GEN)
    Catch-all for WGs that don't fit in other areas
  - > Internet (INT)
    Different ways of moving IP packets and DNS information
  - > Operations and Management (OPS)
    Administration and monitoring
  - > Routing (RTG)
    Getting packets to their destinations
  - > Security (SEC)
    Authentication and privacy
  - > Transport (TSV)
    Special services for special packets
  - > User Services (USV)
    Support for end users and user support organizations

REQUEST FOR COMMENT (RFC)

IETF publishes RFCs, which include

- Any Internet communication topic
- Protocols proposed by member of Internet community
- Track evolutionary standards process
- IAB facilitates ratification

There are six kinds of RFCs. They are:
- Proposed standards
- Draft standards
- Internet standards (sometimes called "full standards")
- Experimental protocols
- Informational documents
- Historic standards

Only the first three (i.e., proposed, draft, and full) are standards within the IETF.

After an Internet Draft has been sufficiently discussed and there is rough consensus that it would be a useful standard, it is presented to the IESG for consideration. The IESG then announces an IETF-wide last call, which lasts for two or four weeks. If the IESG approves the draft to become an Internet Standard, they ask the RFC Editor to publish it as a **Proposed Standard**. After it has been a Proposed Standard for at least six months, the RFC's author (or the appropriate WG chair) can ask for it to become a **Draft Standard**. Before that happens, someone needs to convince the appropriate Area Director that there are at least two independent, interoperable implementations of each part of the standard. A few years after a document has been a Draft Standard, it can become an **Internet Standard**, also known as "full standard."

You can download the RFCs from several web sites.

TRANSMISSION CONTROL PROTOCOL/ INTERNET PROTOCOL (TCP/IP)

TCP/IP is a suite of protocols that are first designed for the Defense Advanced Research Project Agency (DARPA) network called ARPAnet in the early 1970s. In the early 1980s, TCP/IP was included as an integral part of Berkeley's UNIX version 4.2. Today, it is the protocol used by ARPAnet, MILnet and many other networks.

TCP/IP is organized into four conceptual layers.

Network Interface Layer:

This layer is responsible for transmitting datagram over the physical medium. It corresponds to OSI's physical and data link layers.

Network Layer:

This layer encapsulates the packet into datagram, and performs routing. It corresponds to OSI's network layer.

Transport Layer:

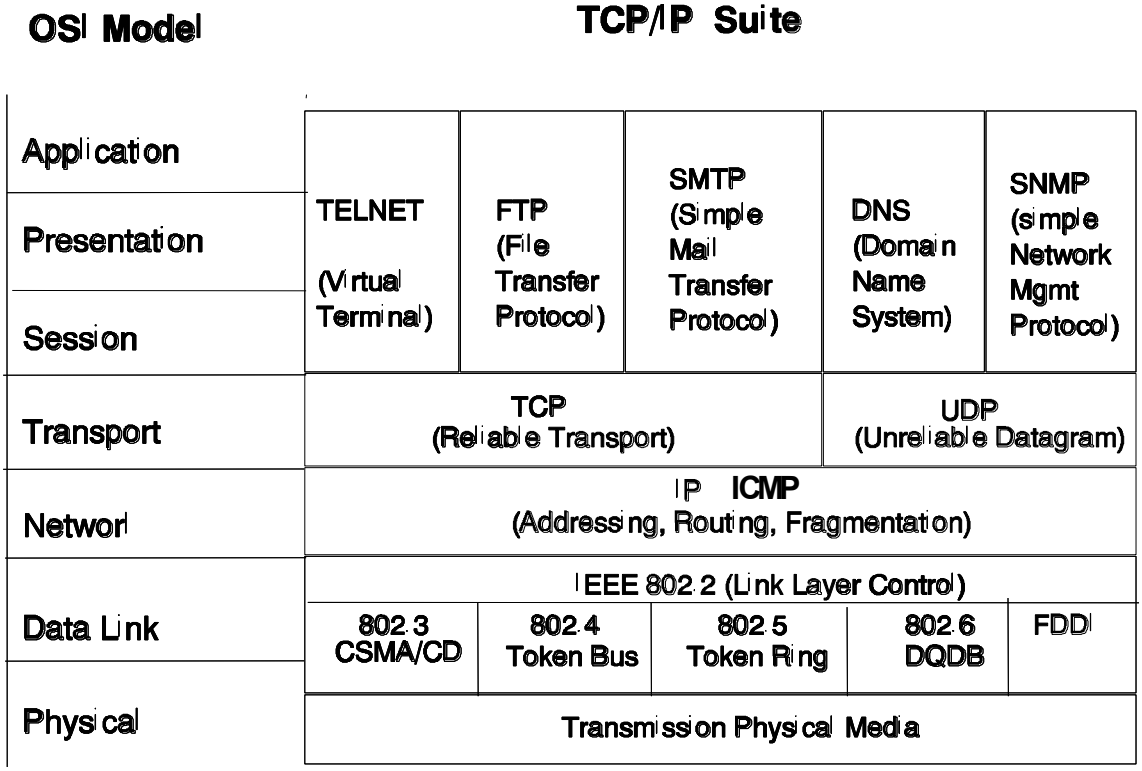
This layer is responsible for providing communication between applications residing in different hosts. It provides either reliable service (TCP) or unreliable service (UDP).

Application Layer:

This layer provides applications such as file transfer (FTP), mail transfer (SMTP), remote terminal access (Telnet), address assignment (DHCP), network management (SNMP). It could use either TCP or UDP for transport.

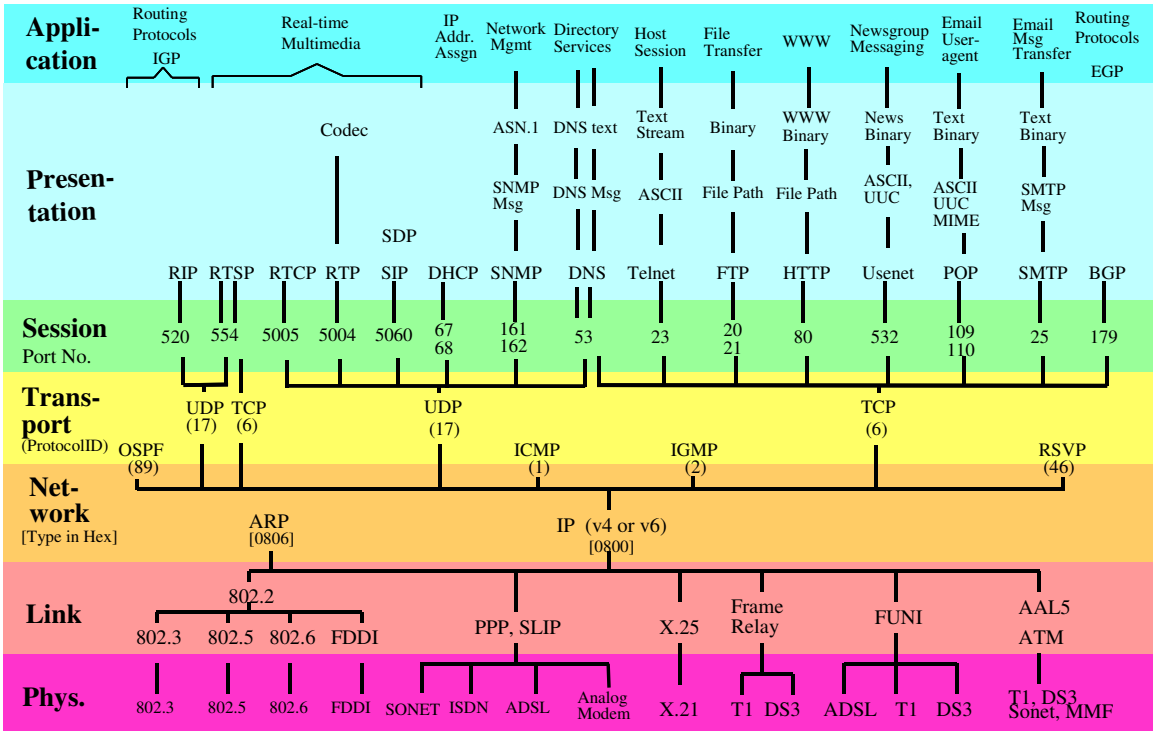
TRANSMISSION CONTROL PROTOCOL/ INTERNET PROTOCOL (TCP/IP)

The correspondence between the OSI reference model and the TCP/IP protocol suite is as follows:



In the above, the data link layer and the physical layer are assumed to be LAN. However, it can be other protocols, such as PPP, frame relay, ATM, X.25, etc. Note that X.25 includes the network layer.

Illustration of TCP/IP Protocol Suite

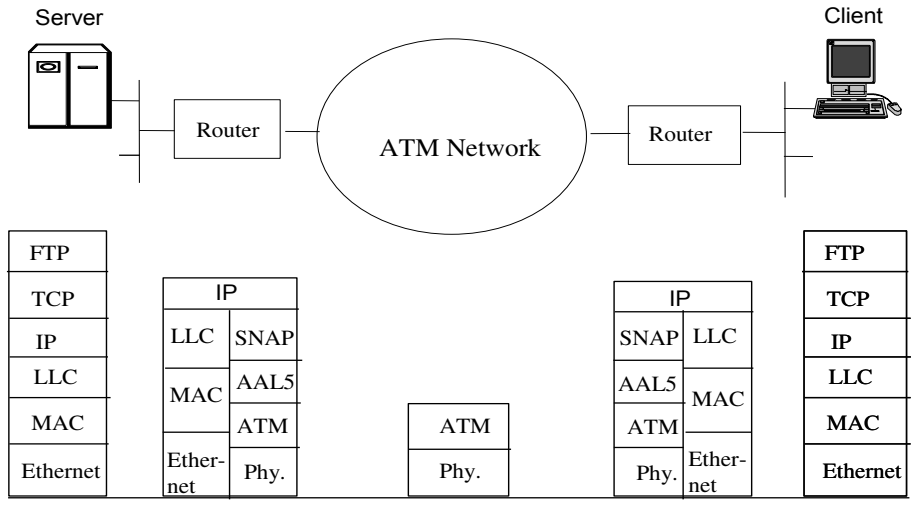


AAL	ATM Adaptation Layer
ARP	Address Resolution Protocol (RFC903)
ASN.1	Abstract Syntax Notation 1
ATM	Asynchronous Transfer Mode (RFC1483, 1577)
BGP	Border Gateway Protocol (RFC1771)
Codec	Coding/encoding spec
DNS	Domain Name System (RFC1035)
DHCP	Dynamic Host Configuration Protocol (RFC1541)
EGP	Exterior Gateway Protocol
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol (RFC959)
FUNI	Frame based User Network Interface (RFC1490)
HTTP	HyperText Transfer Protocol (RFC2616)
ICMP	Internet Control Message Protocol (RFC792)
IGMP	Internet Group Management Protocol (RFC2236)
IGP	Interior Gateway Protocol
IP	Internet Protocol (IPv4 RFC791, IPv6 RFC1752)
ISDN	Integrated Service Digital Network
MIME	Multipurpose Internet Mail Extensions (RFC 1521)
MMF	Multimode Fiber
OSPF	Open Shortest Path First (RFC1583)
POP	Post Office Protocol (RFC1725)
PPP	Point-to-Point Protocol (RFC1661)
RIP	Routing Information Protocol (RFC1058)
RSVP	Resource reSerVation Protocol (RFC2205)
RTP	Real-time Transport Protocol (RFC1889)
RTCP	RTP Control Protocol (RFC1889)
RTSP	Real-time Streaming Protocol (RFC2326)
SDP	Session Description Protocol (RFC2327)
SIP	Session Initiation Protocol (RFC2543)
SLIP	Serial Line Interface Protocol (RFC1055)
SMTP	Simple Mail Transfer Protocol (RFC821)
SNMP	Simple Network Protocol (RFC1441)
TCP	Transport Control Protocol (RFC793)
Telnet	Telnet Network Terminal (RFC854)
UDP	User Datagram Protocol (RFC768)
UUC	User user encoded format
Usenet	Usenet Newsgroup Messaging Protocol (RFC1036)
WWW	World Wide Web (RFC1630)

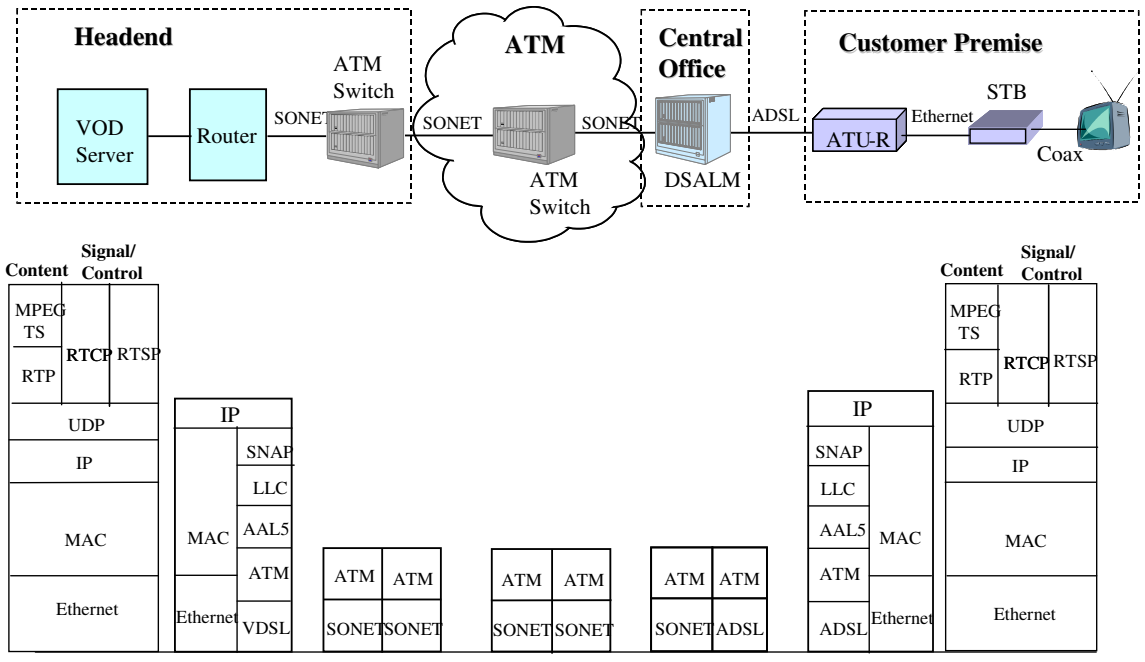
RFC	Request For Comment (Internet Engineering Task Force specification document)

EXAMPLES OF PROTOCOL STACKS

The following figure shows an example network with file transfer applications. The end-to-end protocol stack is also illustrated. The lower layer protocols, such as Ethernet and ATM, have been discussed earlier.



The following figure shows a network that supports video-on-demand using ADSL. The protocol stack is also presented.

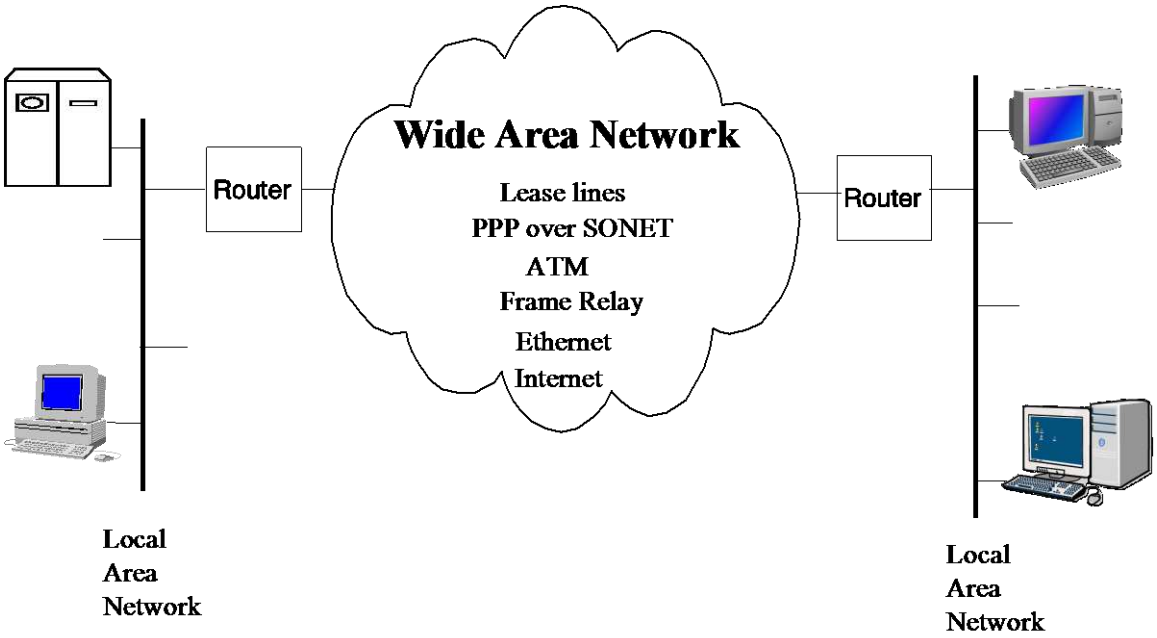


NETWORK INTERFACE LAYER: LINK AND PHYSICAL LAYERS

Possible networking media:

Local Area Network
- IEEE 802.3 (Ethernet)
- IEEE 802.5 (Token Ring)

Wide Area Network
- PPP (Point-to-Point Protocol) over SONET
- MPLS over PPP
- Frame relay network
- ATM network
- Ethernet Network



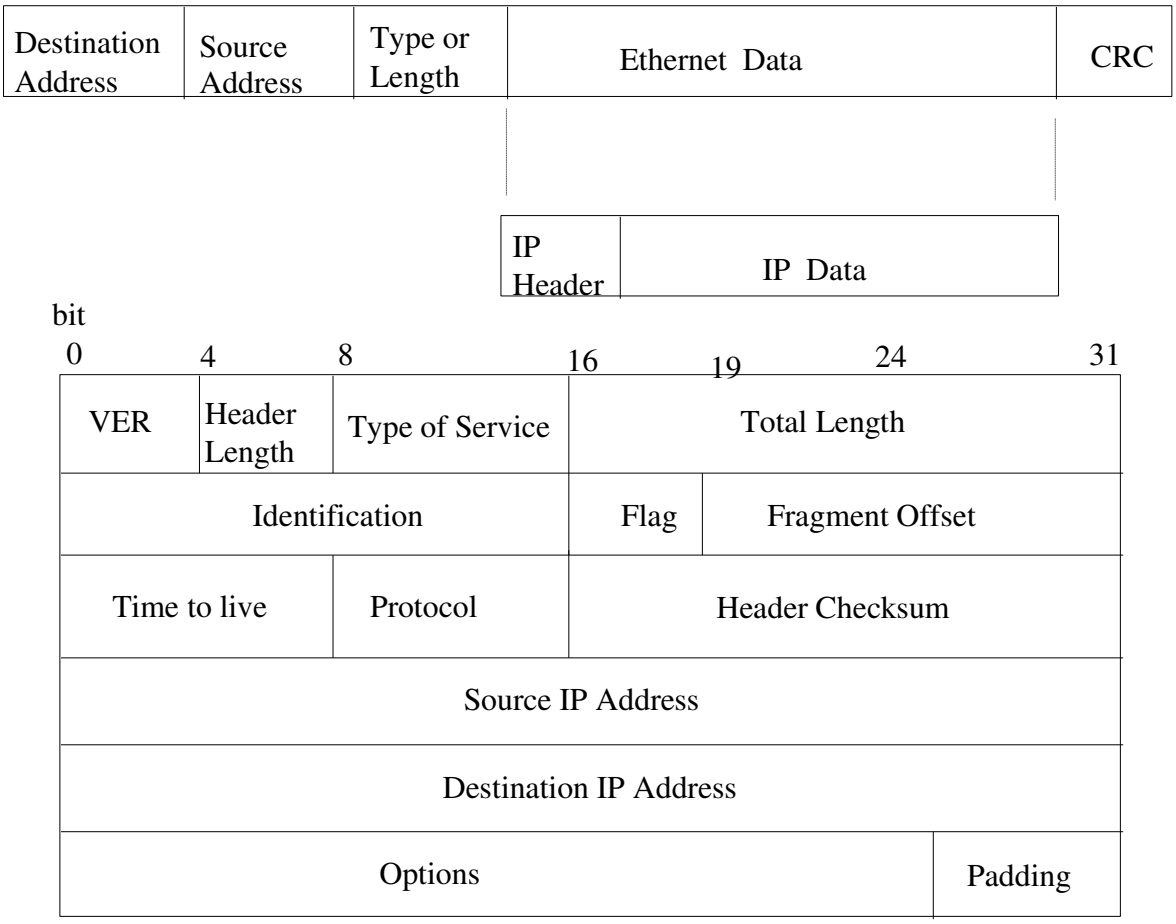
NETWORK LAYER: INTERNET PROTOCOL

Functions performed by IP:

- Fragmentation and re-assembly to fit to the maximum packet size allowed by a network.
- Routing using the IP addresses.
- Header error detection.
- Identification of Higher layer protocol.

NETWORK LAYER: INTERNET PROTOCOL

An IP packet could be encapsulated as the data of an Ethernet frame. The format of IP version 4 [1] is as follows:



The above figure shows the header of IP version 4. We will discuss IP v6 later.

1 RFC 791, "Internet Protocol"
RFC 792, "Internet Control Message Protocol"
RFC 815, "IP Datagram Reassembly Algorithm"

INTERNET PROTOCOL: IP V4

Header for IP v4:

Version	Version of IP format
IHL	Internet Header Length expressed in terms of 32-bit (4 bytes) words
TOS	Type of Service
Total Length	Total length of the datagram in octets
Identification	Value assigned by the sender to aid in assembling the fragments of the datagram
Flag	Used to control fragmentation

Flag bit number	Value
bit 0 - reserved	0 always zero
bit 1 - fragment	0 = may fragment 1 = don't fragment
bit 2 - more fragment	0 = last fragment 1 = more fragments

Fragment Offset	Indicates where within the original datagram this fragment belongs
TTL	Time to live. Indicates the maximum time the datagram is allowed to remain in the Internet
Protocol	Indicates the next level of protocol used in the data portion of the Internet datagram
Header Checksum	
Source Address	Sender IP address
Destination Address	Receiver IP address
Option	
Padding	

TYPE OF SERVICE (TOS)

The Type of Service (TOS) field in the IP protocol is used to indicate the quality of service desired. However, there is no guarantee that other devices on the Internet will honor the request.

bits

0 1 2	3	4	5	6 7
Precedence	D	T	R	nor used

Precedence Bits

Precedence bits indicate the importance of the IP datagram.

Value	Precedence
111	Network Control
110	Interwork Control
101	CRITIC
100	Flash Override
011	Flash
010	Immediate
001	Priority
000	Routine

TYPE OF SERVICE (TOS)

D-T-R Bits in the TOS Field

Three bits used to indicate the quality of service desired.

D-bit Delay bit
D=1 for low delay, D=0 for normal delay

T-bit Throughput bit
T=1 for high throughput, T=0 for normal throughput

R-bit Reliability bit
R=1 for high reliability, R=0 for normal reliability

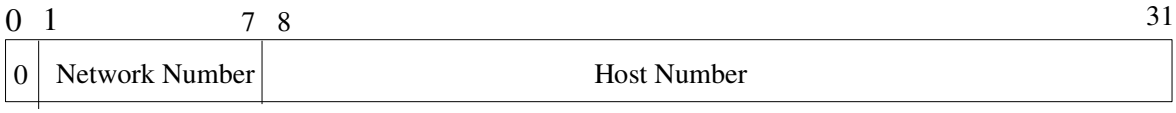
INTERNET ADDRESS

When IP was standardized in 1981, the specification required that each user be assigned a unique 32-bit Internet address. Some network equipment, such as routers may have interfaces to several networks. In this case, each interface needs a unique IP address. IP addresses are expressed as four decimal numbers, each separated by a dot. This is called **dotted-decimal notation**. Examples are 2.15.38.9, 129.19.140.34, 210.120.19.8.

With the rapid growth of the Internet users, the available IP addresses are running out quickly. In the next generation IP, i.e., IPv6, a new IP address has been defined. We will discuss that later.

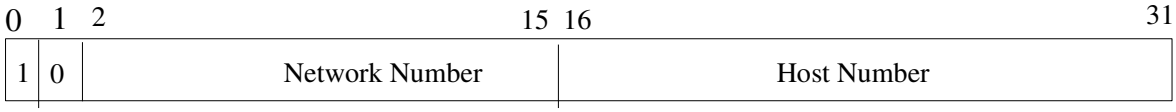
The 32-bit IP address consists of two parts: network address and the host address. In order to provide the flexibility required to support networks of different sizes, the IP addresses are divided into five classes: A, B, C, D and E.

Class A IP address



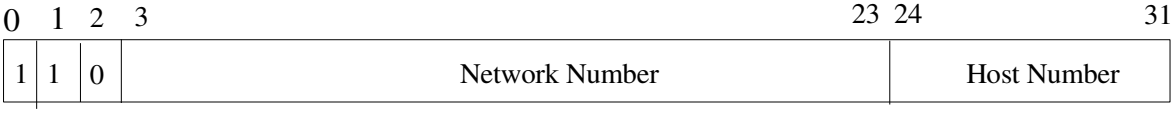
A class A address has the leading bit set to 0, a 7-bit network number and a 24-bit host number. 126 (i.e., 2^7 -2) network numbers can be assigned. Network numbers with all 0s and all 1s have been reserved. Each network can have up to 16,777,214 (i.e., 2^24 -2) hosts. Class A networks are also referred to as "/8s" since they have an 8-bit network address. These networks are identified by their network numbers, such as 5.0.0.0/8, 53.0.0.0/8, 122.0.0.0/8.

Class B IP address



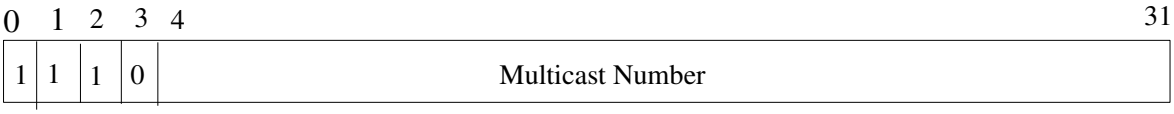
A class B address has the two leading bits set to 1-0, a 14-bit network number field, and a 16-bit host number field. 16,382 class B networks can be assigned with up to with 65,534 hosts in each network. Class B networks are also referred to as "/16s" since they have a 16-bit network number field. Examples of class B networks are 129.52.0.0/16, 190.3.0.0/16.

Class C IP address



A class C address has the leading bits set to 1-1-0, a 21-bit network number field, and an 8-bit host number field. 2,097,152 class C networks may be defined with up to 254 hosts per network. Class C networks are also referred to as “/24s” since they have a 24-bit network address.

Class D IP address



Class D is used for multicast. The leading 4 bits are set to 1-1-1-0.

Class E IP address

Class E addresses have their leading 4 bits set to 1-1-1-1, and are reserved for experimental use.

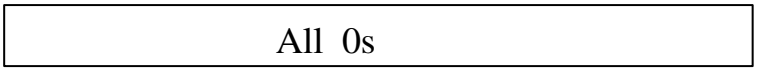
Note that the address spaces of classes A, B and C are 50%, 25% and 12.5% respectively of the total IPv4 unicast address space.

Exercise: Prove that the address spaces of classes A, B and C are 50%, 25% and 12.5% respectively of the total IPv4 unicast address space.

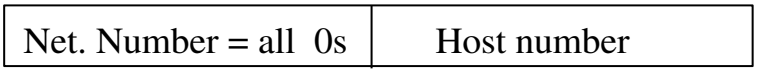
The detailed assignment of public and private Ipv4 addresses is listed below:

Address Class	Address Range	Total number of networks	Max nodes per network	Must be Registered
A	1.0.0.0 to 9.255.255.255 and 11.0.0.0 to 126.255.255.255	125	16,777,214	Yes
	10.0.0.0 to 10.255.255.255	Private address space[2]		No
	127.0.0.0 to 127.255.255.255	Loopback address for testing TCP/IP within a node		No
B[3]	128.0.0.0 to 172.15.255.255 and 172.32.0.0 to 191.255.255.255	16,368	65,534	Yes
	172.16.0.0 to 172.31.255.255	Private address space		No
C	192.0.0.0 to 192.167.255.255 and 192.169.0.0 to 223.255.255.255	2,096,896	254	Yes
	192.168.0.0 to 192.168.255.255	Private address space		No
D	224.0.0.0 to 239.255.255.255	Multicast group addresses		
E	240.0.0.0 to 255.255.255.254	Reserved		

There are some special IP addresses that have been assigned for special purposes. They are as follows:

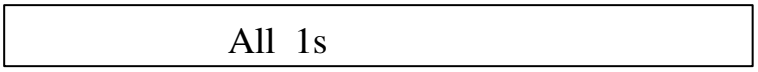


This address is allowed only at the system startup and is used to indicate this host. It is never a valid destination host address.

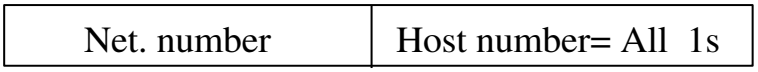


This address is allowed only at the system startup and is used to indicate a host system in this network. It is never a valid destination host address.

2. Y. Rekhter, et. Al., “Address Allocation for Private Internets,” RFC 1918, Feb., 1996.
3. 169.254.0.0 through 169.254.255.255 are reserved for use as self-assigned addresses when DHCP fails or is not available.



This address is used for broadcast within the local network. An example is broadcast within a local area network. It is never a valid source address.



This address is used for broadcast within the network identified by the network number. It is not a valid source address.

SUBNETTING AND MASK

For most organizations (e.g., companies, universities, government agencies), the network consists of many small networks, such as local area networks. These small networks may be geographically separated or owned by different departments. However, these networks are interconnected to form a network, which may be class A, B or C. These small networks are the subnetworks. The network manager likes to assign IP addresses, which identify the subnetworks that the users are connected to. To handle this situation, RFC 950 has defined a procedure to support the subnetting.

Subnetting is to further divide the host number field into subnet number field and host number field. In another word, it makes the two-level address hierarchy (network number + host number) into a three-level hierarchy (network number + subnet number + host number). The address format is as follows;

Network Number	Subnet Number	Host Number

The combination of the network number and the subnet number is called the **extended network number**. Internet routers use only the network number of the destination host address to route traffic to a network (or the gateway of that network). The network then uses the extended network number to route the traffic to the particular subnetwork, which the destination user is connected.

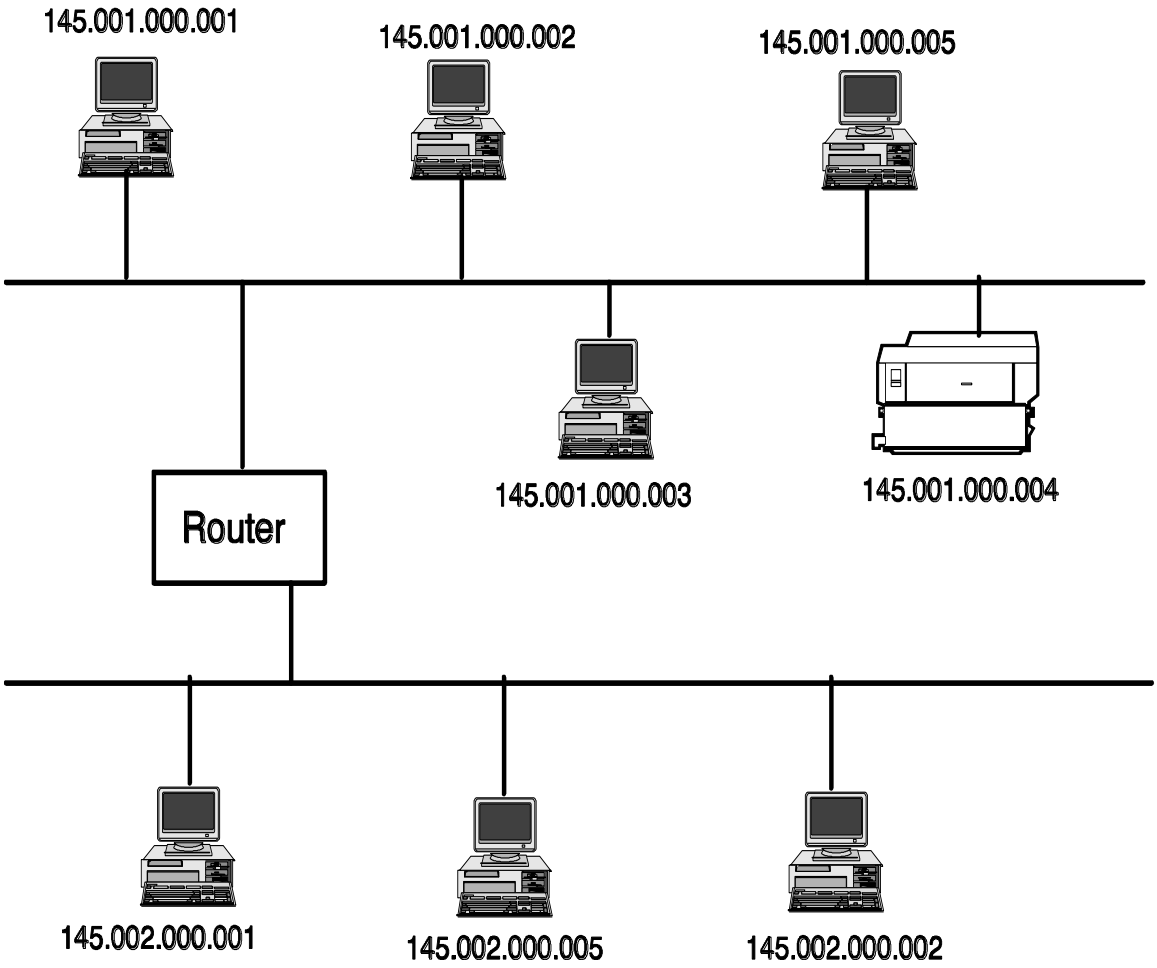
While the network number field is 8, 16 or 24 bits long, the subnet number field can be of any length. The extended network number field has traditionally been identified by the **subnet mask**. The bits in the mask and the Internet address have a one-to-one correspondence. The bits of the subnet mask are set to 1 if the system (i.e., router, switch, gateway, etc.) examining the address should treat the corresponding bit in the IP address as part of the extended network number field. The bits in the mask are set to 0 if the system should treat the bit as part of the host number.

For example, if you have a class B network (i.e., /16 address space) of 148.75.00 and there is no subnetworks, then the subnet mask is 255.255.0.0. However, if you have many subnetworks and you want to use the entire third octet to represent the subnetwork number, you have a subnet mask of 255.255.255.0.

If you have a class B network and you have 16 subnetworks with each subnetwork has more than 256 users connecting to it, you may want to use the first 4 bits in the third octet as the subnet number field. In this case, the subnet mask is 255.255.240.0.

Exercise:
Explain why in the last example, the subnet mask is 255.255.240.0.

EXAMPLE NETWORK ADDRESS



Question:
What class of network address do these two networks belong to?

ADDRESS RESOLUTION PROTOCOL

Problem:

Ethernet address and Internet address are different. How can a computer find out the Ethernet physical address for the destination node? In another word, how to map the destination node's Internet address to its physical address?

Solution:

The address resolution protocol (ARP) provides a method to dynamically find out the destination node's physical address through broadcast mechanism.

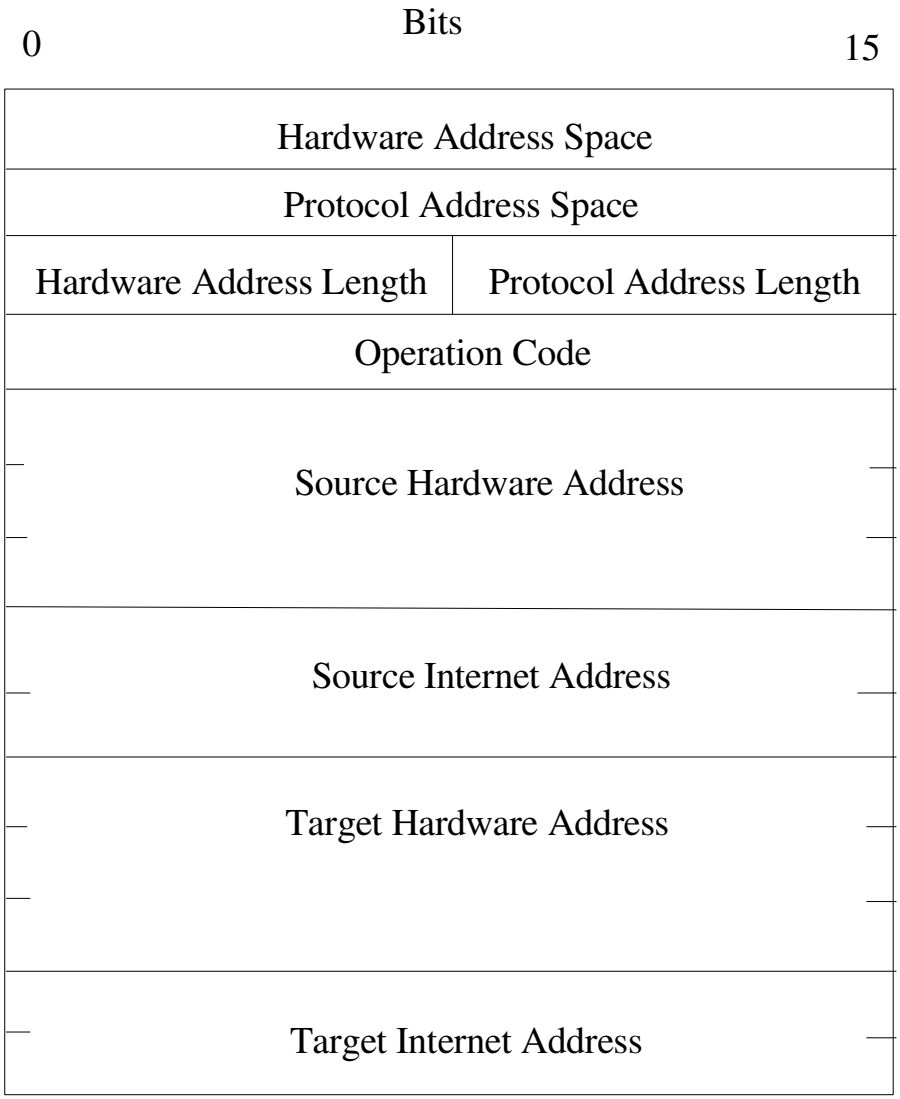
A node broadcasts an ARP packet with the destination node's Internet address to every node in the same Ethernet. The node, which has the specific Internet address, replies by sending a response with its Ethernet physical address.

REVERSE ADDRESS RESOLUTION PROTOCOL

The reverse address resolution protocol (RARP) is used, when a node (e.g., a diskless workstation) is initializing and does not know its own Internet address. (Note: it knows its own Ethernet physical address, which is assigned by the manufacturer.) Because the Internet address is recorded on the Ethernet controller, and the physical address is available to the node.

Using the broadcast capability, the node broadcast a RARP packet with its physical address to all the nodes on the same Ethernet. Upon receiving the RARP packet, the controller responds by sending a packet with the inquiring node's Internet address to the inquiring node.

ARP PACKET FORMAT

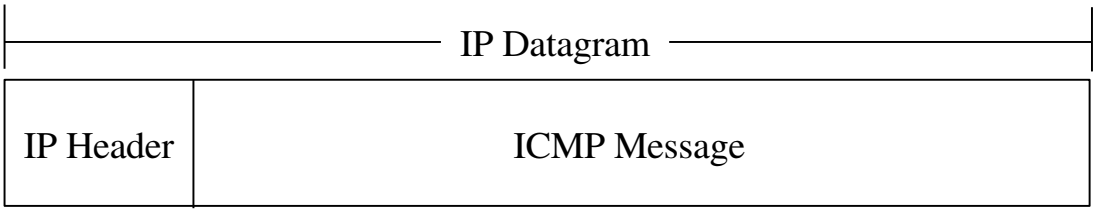


Hardware Address Space: type of hardware used in the network interface layer (set to 1 for Ethernet)
Protocol Address Space: protocol used in the network interface layer
Hardware Address Length: in octets (=6 for Ethernet)
Protocol Address Length: in octets (=4 for IP)
Operation Code: type of packet (e.g., ARP request, ARP response, etc.)

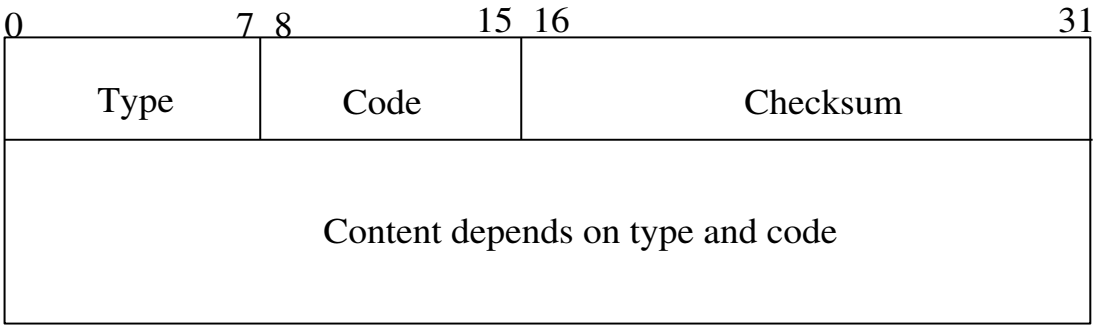
INTERNET CONTROL MESSAGE PROTOCOL (ICMP)

ICMP is used to report errors or other conditions that require attention. ICMP is considered as in the IP layer. It is usually acted on by either the IP protocol or the higher layer protocols.

Each ICMP message is attached to an IP header described previously. The format is as follows:



Format of the ICMP message is:



Type field is used to identify the ICMP message. There are 15 different types. Each type may have different codes to provide further information. The 16-bit checksum covers the entire ICMP message.

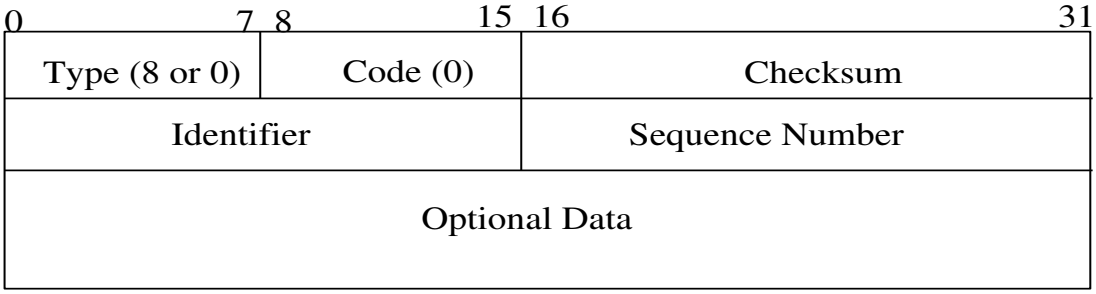
The message types and their codes are as follows:

Type	Code	Description
0	0	Echo reply (Ping reply)
3		Destination unreachable (reporting problems)
	0	Network unreachable
	1	Host unreachable
	2	Protocol unreachable
	3	Port unreachable
	4	Fragmentation needed but do not fragment bit set
	5	Source route failed
	6	Destination network unknown
	7	Destination host unknown
	8	Source host isolated
	9	Destination network administratively prohibited
	10	Destination host administratively prohibited
	11	Network unreachable for TOS
	12	Host unreachable for TOS
	13	Communication administratively prohibited by filtering
	14	Host precedence violation
	15	Precedence cutoff in effect
4	0	Source quench (for flow control)
5		Redirect
	0	Redirect for network
	1	Redirect for host
	2	Redirect for TOS and network
	3	Redirect for TOS and host
8	0	Echo request (ping)
9	0	Route advertisement
10	0	Route solicitation
11		Time exceeded (reporting TOL status)
	0	TOL equals 0 during transit
	1	TOL equals 0 during reassembly
12		Parameter problem (reporting IP datagram problem)
	0	IP header bad
	1	Required option missing or incorrect
13	0	Timestamp requested
14	0	Timestamp reply
15	0	Information request (obsolete)
16	0	Information reply (obsolete)
17	0	Address mask request (used by diskless workstation)
18	0	Address mask reply (used by diskless workstations)

In the following, we will discuss some example ICMP messages. We will discuss their functionalities and their formats.

Echo request (Ping)/ Reply

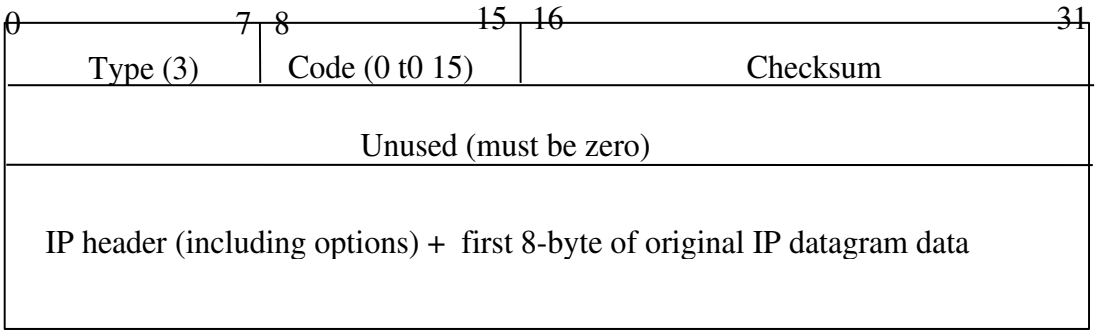
An echo request is sent by a host to a destination host to test whether the destination host is alive and reachable. The destination host, once receives the echo request, must return an echo reply message to the originating host. Format of the echo request/reply is as following:



Type=8 or 0 indicates an echo request or reply. Identifier and sequence number are used to match the reply to the request. In the UNIX of echo request (called ping), the identifier field is set to the process ID of the originating process. Sequence number is used, because there may be more than one request sent. The optional data field contains data to be returned by the destination host.

Destination unreachable

The ICMP “destination unreachable” message can be sent by a router when it receives an IP datagram that it can not deliver or forward. The format of the ICMP message is as following:



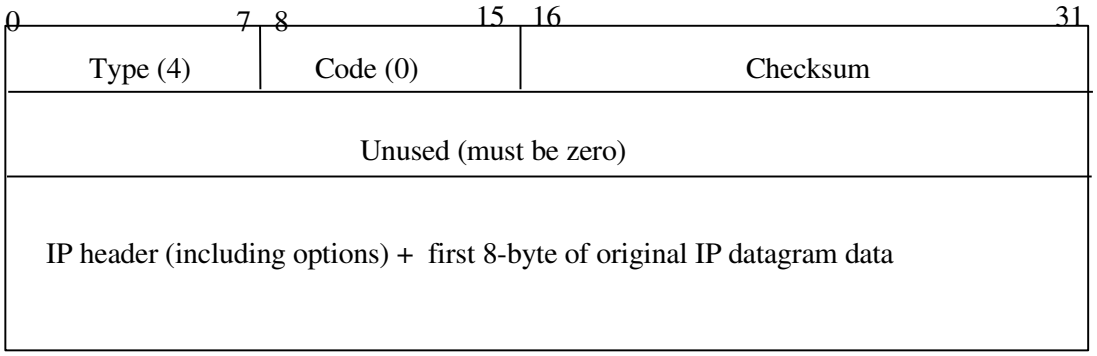
The code field describes the problem. ICMP message includes a short prefix of the IP datagram that can not be delivered. This information will let the originating host know exactly which datagram causes problem. Note that the first 8-byte of the IP datagram data could include the

whole UDP header or the first 8-byte of the TCP header. In either case, it includes the port number.

Source quenching

ICMP “source quench” error message maybe sent by a router to a host, if its buffer has been overflown. When the router buffer overflow occurs, the router discard the datagrams received. The router sends this message to request the originating host to slow down the rate of sending datagrams to the router. This is one of the datagram flow control methods.

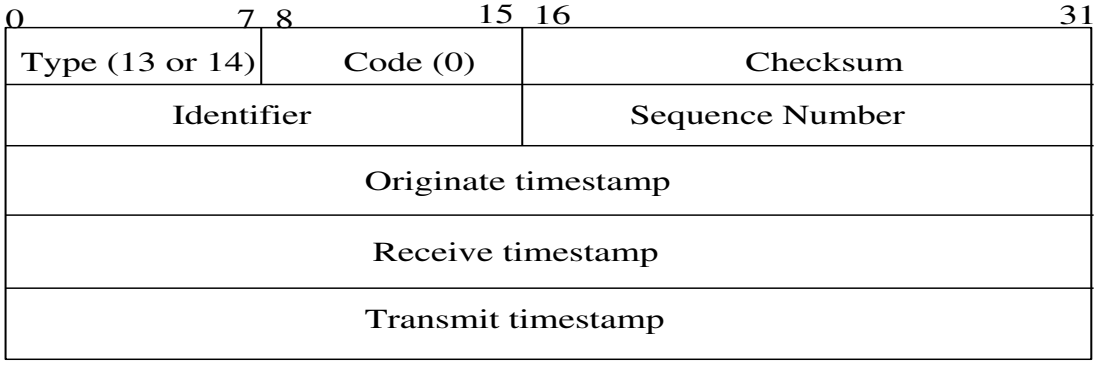
The format of the source quenching error message is as follows:



The IP header and the first 8-byte of the IP datagram data will inform the originating host the datagram discarded.

Timestamping

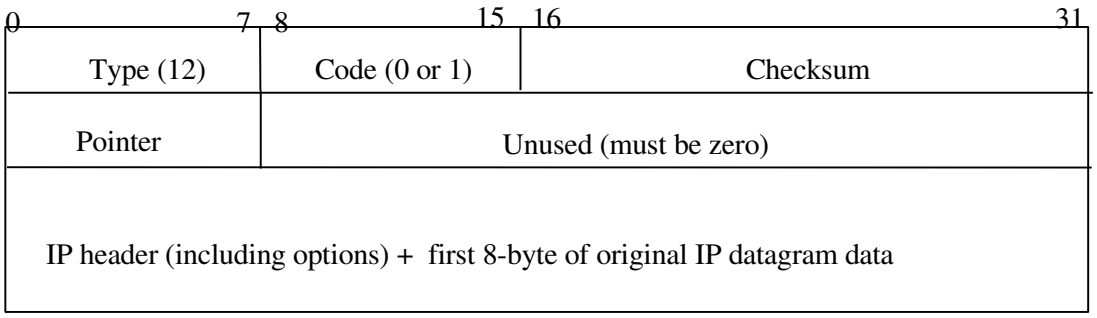
A host can send a timestamp request to another host to request that destination host return its current value of the time of the day.



The timestamp fields specify times given in milliseconds since midnight, universal time. The originate timestamp field is filled by the originating host before the datagram is sent, the receive field is filled by the destination host immediately upon receiving the request, and the transmit field is filled by the destination host immediately before the reply is transmitted.

Reporting incorrect datagram headers

When a host or a router finds problems with an IP datagram received, it sends an ICMP “parameter problem” message to the original host. The format is as follows:



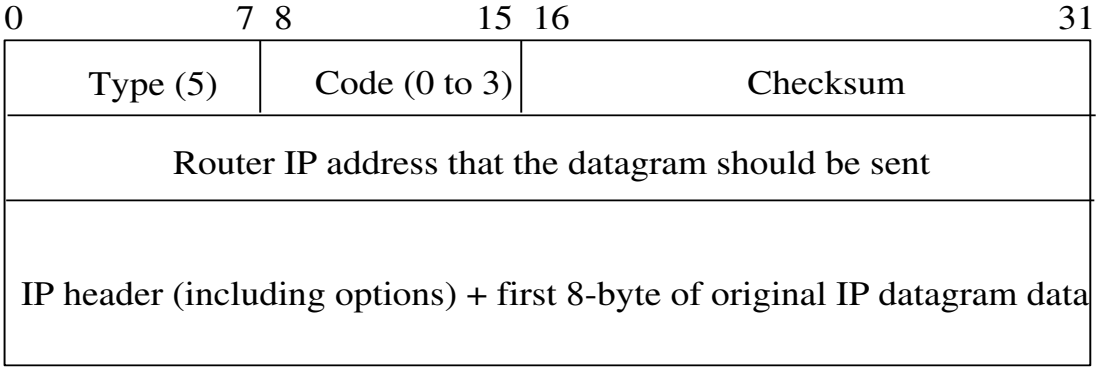
The code indicates the problem. The pointer field identifies the octet in the datagram’s header that causes problems.

Redirect

A router sends the ICMP “redirect error” message to the sender of an IP datagram it has received when this IP datagram should have been sent to another router. This could happen when a host or a router has only minimum routing knowledge. This would help the host or router to build up a better routing table over time.

An example of the ICMP redirect is a host is connected to an Ethernet, which has two routers (routers A and B) connected to it. The host may send an IP datagram to router A. Based on its routing table, router A may find that the IP datagram should be sent to router B. Thus, router A will send ICMP redirect error messages to the host and ask it to send datagrams to router B.

The format of the ICMP redirect error message is as follows:



Other ICMP messages include router solicitation and router advertisement. They are used by router in the router discovery process.

INTERNET PROTOCOL: IP V6[4]

With the rapid growth of the number of host computers that are connected to the Internet and require IP addresses, it became apparent that the address allocation in IP v4 (i.e., 32 bits) is not enough. In addition, IP v4 does not have security features. So, in 1992, Internet Engineering Task Force (IETF) issued a call for proposal for a next generation IP. In January 1995, RFC 1752, "The Recommendation for the IP Next Generation Protocol", was published. In 1995 and 1996, many RFCs[5] dealing with many aspects of IP version 6 were published.

An Ipv6 packet consists of the mandatory IPv6 header and several extension headers. The IPv6 header contains the information such as version number, source and destination IP address. The extension headers are:

- Hop-by-Hop Options Header: This header carries optional information that must be examined by every node along a packet's delivery path. One example is the jumbo payload option, which is used to declare the length of the payload greater than 64Kbytes.

4 W. Stallings, "IPv6: The New Internet Protocol," IEEE Communications Mag., pp. 96-108, July, 1996.
R. M. Hinden, "IP Next Generation: Overview," Communications of the ACM, vol. 39, no. 6, pp. 61-71, June 1996.

5 RFC 1752, "The Recommendation for the IP Next Generation Protocol"
RFC 1825, "Security Architecture for the Internet Protocol"
RFC 1826, "IP Authentication Header"
RFC 2460, "Internet Protocol, Version 6 Specification"
RFC 1884, "IP Version 6 Address Architecture"
RFC 1885, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol version (Ipv6) Specification"
RFC 1933, "Transition Mechanisms for Ipv6 Hosts and Routers"
RFC 2406, "IP Encapsulating Payload (ESP)"

- Routing Header: This header contains a list of intermediate nodes to be visited on the way to the destination. It also contains the number of explicitly listed intermediate nodes still to be visited. Each node must update the list, puts the next node address to the top of the list, and decreases the number of nodes to be visited.

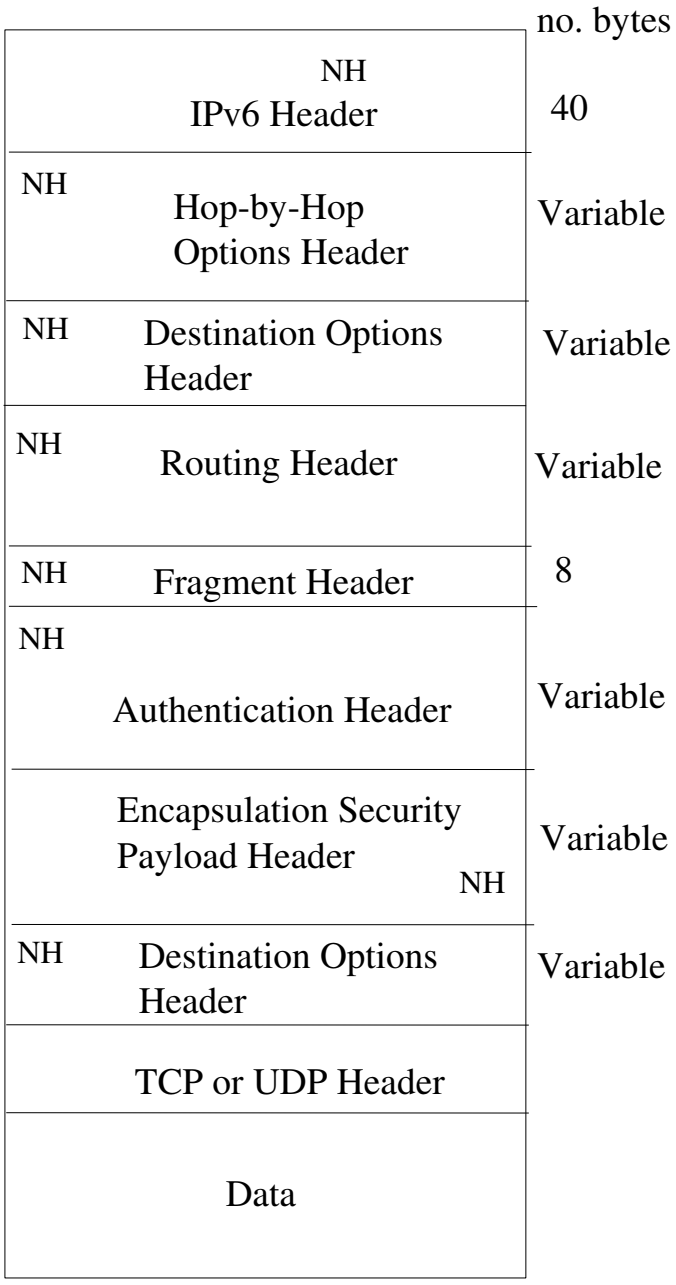
- Fragment Header: This header contains the fragment offset (like IPv4), M flag (for more fragments), and identification for the packets that consist of the same higher layer data. This header is used by an IPv6 source to send a packet larger than would fit in the path MTU to its destination. In IPv6, fragmentation is performed only by source nodes, not by routers along the path.

- Authentication Header: This header provides support for data integrity and authentication of the IP packets. It contains the security parameter index.

- Encapsulation Security Payload (ESP) Header: This header provides support for privacy. For privacy, the user can encrypt either the transport layer only (e.g., TCP, UDP), known as transport-mode ESP, or the entire IP packet, know as tunnel-mode ESP.

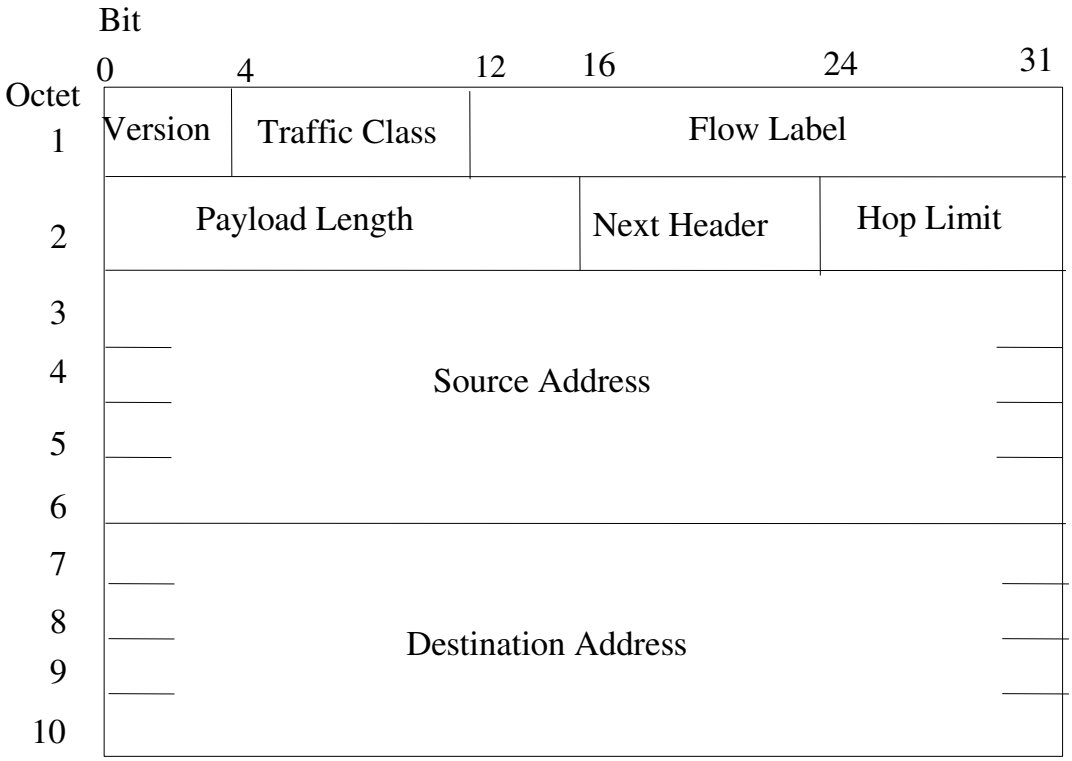
- Destination Options Header: This header contains specific information to be processed by the destination node. There could be two destination options headers. If there is a routing header, the destination options header before it shows the options to be processed by the first destination and the subsequent destinations listed in the routing header. The second destination options header, which is also the last header, provides the options to be processed by the last destination.

The following is a typical IPv6 packet:



Where NH is the 8-bit Next Header field, which indicates the type of header or protocol ID that immediately follows.

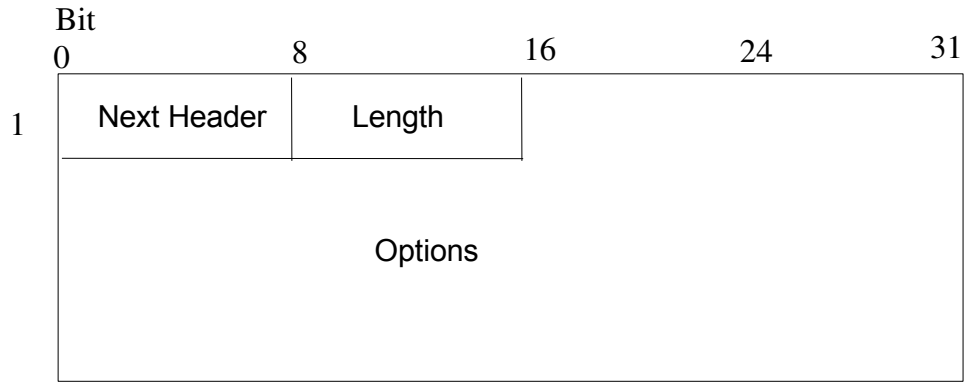
The format of the IPv6 base header is as following:



Note that the length of the IPv6 header is 40 octets. The fields are:

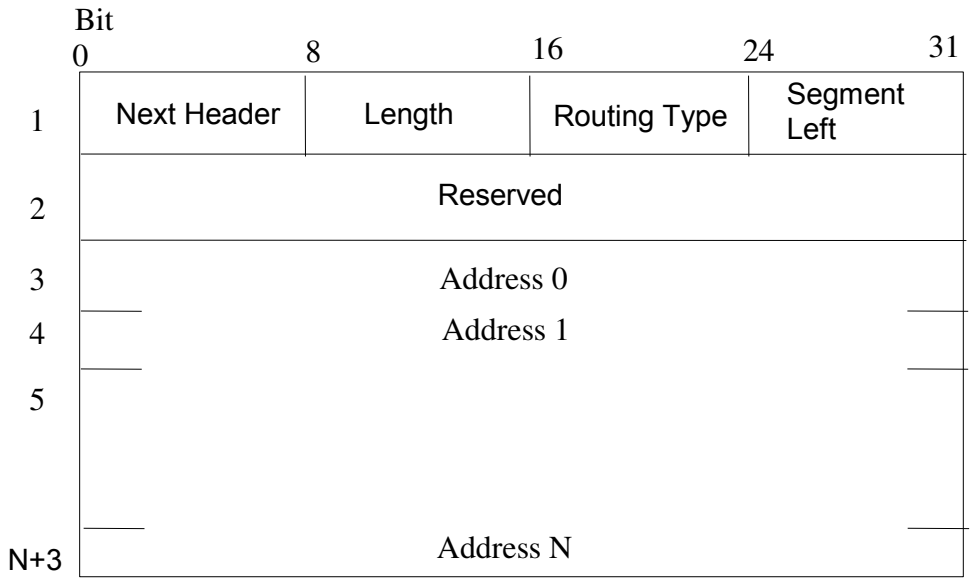
- Version (4 bits): 6 for IPv6.
- Traffic Class (8 bits): Indicates the class and priority of the packet.
- Flow Label (24 bits): May be used by the host to label a sequence of packets that require special handling. It is used by DiffServ (differentiated Services).
- Payload Length (16 bits): Length of the remainder of the IPv6 packet following the header.
- Next Header (8 bits): Indicates the type of header immediately follows the IPv6 header.
- Hop Limit (8 bits): The remaining number of allowable hops for this packet.
- Source Address (128 bits): The IP address for the source host.
- Destination Address (128 bits): The IP address for the destination.

The format of the hop-by-hop options and the destination options headers is as following:



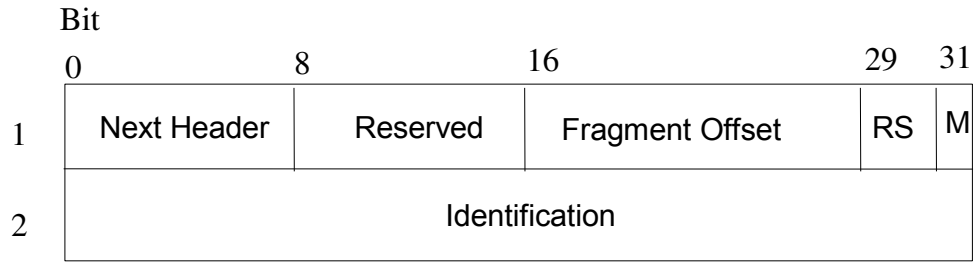
Where length is the length of the header. Each option consists of three fields: Type (8 bits), Length (8 bits) and Value (variable length).

The format of the routing header is



Routing type field specifies the type of routing information; the only one type has been defined, type 0, corresponds to loose source routing. Loose source routing means there can be intermediate nodes in between. Segment left field indicates the number of addresses that remain in the list. Address N is the true final destination address.

The format of the fragment header is



RS field is reserved. M is the more bit that indicates there are packets to come. The fragment offset and the identification fields are the same as those of IP v4.

The header identifier (i.e., Next Header) values defined are:

Header	Value in previous NH field
Hop-by-Hop Options	0
Destination Options	60
Routing	43
Fragment	44
Authentication	51
Encapsulation Security Payload	50
No Next Header	59

The IPv6 address[6] is 16 octets long and is assigned to interfaces. If the IPv6 address is expressed in dotted decimal notation, it would look very cumbersome. Thus, the address is expressed using **colon hexadecimal notation** in which the value of each 16-bit quantity is represented in hexadecimal and separated by a colon.

For example, a 16-octet number expressed in dotted decimal notation

195.104.0.140.0.0.149.10.80.98.115.38.140.0.200.12

can be expressed in terms of the colon hexadecimal notation by

C368:8C:0:950A:5062:7326:8C00:C80C

It is not necessary to write the leading zeros in a field. However, there must be at least one numeral in every field.

The colon hex notation allows zero compression in which a string of repeated zeros can be replaced by a pair of colons. For example, the IPv6 address

A371:23C8:0:0:0:A12:0:C80A

can be rewritten as

A371:23C8::A12:0:C80A

Note that zero compression can only be used once in order to avoid ambiguity.

Colon hex notation can incorporate dotted decimal suffixes. This is intended to be used during the transition from IPv4 to IPv6. The following is an example:

0:0:0:0:132.82.10.100

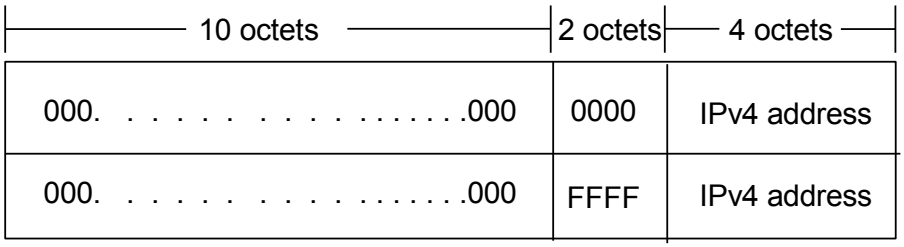
Using the zero compression, the above address can be expressed in

[6] RFC4291, "IP Version 6 Addressing Architecture," R. Hinden, S. Deering, February 2006.

::132.82.10.100

Note the dotted decimal suffix is an IPv4 address.

The IPv4 address can be encoded in an IPv6 address as following:



The 16-bit field contains 0000 if the node also has conventional IPv6 addresses and FFFF if it does not.

The IPv6 addresses have three types:

- Unicast: Specifies a single interface. A packet sent to a unicast address is delivered to the interface specified by that specific address.
- Anycast: Specifies a group of interfaces. A packet sent to an anycast address will be delivered to one of the interfaces in the group.
- Multicast: Specifies a group of interfaces. A packet sent to a multicast address will be delivered to all the interfaces in that group.

IPv6 supports CIDR. The notation is similar to that for IPv4. Thus,

(IPv6 address)/(prefix length)

Prefix length is a decimal value specifying how many of the leftmost contiguous bits of the address comprise the prefix.

The type of an IPv6 address is identified by the high-order bits of the address. They are:

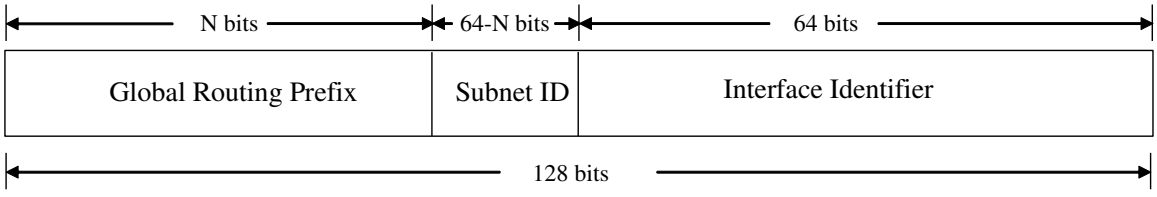
Zero Address Type	Binary Prefix	IPv6 Notation
Unspecified	00…0 (128 bits)	::/128
Loopback	00…1 (128 bits)	::1/128
Multicast	11111111	FF00::/8
Link-Local Unicast	1111111010	FE80::/10
Global Unicast	(everything else)	

The loopback address is used by a node to send an IPv6 packet to itself.

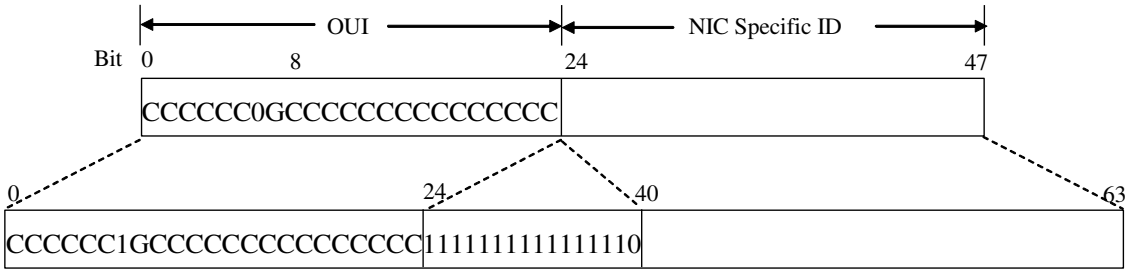
The link-local unicast addresses are used for addressing on a single link for purposes such as automatic address configuration, neighbor discovery. Routers must not forward these packets with link-local addresses to other links.

IPv6 anycast addresses are allocated from the unicast address space and are using the defined unicast address formats. A unicast address is turned into an anycast address when it is assigned to more than one interface. The devices to which the anycast address is assigned must be explicitly configured to know that its IP address is an anycast address.

The following figure illustrates the IPv6 unicast address structure. Each address consists of three parts: the global routing prefix, the subnet identifier, and the interface identifier. The interface identifier identifies a particular interface and is 64 bits long. It contains the hardware address, such as Ethernet MAC address. The global routing prefix is used to route the packets, and the subnet identifier is used to identify the subnetwork.



The interface identifier is 64 bits long. The 48 bits Ethernet MAC address needs to be mapped into this field[7]. The mapping is shown in the following figure.



Note that the C bits in the OUI portion of the MAC address are used to identify the company. The 6th bit in the first byte of the MAC address, which is used to indicate whether the address has global scope, is changed from to 0 to 1.

To simplify the configuration of the host, IPv6 supports **autoconfiguration**. It is also called serverless autofiguration or stateless autoconfiguration. The process is as follows:

1. To begin autoconfiguration, the host generates a link-local IPv6 address by combining the 10-bit link-local prefix (1111 1110 10) with 54 zero bits and its 64-bit interface identifier.
2. The host sends a *route solicitation* to the network it is connected to request additional information from a router in the network.
3. The router responds by sending a *router advertisement* to inform the host about the prefix that can be used for global address.
4. Once the host receives the *router advertisement*, it makes the sender as its default router, and uses the prefix received as its address.

7 M. Crawford, "Transmission of IPv6 Packets over Ethernet Networks," RFC 2462, Decd. 1998.

TRANSMISSION CONTROL PROTOCOL (TCP)

Services provided by TCP to the upper layer are:

- Multiplexing Service

- Connection Management Service
  -- Connection Establishment
  -- Connection Maintenance
  -- Connection Termination

- Data Transport Service
  -- Full Duplex
  -- Timely
  -- Ordered
  -- Flow Control
  -- Error Check/ Error Correction
  -- Security

- Error Reporting

TCP SERVICE PRIMITIVES

TCP service primitives are:

- OPEN
  - Passive Open
  - Active Open
  - Open Parameters
    - Addressing
    - Time out
    - Security
    - Quality of Service

- DATA TRANSFER
  - Send
    - Urgent
    - Push
    - Time out
  - Allocate (receive)

- TERMINATE
  - Close
  - Abort

- STATUS

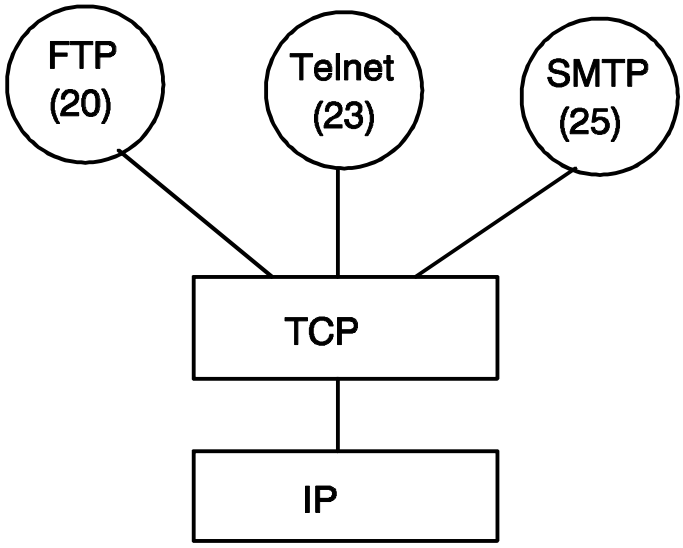
PORT

Both TCP and UDP may provide services to more than one application processes. They use port number as the ultimate destination within a computer to identify the application process.

The well known ports are:

FTP Data (20)
FTP Control (21)
Telnet (23)
SMTP (25)
Domain Name Server (53)
Authentication Service (113)

An example of several ports using one TCP connection is as follows:

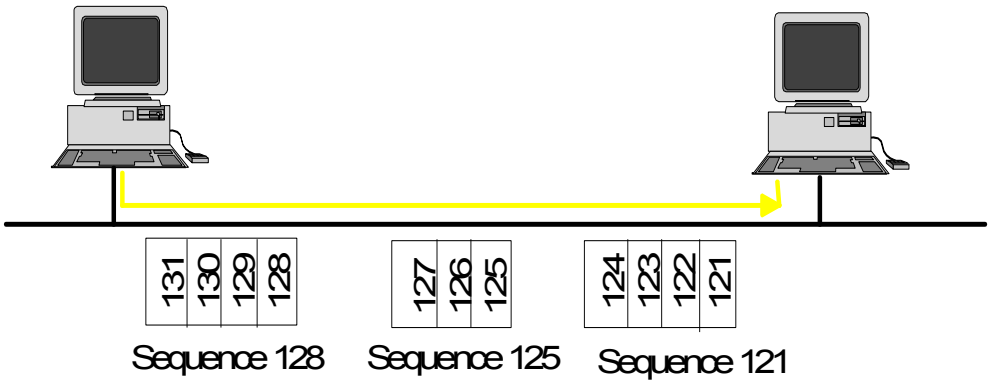


SEQUENCE NUMBER

In TCP and UDP, a data stream is treated as a sequence of octets that are grouped into segments. Each octet of data in the data stream is assigned a unique sequence number. The sequence number of the very first octet is assigned when the connection is established.

The sequence number included in the packet is the sequence number assigned to the first octet of data that is placed in the data field of the packet.

In the following example, three packets are sent from one PC to another. The sequence number for each packet is shown.



PUSH

To make the transmission more efficient, TCP usually collects octets from the data stream in the buffer to make a long segment before it transmit the segment.

The application process can require TCP to transmit all outstanding data up to and include that labeled with PUSH flag. The destination TCP notices the PUSH flag, it immediately passes the data up to the destination application process.

URGENT

URGENT flag indicates that the data packet is urgent and needs to be processed immediately.

TRANSMISSION CONTROL BLOCK (TCB)

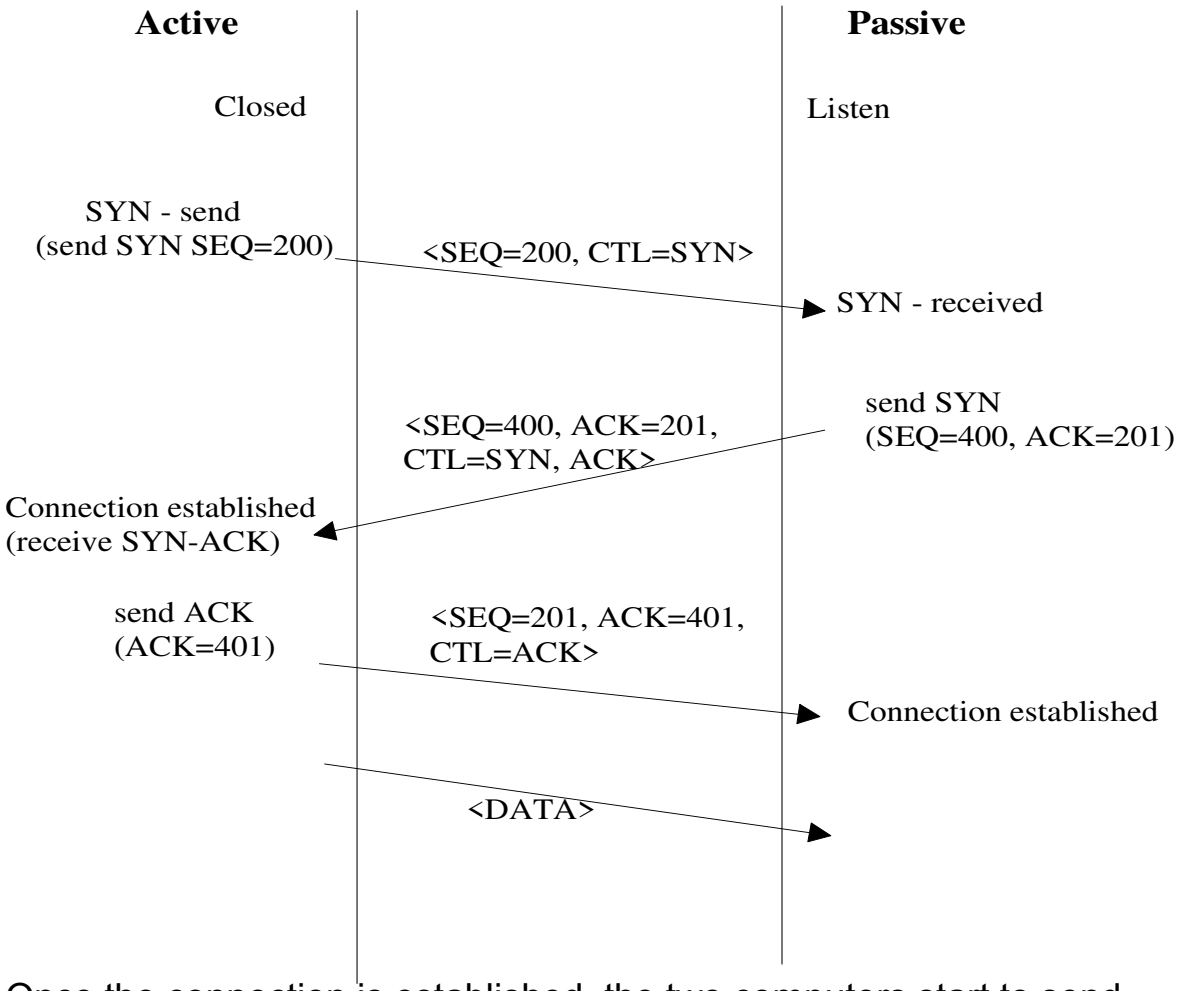
Both the active and passive ends of the connection must create a data structure known as the Transmission Control Block (TCB).

The TCB contains:

The type of connection (active or passive)
The local and remote port numbers
The local and remote Internet addresses
The state of the connection
A pointer to the input buffer that contains the initial SYN packet (for passive end only)

CONNECTION ESTABLISHMENT

A connection between two computers running TCP is established by successfully exchange SYN packets as following:



Once the connection is established, the two computers start to send and receive data packets.

FLOW CONTROL USING VARIABLE SIZED WINDOW

The TCP packet header includes a window field. It indicates the range of sequence numbers that it is willing to receive. When the receiver buffer starts to fill up, the window starts to decrease. At the time, the transmitter will send smaller data segments. After the data in the receiver buffer have been sent to the application process, the receiver window size can be increased. The sender will then send larger data segments.

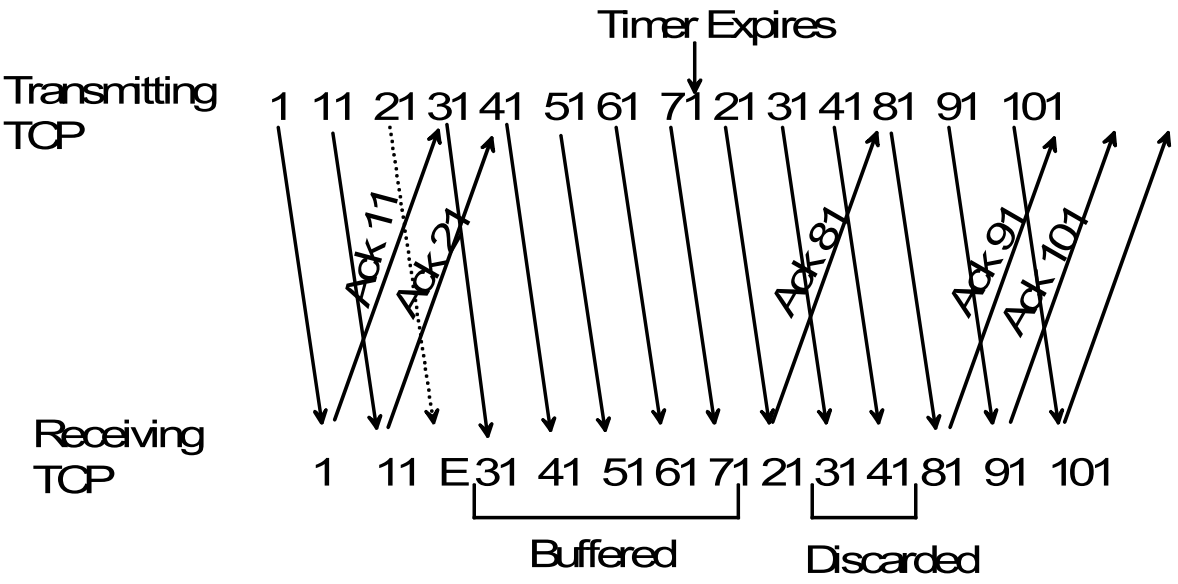
A TCP process with zero send window must periodically send probe packets with invalid sequence and acknowledgment number and a single byte of invalid data. The receiving TCP process then responds by sending an ACK with a window size possibly greater than zero.

ERROR DETECTION AND CORRECTION

TCP uses checksum for error detection.

TCP uses selective repeat procedure for error correction. It utilizes positive acknowledgment and time-out for retransmission of errored packets.

In the following example, each packet is 10 octets long. Sequence number 2 packet is received with error.



TIME-OUT VALUE

TCP uses "SMOOTHED ROUND-TRIP DELAY" to define the time-out value for retransmission.

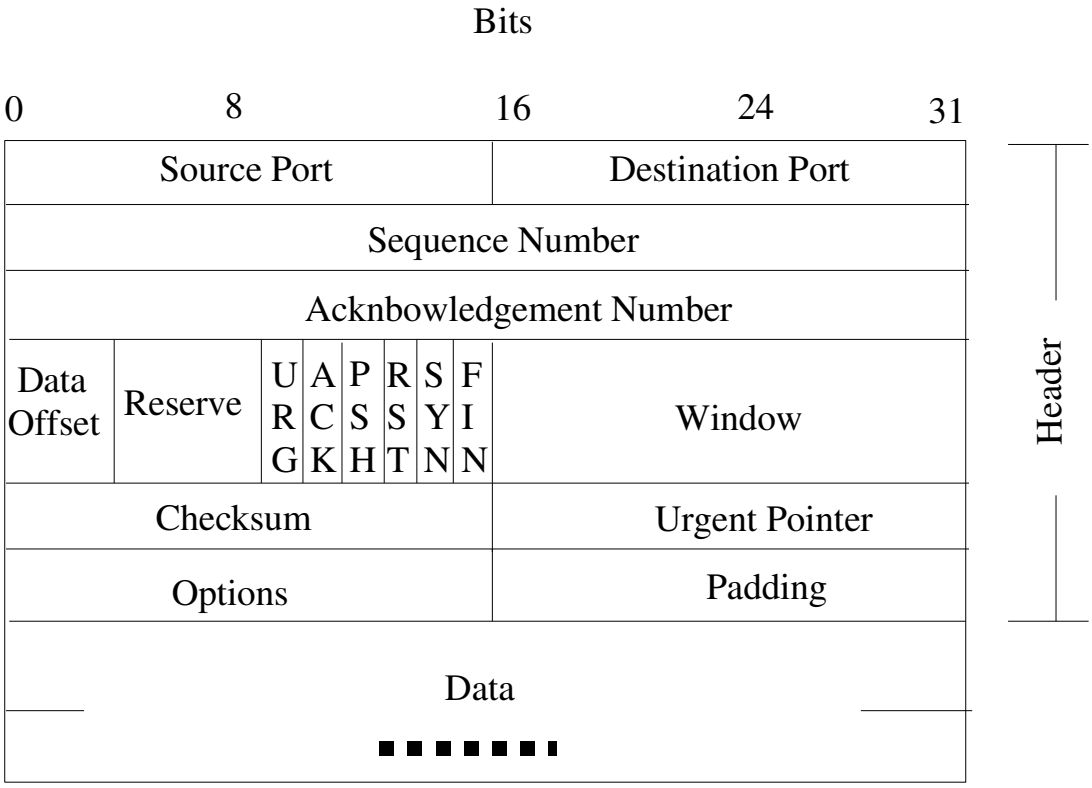
Due to the complexity of Internet, a TCP packet may travel through several different networks. It may experience very different delay from connection to connection.

The retransmission timer is set dynamically based on smoothed round-trip delay. When a data packet is sent, the sending TCP records the time of the transmission and the sequence number in the TCB. When the sending TCP receives an acknowledgment for that sequence number, it calculates the round-trip delay. TCP keeps the average round trip delay as a weighted delay and sets the retransmission timer accordingly.

We will have more discussions on time-out values latter when we discuss flow control mechanisms.

TCP FORMAT

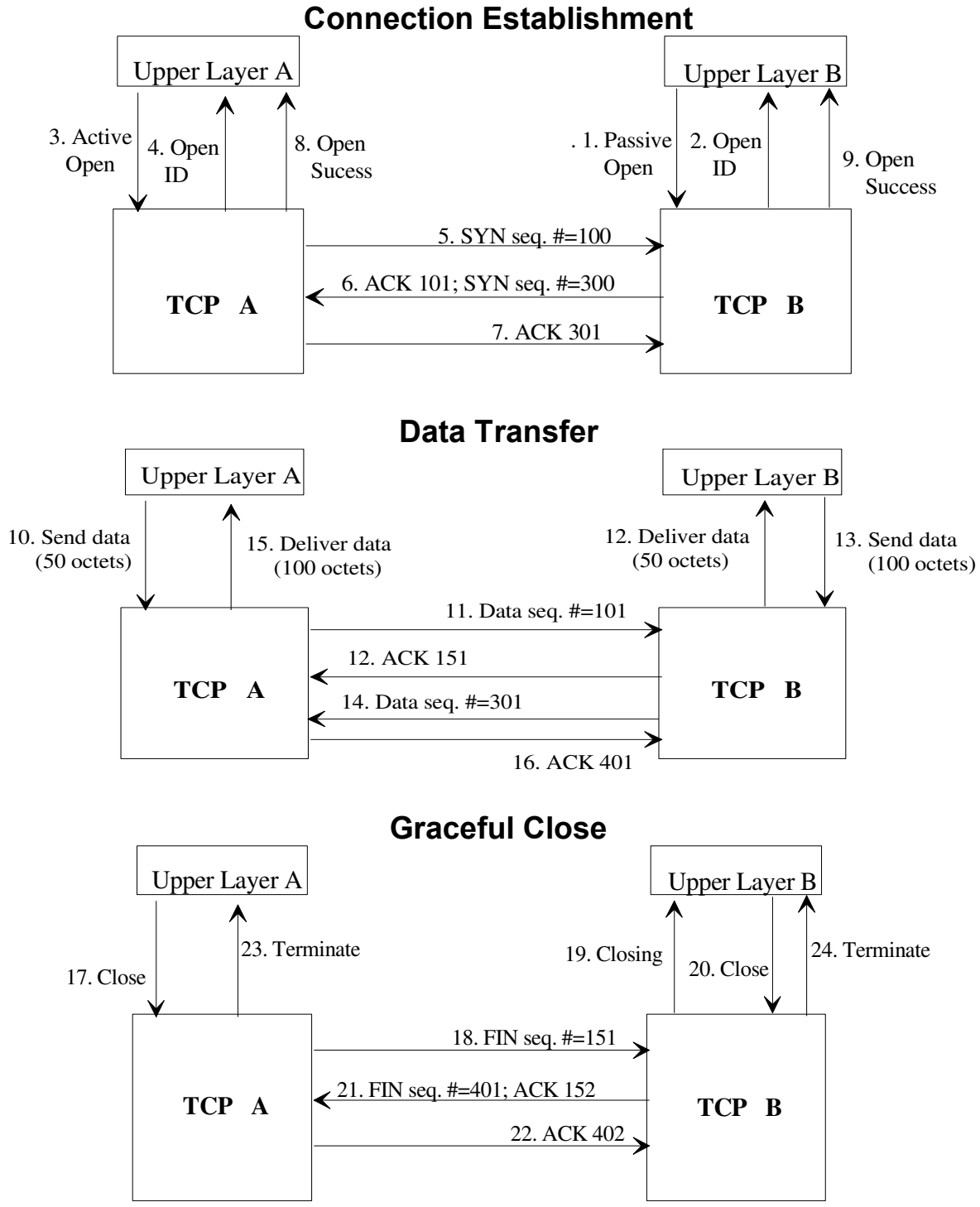
The TCP header format is as follows:



TCP FORMAT

Source Port	Identifies the application at the source computer
Destination Port	Identifies the application at the destination
Sequence Number	The sequence number of the first data octet in the segment
Acknowledgment	Applies only if the ACK field is set to 1 acknowledges the reception of data up to (ack. number - 1)
Data Offset	Specifies the number of 32 bit words in the TCP header
Reserved	
Control Bits	Controls the handshaking and data transfer URG - Urgent ACK - Acknowledgment PSH - Push flag RST - Reset the connection SYN - Synchronize the sequence number FIN - No more data from sender
Window	The number of octets beginning with the one in the acknowledgment field, that can be accepted
Checksum	Checksum for the whole TCP packet (TCP header + data) and the pseudo header (source and destination IP addresses, protocol and total length)
Urgent Pointer	Only significant if URG is set. The value is a offset from the sequence number, and indicates the end of the urgent data
Options	
Padding	

AN EXAMPLE OF TCP OPERATION



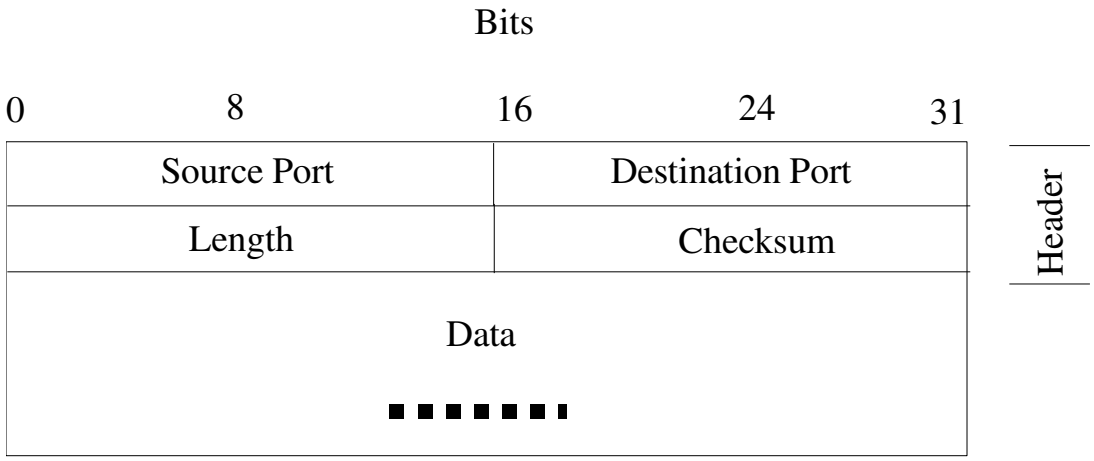
USER DATAGRAM PROTOCOL (UDP)

- Simple connectionless datagram service

  -- Dependent on IP
  -- Low overhead
  -- No loss, mis-ordered, or duplication protection
  -- Checksum optional

UDP is used by applications that
(1) do not need error correction, such as voice telephony,
(2) send short messages, so there is no need for connection establishment. Examples are SNMP, DNS, etc.

USER DATAGRAM PROTOCOL (UDP)



Source Port	(optional) Indicates the port of the sending processor, as well as the port to which a reply should be addressed.
Destination Port	Indicates the port of the receiving processor.
Length	The combined length of the UDP header and data, expressed in number of octets.
Checksum	(optional)

REAL-TIME TRANSPORT PROTOCOL (RTP)
RTP CONTROL PROTOCOL (RTCP)
REAL TIME STREAMING PROTOCOL (RTSP)[8]

Many real-time applications, such as voice over IP, video conferencing, etc. require that the IP packets are delivered in a timely fashion and errored packets are not retransmitted. Some real-time applications require multicasting of packets to many different destinations. In addition, most of these applications require timestamping of the data, so that applications at the destination hosts can play out the data accordingly.

Most of these real-time applications use UDP as the transport layer protocol. UDP does not require point-to-point connection set up, thus it is more suitable for multicasting environment. In addition, UDP does not perform retransmission of errored IP datagrams. However, both IP and UDP do not provide timing information. This is one of the main functions of Real Time Protocol.

Real time protocol consists of two protocols: (1) Real-time Transport Protocol (RTP) to carry data that has real-time properties, and (2) RTP Control Protocol (RTCP) to monitor the quality of service and to convey information about participants in an on-going session. The port number in the header of UDP identifies RTP and RTCP.

RTP is responsible for the transfer of data. It is the responsibility of the application at the originating source to perform coding of the data, such as compression of voice and video. It is up to the application at the receiving host to play out the data (e.g., audio, video), to detect any packet loss and to determine how handle this situation, to detect any delayed packets and to decide how to handle them.

---

[8] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.
H. Schulzrinne, A. Rao and R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998.

The following sections describe some aspects of the use of RTP. The examples were taken from RFC1889 to illustrate the basic operation of applications using RTP, not to limit what RTP may be used for. In these examples, RTP is carried on top of IP and UDP.

Scenario 1: Simple Multicast Audio Conference

An IETF working group meets to discuss the latest protocol draft, using the IP multicast services of the Internet for voice communications. The working group chair, through some allocation mechanism, obtains a multicast group address (class D IP address) and pair of ports. One port is used for audio data, and the other is used for control (RTCP) packets. This address and port information is distributed to the intended participants.

The audio conferencing application used by each conference participant sends audio data in small chunks of say, 20 ms duration. An RTP header precedes each chunk of audio data. This header and the voice data are in turn contained in a UDP packet. The RTP header indicates what type of audio encoding (such as PCM, ADPCM or LPC) is contained in each packet, so that senders can change the encoding during a conference. This allows accommodation of new participant whom may have different capability or reaction to network congestion.

To cope with impairments, such as packet loss, packet out-of-order, and delay, the RTP header contains timing information and a sequence number. These two fields allow the receivers to reconstruct the timing produced by the source. This timing reconstruction is performed separately for each source of RTP packets in the conference. The sequence number can also be used by the receiver to estimate how many packets are being lost.

Since members of the working group join and leave during the conference, it is useful to know who is participating at any moment and how well they are receiving the audio data. For that purpose, each instances of the audio application in the conference periodically multicasts a reception report plus the name of its user on the RTCP (control) port. The reception report indicates how well the current speaker is being received and may be used to control adaptive

encoding. In addition to the user name, other identifying information may also be included subject to control bandwidth limits. A site sends the RTCP BYE packet when it leaves the conference.

Scenario 2: Audio and Video Conference

If both audio and video media are used in a conference, they are transmitted as separate RTP sessions. RTCP packets are transmitted for each medium using two different UDP port pairs and/or multicast addresses. There is no direct coupling at the RTP level between the audio and video sessions, except that a user participating in both sessions should use the same distinguished (canonical) name in the RTCP packets for both, so that the sessions can be associated.

One motivation for this separation is to allow some participants in the conference to receive only one medium if they choose. Despite the separation, synchronized playback of a source's audio and video can be achieved using timing information carried in the RTCP packets for both sessions.

Scenario 3: Mixers and Translators

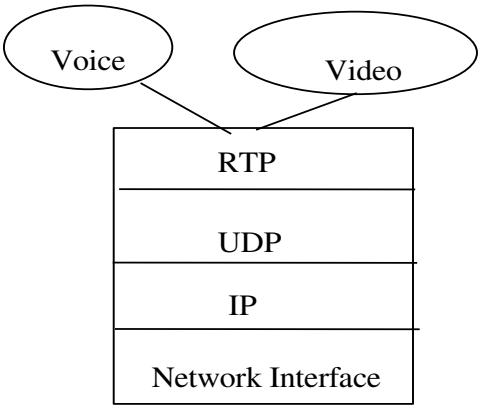
Now, consider the case where participants in one area are connected through a low-speed link to the majority of the conference participants who enjoy high-speed network access. Instead of forcing everyone to use a lower-bandwidth, reduced-quality audio encoding, an RTP-level relay called a **mixer** may be placed near the low-bandwidth area. This mixer resynchronizes incoming audio packets to reconstruct the constant 20 ms spacing generated by the sender, mixes these reconstructed audio streams into a single stream, translates the audio encoding to a lower-bandwidth one and forwards the lower-bandwidth packet stream across the low-speed link. These packets might be unicast to a single recipient or multicast on a different address to multiple recipients. The RTP header includes a means for mixers to identify the sources that contributed to a mixed packet so that correct talker indication can be provided at the receivers.

Some of the intended participants in the audio conference may be connected with high bandwidth links but might not be directly reachable

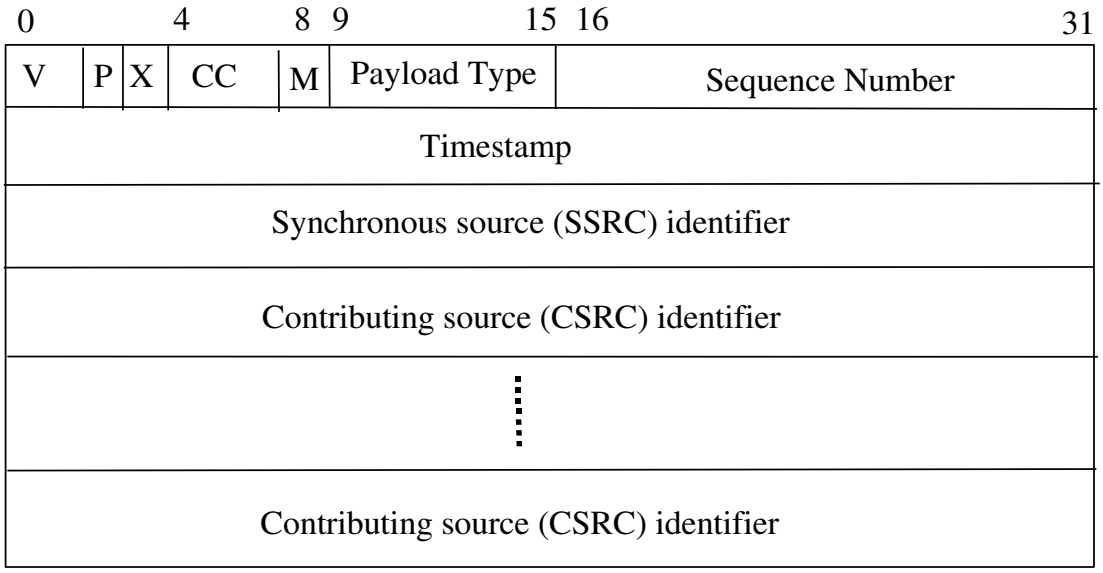
via IP multicast. For example, they might be behind an application-level firewall that will not let any IP packets pass. For these sites, mixing may not be necessary, in which case another type of RTP-level relay called a **translator** may be used. Two translators are installed, one on either side of the firewall, with the outside one funneling all multicast packets received through a secured connection to the translator inside the firewall. The translator inside the firewall sends them again as multicast packets to a multicast group restricted to the site's internal network.

Mixers and translators may be designed for a variety of purposes. An example is a video mixer that scales the images of individual people in separate video streams and composites them into one video stream to simulate a group scene. Other examples of translation include the connection of a group of hosts speaking only IP/UDP to a group of hosts that understand only ST-II, or the packet-by-packet encoding translation of video streams from individual sources without resynchronization or mixing.

The protocol stacks for these real-time applications are:



The header format of the RTP for data transfer is as follows:



Where:

V (2 bit): version number.

P (1 bit): padding. If this bit is set, the packet contains one or more additional padding octets at the end of the payload. The last octet of the padding contains a count of how many padding octets should be ignored.

X (1 bit): extension bit. If set, the fixed header is followed by exactly one extension header.

CC (4 bits): CSRC count. The number of CSRC identifiers in the header.

M (1 bit): marker. The interpretation of the marker is defined by a profile. It is intended to allow significant events, such as frame boundaries to be marked in the packet stream. For example, it is set to mark the end of a video frame, or the beginning of a talk spurt.

Payload Type (7 bits): identifies the media type of the payload and the format of the data, such as compression, encryption.

Sequence Number (16 bits): For the receiver to reorder the packets received and to identify any packet loss. Note that some long data stream (e.g., a video frame) maybe splitted into several packets. These packets all have the same timestamp. Thus, sequence number is needed.

Timestamp (32 bits): timestamp of the first octet of data in the payload.

Synchronous source identifier (32 bits): unique identifier of the synchronization source. This identifier is randomly chosen.

Contributing source identifier (32 bits): Identifiers of the contributing sources for the payload contained in this packet. The number of identifiers is given in the CC field.

RTCP:

RTP control protocol (RTCP) is used to provide feedback to the RTP data source and the session participants. During the session, each participant periodically issues an RTCP packet to all other session participants.

The data transport protocol RTP is augmented by the real-time control protocol, RTCP. RTCP performs the following functions:

Quality of service and congestion control: The primary function of RTCP is to provide feedback on the quality of the data distribution. Every control packet contains reception statistics in the form of receiver reports. Every receiver for every source issues receiver reports. They report the number and fraction of lost packets and the inter-arrival jitter as well as to which packets these statistics refer. The applications may use this feedback to adapt to different network conditions, e.g. by using adaptive encoding. Feedback about transmission quality is also useful to locate problems and diagnose faults.

Identification: Another function of RTCP is to keep track of participants when their internal identifiers change, and to distribute information about all participants in a session, such as names and email addresses.

Session size estimation and scaling: Control packets are periodically transmitted to all participants in the session. The rate of transmission of such packets must be varied according to the number of participants. RFC1889 includes an algorithm by which each participant limits its RTCP packet rate on the basis of the total session population.

Session control: RTCP optionally provides minimal session control information, for example, participant identification to be displayed in the user interface. This is most likely to be useful in "loosely controlled" sessions where participants enter and leave without membership control or parameter negotiation. RTCP serves as a convenient channel to reach all the participants, but it is not necessarily expected to support all the control communication requirements of an application. A higher-level session control protocol may be needed.

RTCP consists of the following different RTCP packets:

SR: Sender report, for transmission and reception of statistics from participants that are active senders.

RR: Receiver report, for reception statistics from participants that are not active senders.

SDES: Source description items, such as canonical name, real user name, e-mail address, etc.

BYE: Indicates end of participation.

APP: Application specific functions.

Real Time Streaming Protocol (RTSP)

The Real Time Streaming Protocol (RTSP) is used to establish and control either a single or several time-synchronized streams of continuous media. Examples are voice and video. The combination of media streams presented to the client is called a presentation.

RTSP supports the following operations:

1. Retrieval of media from media server:
   The client can request from the server the information about the media streams, called presentation description, via hypertext transport protocol (HTTP) or some other method. If these media streams are being multicast, the presentation description contains the multicast addresses and ports to be used for the continuous media. If the presentation is to be sent only to the client via unicast, the client provides the destination for security reasons.

2. Invitation of a media server to a conference:
   A media server can be "invited" to join an existing conference, either to play back media into the presentation or to record all or a subset of the media in a presentation. This mode is useful for distributed teaching applications. Several parties in the conference may take turns "pushing the remote control buttons."

3. Addition of media to an existing presentation:
   Particularly for live presentations, it is useful if the server can tell the client about additional media becoming available.

RTSP acts as a “network remote control” for multimedia servers.

RTSP is a text-based protocol. Each line is terminated by CRLF. The messages can be carried over either TCP or UDP. In general a message consists of message type, message headers and message body. The message types can be request or response.

Format of a request message is

```
Request    =    Request-Line
           *(   General-header
           |    Request header
           |    Entity header )
                CRLF
                [message body]
```

Request-Line = Method SP Request-URI SP RTSP-Version CRLF

Examples of methods are: DESCRIBE, ANNOUNCE, PAUSE, PLAY, RECORD, SETUP, TEARDOWN.

Request header provides information about the request. It indicates information, such as the language supported, the coding scheme acceptable, etc.

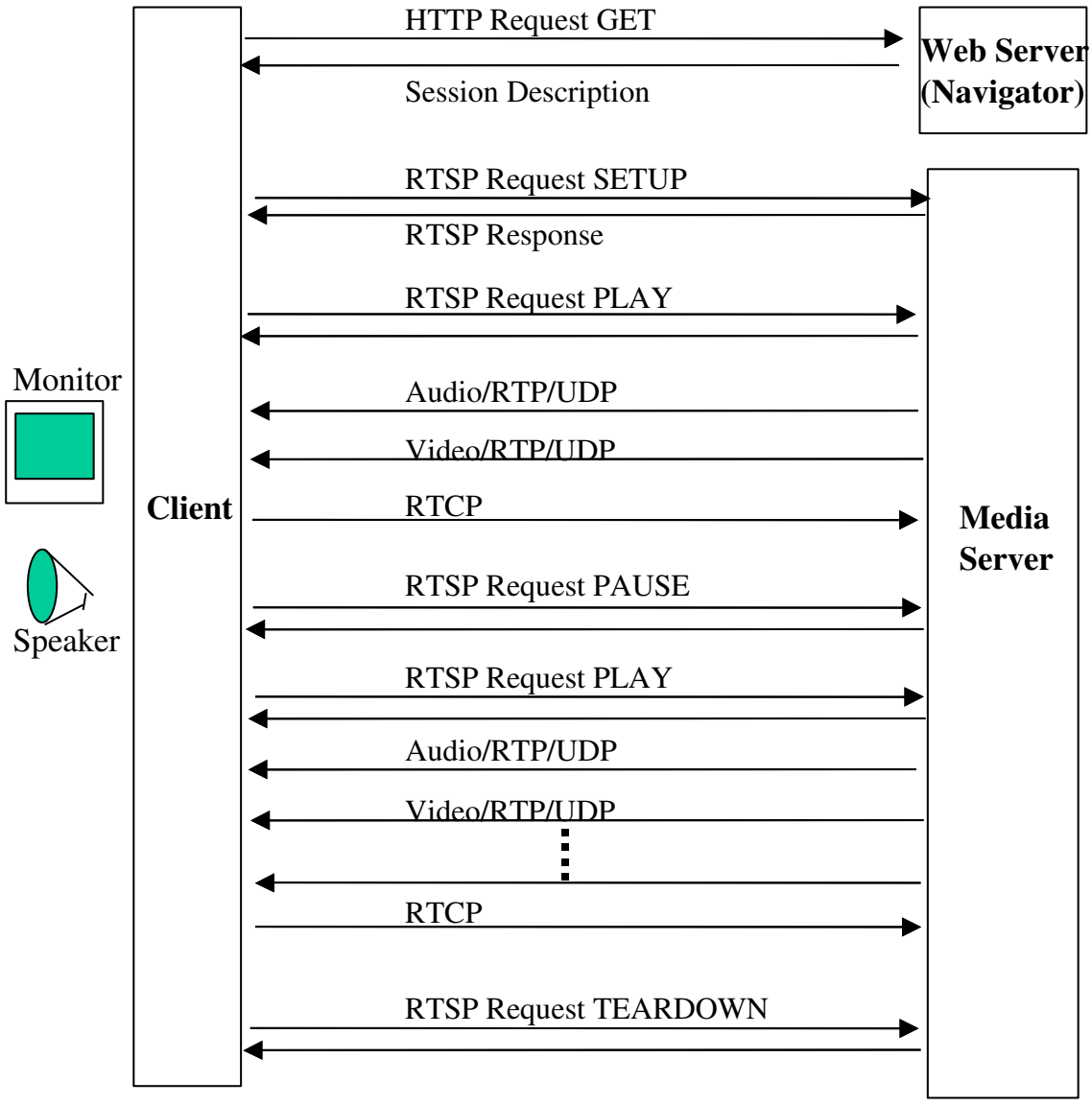
An example of SETUP request is

```
C -> S: SETUP rtsp://example.com/foo/bar/baz.rm RTSP/1.0
        CSeq: 302
        Transport: RTP/AVP; unicast;client_port=4588-4589

S-> C:  RTSP/1.0 200 OK
        CSeq: 302
        Date: 23 Jan 1997 15:35:06 GMT
        Session: 47112344
        Transport: RTP/AVP;unicast;client_port=4588-4589;server_port=6256_6257
```

Examples of PLAY requests are:

```
C->S:   PLAY rtsp://example.com/audio RTSP/1.0
        CSeq: 835
        Session: 12345678
        Range: npt=10-15

C->S:   PLAY rtsp://example.com/audio RTSP/1.0
        CSeq: 836
        Session: 12345678
        Range: npt=20-
```

Put these three, RTP, RTCP and RTSP, together, we have the following scenario.

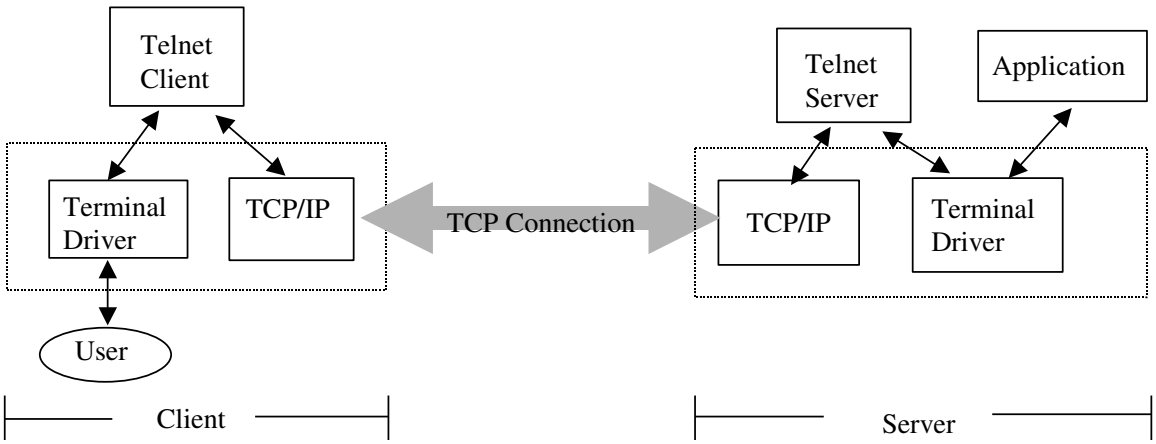


TELNET

Telnet[9] (terminal networking) is a standard application that almost every TCP/IP implementation supports. It provides emulation of various types of terminals, so that they can communicate with each other.

Many computers are running under different environment. RFC854 defines the lowest common denominator terminal, called network virtual terminal (NVT). The NVT is an imaginary device that both the client and the server map their real terminal to. In another word, both the client and server operating systems must map their terminals to NVT. Because both the client and the server are “on NVT” they can communicate.

The imaginary NVT is a character device with a keyboard (input) and printer (output). Data typed by the user at the client on the keyboard is sent to the server through a TCP connection, and data received from the server is output to the printer (i.e., the monitor).



The client operating system provides the terminal driver and the TCP/IP protocol. The user invokes and input to the Telnet client via the terminal driver. The client Telnet then requests a TCP connection to the server.

9 J. Postel, J.K. Reynolds, “Telnet Protocol Specification,” RFC 854, May 1983.

The Telnet server deals with a terminal driver (or pseudo-terminal driver in UNIX case). To the application program (e.g., login shell) at the server, it is invoked by the terminal driver.

The NVT protocol models an old-fashion half duplex keyboard and printer operating in line-at-a-time mode. RFC854 defines the following characteristics for the NVT:

- NVT data bytes use seven-bit USA ASCII character set. The initial bit is set to zero.
- Data bytes are sent a line at a time.
- Each line ends with an ASCII carriage return and line feed.
- Bytes with the high order bit set to 1 are command codes. For example code 236 (decimal) is end-of-file, 243 (decimal) is break, and 249 for go-ahead.
- It is a half duplex operation. After transmitting the data, the client waits for the response from the server. The server sends its response and then sends a go-ahead command to indicate that the client can send another line of data.
- Telnet uses in-band signaling in both directions. The byte 255 decimal is called “interpreted as command” (IAC). The next byte is the command byte.

The following is an example of using Telnet for logging into a remote server, server.xyx.com.

> **telnet server.xyz.com**
> Trying 152.132.50.123
> Connected to server.xyz.com.
> Escape character is ‘^]’.
>
> SunOS UNIX (server.xyz.com)
>
> Login: **david**
> Password:
> Last login: Fri Sep 3 11:31:20
> TERM=vt100, PRINTER=1p

Where the bold-faced characters are user typed in.

FILE TRANSFER PROTOCOL (FTP)

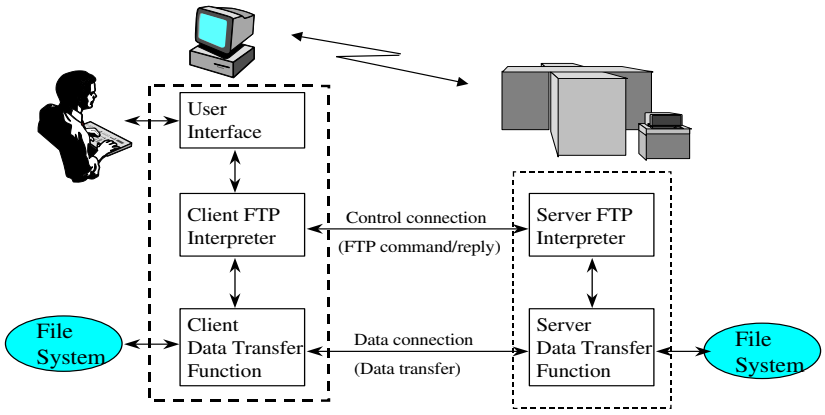
File Transfer Protocol (FTP)[10] allows users to login, provide user's name and password, list directory, and send and receive files to/from a remote server. While users can exchange files interactively, FTP is designed primarily for program to program data transfer. Data can be in the form of text file, encoded data or program. It uses the services of TCP.

FTP handles the following issues for file transfer:

- Data Representation
  - Text file
  - Non-text file
  - Format control
- File Structures
  - File structure
  - Record structure
  - Page structure
- Transmission Modes
  - Stream mode
  - Block mode
  - Compresses

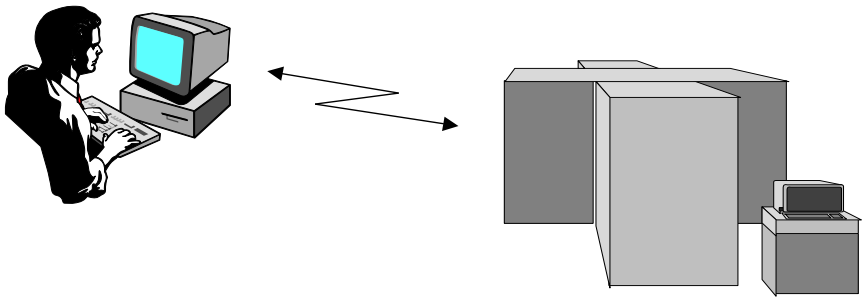
During the ftp process, actually, two TCP connections are established: one for FTP command and reply, the other for data transfer. The command and reply connection uses port number 20 and the other port number 21. The user uses the control connection to login and to request for file transfer. The data connection is then established to transfer data from the server to the user.

10 J. Postel, J.K. Reynolds, "File Transfer Protocol," RFC 959, Oct. 1985.



An example of interaction between a host and an ftp server:

The following is an example of a user (david) accessing a remote ftp server (ftp.xyzuniv.edu) to retrieve a binary file, david-fl.z, at the directory files/myfile.



Sun% **ftp ftp.xyzuniv.edu**
connected to ftp.xyzuniv.edu
ftp.xyz.edu server ready

Name: **david**
Password: *******

Guest login OK.
ftp> **cd files/myfile**
CWD command successful.

ftp> **binary**
Type set to I

```
ftp>  get david-fl.z
PORT command successful.
Opening BINARY mode data connection for david-fl.z
Transfer complete.

ftp>  quit
Goodbye

Sun %
```

Note: Bold characters are typed in by the user.

DOMAIN NAME SYSTEM

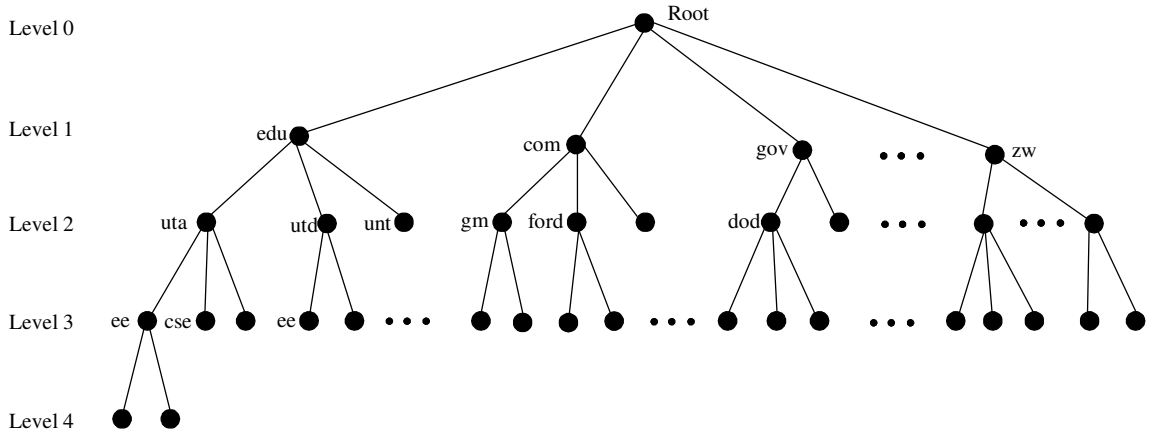
In the IP networks, each host is assigned an IP address. The routers use the destination IP addresses to route the packets. However, it is not easy to remember the IP addresses. Users remember and prefer to use the names of the hosts (or servers). Mapping between the host (or server) names and the IP addresses is the function of the domain name system (DNS)[11].

With the rapid growth of hosts and servers, the number of names of the hosts and servers has increased dramatically. Currently, there are millions of devices connected to the Internet. Each one requires a name. With this huge number of names, a flat naming system can not used. A hierarchical namespace is used. In addition, a decentralized naming mechanism with delegated authority for parts of the namespace and distributed responsibility of mapping between names and IP address is used.

In domain name system (DNS), a hierarchical naming scheme called domain name is used. The hierarchy uses an inverse tree structure with the root at the top. The tree can have up to 128 levels. Each node of the tree has a label, which is a string with a maximum of 63 characters. The root label is a null string. Children of a node must have different labels in order to guarantee uniqueness of the domain names. Each node of the tree has a domain name. A full domain name is a sequence of labels separated by periods (.).

The following figure illustrates the domain name space:

11 P.V. Mockapetris, "Domain names - implementation and specification," RFC 1035, Nov.1987.



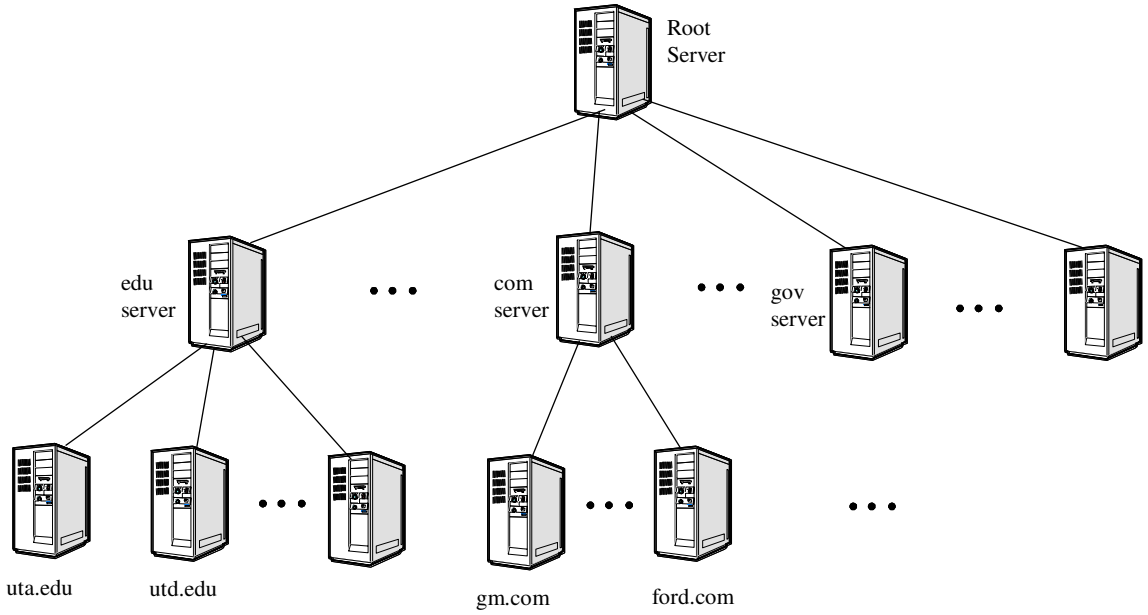
The above figure shows some domain names: ee.uta.edu; uta.edu; ee.utd.edu; utd.edu; gm.com; ford.com, etc.

A domain is a subtree of the domain name space. For example, the domain of **uta.edu** is the subtree under the node of uta. The domain of **edu** is the subtree under the node of edu.

The top level domain names are:

Domain Name	**Meaning**
aero	Air transport industry
arpa	Infrastructure domain
biz	Business
com	Commercial organizations
coop	Cooperative associations
edu	Educational institutes
gov	US government
info	Information
int	International organizations
mil	US military
museum	Museums
name	Individuals
net	Network service providers
org	Organizations
pro	Professional
Country code	Countries

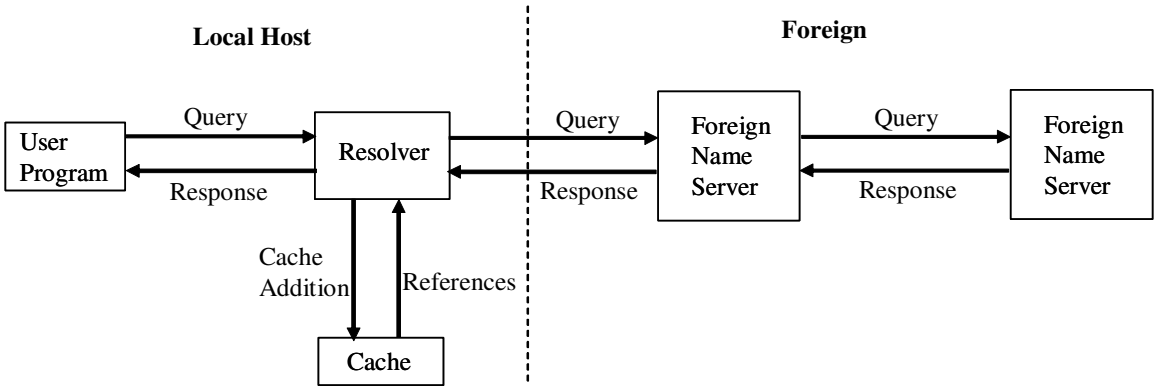
The domain name space information is stored in servers in a distributed, hierarchical fashion, as shown in the following. Each server must know the IP address of its parent server.



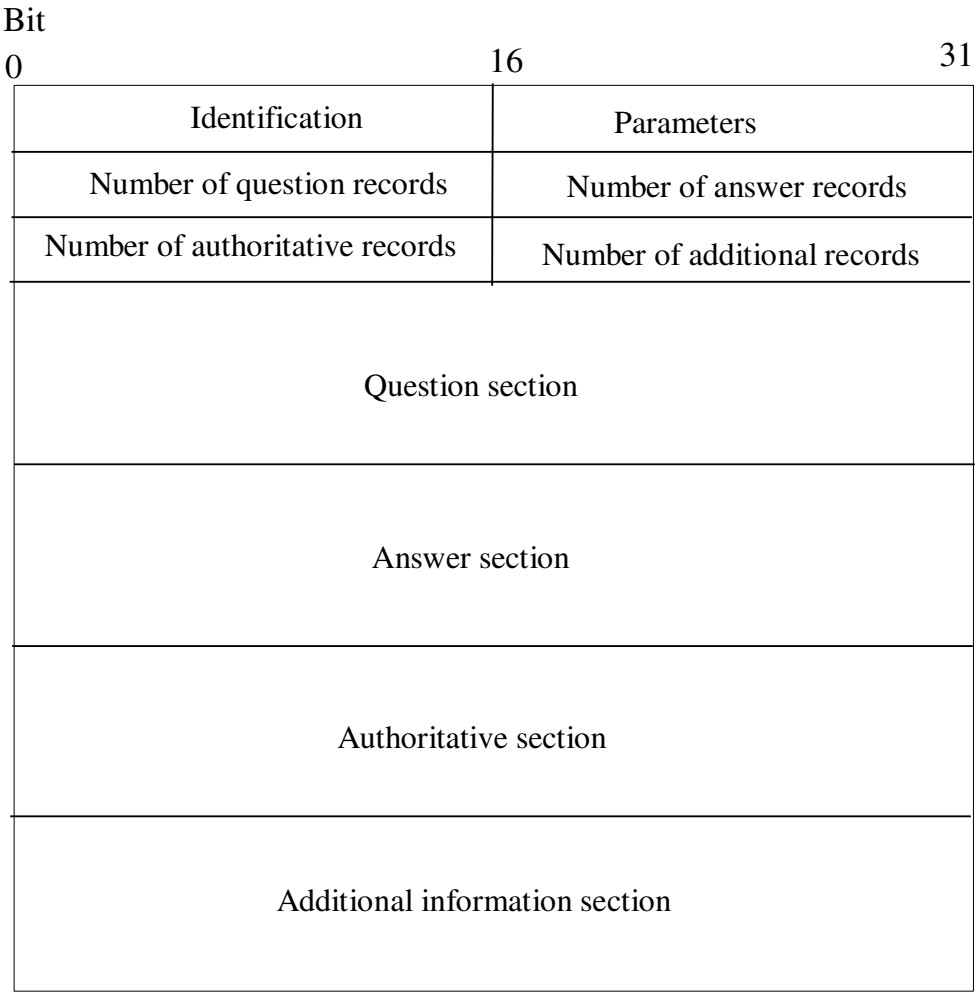
A server is responsible for a zone. A zone is a subdivision of a domain. If a domain is not subdivided, then a zone is equal to a domain. If a domain is subdivided into zones, the domain server can delegate parts of its authority to other servers.

Operation of DNS

A user program in a host can request a resolver to formulate a query for the mapping of a domain name to the Internet address. If the resolver at the local host can not find the Internet address, it then queries another domain name server. This foreign domain name server's IP address must be stored in the local host. If the foreign domain name server can not find the IP address of the domain name requested, it then query another domain name server until the mapping is complete. Once the mapping is completed, the server sends the response to the querier. The local host, after receiving the answer, can cache the IP address resolved for future use.



Domain Name System uses the service of UDP or TCP, with the port number of 53. The format of the DNS messages is as following:



The Identification field contains a value that the querier can uniquely match the response. The Parameter field specifies the type of message, the type of answer, etc. The Number of Question Records contains the number of queries in the

question section. The Number of Answer Records contains the number of answers in the answer section. The value of this field is zero in the query message. The Number of Authoritative Records contains the number of authoritative records in the authoritative section. Its value is zero in the query message. The Number of Additional Records contains the number of additional records in the additional information section. Its value is zero in the query message.

The subfields of the parameter field are:

QR: =0 if query; =1 if response
OpCode: define the type of the message.
=0 for standard query, =1 for inverse query
=2 for server status request; other values reserved
AA (authoritative answer): =1 if server is authoritative server.
Used in response message only.
TC (truncated): =1 if the response is more than 512 bytes and truncated to 512 bytes.
RD (recursion desired): =1 if the client desires a recursive answer.
RA (recursion available): =1 if recursive response is available.
It is used in response only.
rCode: Error codes

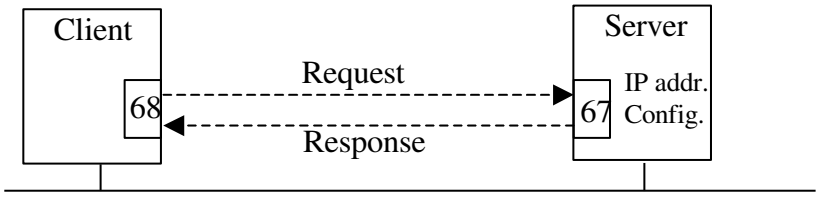
The Question Section contains the questions. The Answer Section contains the answers. The Authoritative Section contains the information about authoritative server for the query. The Additional Information Section contains additional information to help the resolver.

DHCP AND BOOTP

To solve the issue of IP address shortage, many companies and networks are assigning temporary IP addresses to active users from an IP address pool. The Dynamic Host Configuration Protocol (DHCP) provides a mechanism through which host computers using IP can obtain IP addresses when they are booted. In addition to assigning temporary IP address, DHCP is used to provide protocol configuration parameters automatically to the host computers through the network. The protocol parameters include subnet mask, address of default routers, address of the domain name server, IP address lease time, etc.

DHCP has been evolved from its predecessor, BOOTP, Bootstrap Protocol. BOOTP was originally designed to provide clients, such as diskless computers, with configuration parameters. After boot, the host computer, using BOOTP, broadcasts a request to the server to request for the configuration parameters. The server responds with a set of parameters to the client.

Both BOOTP and DHCP are based on a client-server paradigm, in which the client contacts a BOOTP/DHCP server and requests for the configuration parameters. The server is typically centrally located and operated by the network administrator. Because the server is run by a network administrator, clients can be reliably and dynamically configured with parameters appropriate to the current network architecture. Both BOOTP and DHCP use UDP as the transport protocol. The client has the port number 68 and the server the port number 67.



While DHCP is popular and has many enhancements over BOOTP, BOOTP is still being used by many legacy clients and servers. We will discuss BOOTP first.

BOOTP[12,13]

BOOTP is used by hosts (Clients) that require configuration parameters, including IP addresses, at the bootstrap. Another protocol, RARP, is also used for the diskless hosts to obtain the IP addresses. However, there are two problems with RARP: (1) It only returns IP address, not other configuration parameters, and (2) RARP relies on link layer broadcast, its requests are not forwarded by routers, thus every network will need a RARP server. BOOTP solves these two shortcomings.

In BOOTP, the server administrator has to manually input the hardware address and the corresponding IP address into a configuration database that BOOTP uses to look up the IP address. An example of the database is shown in the following.

Hardware Type	Hardware Address	IP Address	Other Parameters
1	02 61 8C 23 15 AC	129.32.19.110	
1	02 61 A2 38 B7 17	129.32.19.15	
1	07 00 52 A4 20 14	129.32.19.23	

The hardware type 1 indicates Ethernet. And the hardware addresses are the Ethernet MAC addresses. Examples of other parameters are DNS server address, gateway router address, etc.

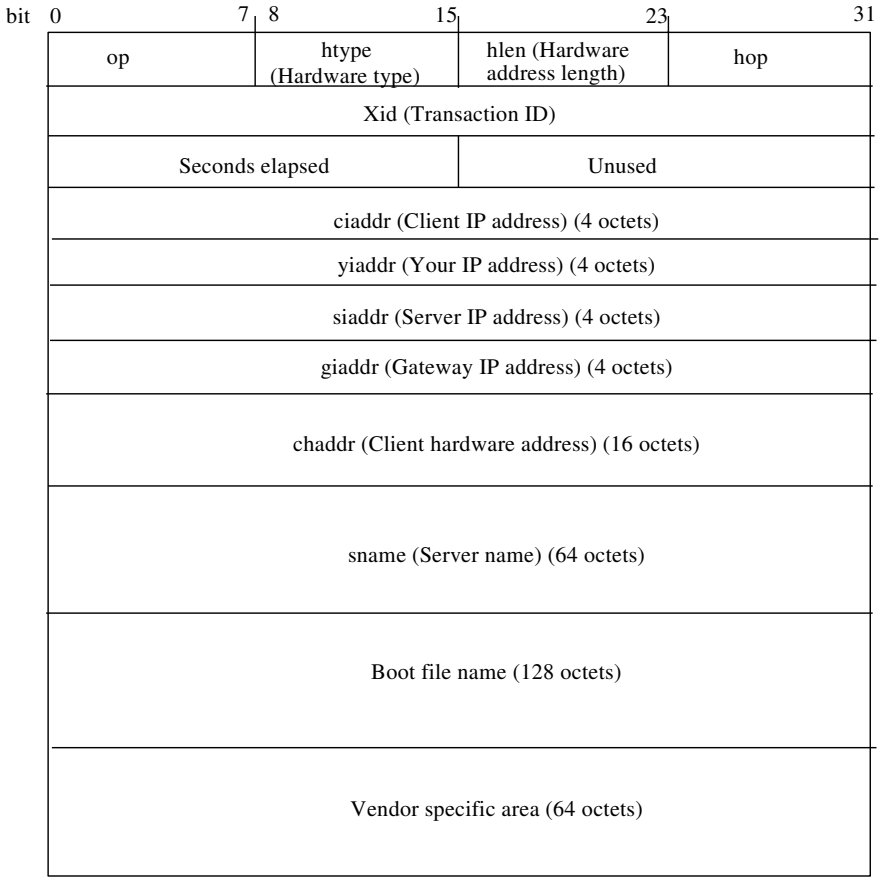
After bootstrap, the host (client) broadcasts a BOOTP request message with its hardware address identified. The source IP address is 0.0.0.0, and the destination IP address is 255.255.255.255. The server receives the request, and looks up the above table to obtain the corresponding IP address and the other parameter information. It then broadcasts the response back to the host.

---

12 W.J. Croft, J. Gilmore. Bootstrap Protocol. RFC951, Sep 1985.

13 W. Wimer , Clarifications and Extensions for the Bootstrap Protocol. RFC1542, October 1993.

The BOOTP message format is as following:



where:

op =1 if the message is a request, =0 if it is a reply.

hop Hop count is set to zero by the host. Each proxy server increases hop count by 1.

Transaction ID is set by the host and returned by the server. It is used to match a response with the request.

Seconds elapsed is set by the host to the time since it started trying to bootstrap.

If the host already knows its IP address, it fills in the *client IP address*. Otherwise the host set this to zero. In the latter case, the server fills in your IP address field with the host's IP address.

The *server IP address* is filled by the server.

If a proxy server is used, the proxy server fills in its *gateway IP address*.

The *client hardware address* is the host's hardware address, such as Ethernet MAC address.

The *server name* is filled by the server.

The *boot filename* is filled by the server. Once the host (client) receives the boot filename, it uses TFTP to retrieve it.

The *vendor specific area* is used for various extensions to BOOTP and provides additional parameters.

The fields that must be filled in by the client are: op (=request), hardware type, hardware address length, hop, transaction ID, seconds elapsed, and client hardware address.

The vendor specific area field contains optional information to be passed from the server to the client. The first four bytes of this field are called a magic cookie and define the format of remaining field. The standard format uses a magic cookie value of 99.130.83.99 (dotted decimal notation).

Options may be fixed length or variable length. All options begin with a tag octet, which uniquely identifies the option. Fixed-length options without data consist of only a tag octet. Only options 0 and 255 are fixed length. All other options are variable-length with a length octet following the tag octet. The value of the length octet does not include the two octets specifying the tag and length. The length octet is followed by "length" octets of data.

Option Type	**Code**	**Length**	**Content**
Padding	0	-	used for padding
Subnet Mask	1	4	subnet mask
Time offset	2	4	number of seconds since midnight, Jan.1, 1900 UTC
Router	3	N*4	IP addresses of gateway routers
Time server	4	N*4	IP addresses of time servers
IEN116 server	5	N*4	obsoleted
Domain server	6	N*4	IP addresses of DNS servers
Log server	7	N*4	IP addresses of log servers
Quote server	8	N*4	IP addresses of quote servers
Lpr server	9	N*4	IP addresses of lpr servers
Impress	10	N*4	IP addresses of impress servers
RLP server	11	N*4	IP addresses of rlp servers
Hostname	12	N	Client host name
Boot size	13	2	size of the boot file
Reserved	128-254		
End	255	-	end of item list

The operation of BOOTP is very simple. The BOOTP request message is sent by the client using UDP/IP and the network physical layer to the

server. The server responds by returning the BOOTP response message using UDP/IP. UDP checksum is used to detect any errors in the received BOOTP messages. If after timeout, the response is not received, the request message will be retransmitted by the client after a random delay.

BOOTP has several shortcomings. First, BOOTP works well in the relatively static environment. Each client (host) has its own IP address and configuration file. BOOTP provides a static mapping from the host identifier to the configuration parameters. Thus, it does not solve the problem of IP address shortage. Also, it does not work well for the environment where the hosts are mobile. Second, BOOTP requires the server administrator to manually input to the configuration database. To solve these shortcomings, DHCP was designed.

DHCP[14,15]

DHCP extends the capabilities of BOOTP, and has many enhancements over BOOTP. It is easier to administer. It allows automatic configuration of the client computers. It allows the client to request for specific configuration parameter values. It adds new message types that support more robust client/server interactions.

In DHCP, the server administrator allocates a block of IP addresses to be used. A block of addresses is called a scope. When the DHCP server receives a request, it can automatically assign an IP address from this block of addresses to the host (client) computer. This greatly simplifies the server administrator's work.

In DHCP, the IP addresses can be leased for a period of time or can be assigned permanently. If leased, the IP addresses are temporarily assigned, and the clients need to renew the lease before its term expires. In fact, there are three types of address allocation supported in DHCP:

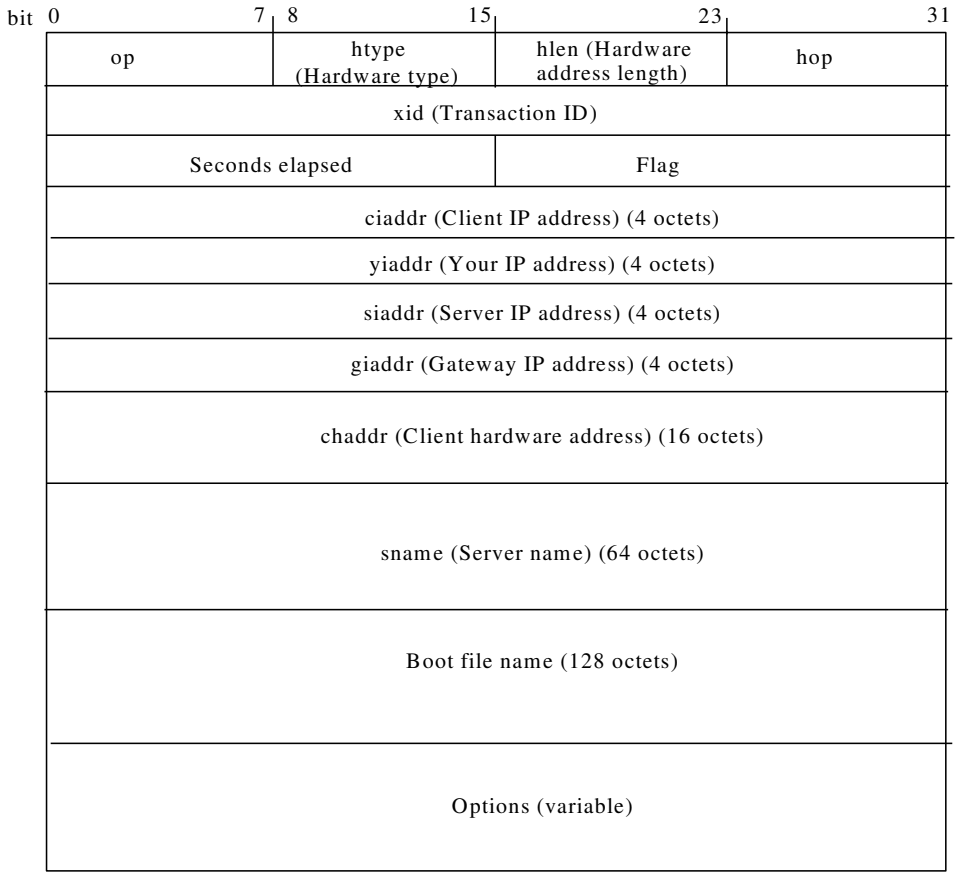
---

14 R. Droms, Dynamic Host Configuration Protocol, RFC2131, March 1997.

15 S. Alexander, R. Droms, DHCP Options and BOOTP Vendor Extensions.. RFC2132, March 1997.

- Manual allocation: Like BOOTP, an IP address that has been manually input at the server is permanently assigned to the client with an identified MAC address.
- Automatic allocation: An IP address is selected from the scope and is permanently assigned to the client.
- Dynamic allocation: An IP address is assigned to a client for a limited period, or until it is given up by the client.

The DHCP message format is the same as that for BOOTP. The exception is that the vendor specific area is renamed to options field and the length of this field is a variable (instead of 64 bytes in BOOTP). The DHCP message format is as following:



The 16-bit flag field is used by the client to request server to respond by using broadcast instead of unicast. The DHCP request message contains the client's hardware address, a server normally send its response to the client hardware address using unicast. The client can

set the leftmost bit of the flag to indicate that it requests the response be sent using broadcast.

The options field in the DHCP Message can contain all the options defined for the vendor specific area in the BOOTP message. In addition, many new options have been defined. These new items all use the same coding (tag, length, value).

Examples of the new options are:
- DHCP message types (code=53, length=1)
- Address renewal timers (T1) (code=58, length=4)
- Address rebinding timer (T2>T1) (code=59, length=4)
- Parameter request list (code=55, length=variable)
- Maximum message size (code=57, length=2)
- Request specific IP address (code=50, length=4)

BOOTP uses very simple request/response message exchange. DHCP requires a fairly complicated dialog between the client and the servers using several types of messages. The type of message is identified by the message type option in the options field.

Type Value	Message Type	Description
1	DHCPDISCOVER	Client sends to discover DHCP servers.
2	DHCPOFFER	One or more servers responds and offer an IP address and parameters
3	DHCPREQUEST	Client selects one server and send request.
4	DHCPDECLINE	Client refuses an offer.
5	DHCPACK	The server responds and provides parameters.
6	DHCPNAK	A server refuses a request. The client must start the process again.
7	DHCPRELEASE	The client releases the IP address.
8	DHCPINFORM	The client informs the server that it has a preconfigured IP address, and needs other parameters.

The operation of the DHCP protocol is as follows:

1. The client broadcasts a DHCPDISCOVER message on its local physical subnet. This message may include options that suggest values for the network address and lease duration.

2. There may be more than one DHCP servers. Each server may respond with a DHCPOFFER message that offers an available network address in the 'yiaddr' field and other configuration parameters in DHCP options.

3. The client receives one or more DHCPOFFER messages from one or more servers, and chooses one server. The client broadcasts a DHCPREQUEST message that must include the 'server identifier' option to indicate which server it has selected, and that may also include other options specifying desired configuration values. The 'requested IP address' option must be set to the value of 'yiaddr' in the DHCPOFFER message received from the server. This DHCPREQUEST message is broadcasted and relayed through DHCP/BOOTP relay agents.

4. The servers receive the DHCPREQUEST broadcast from the client. Those servers not selected by the DHCPREQUEST message use this message as notification that the client has declined that server's offer.

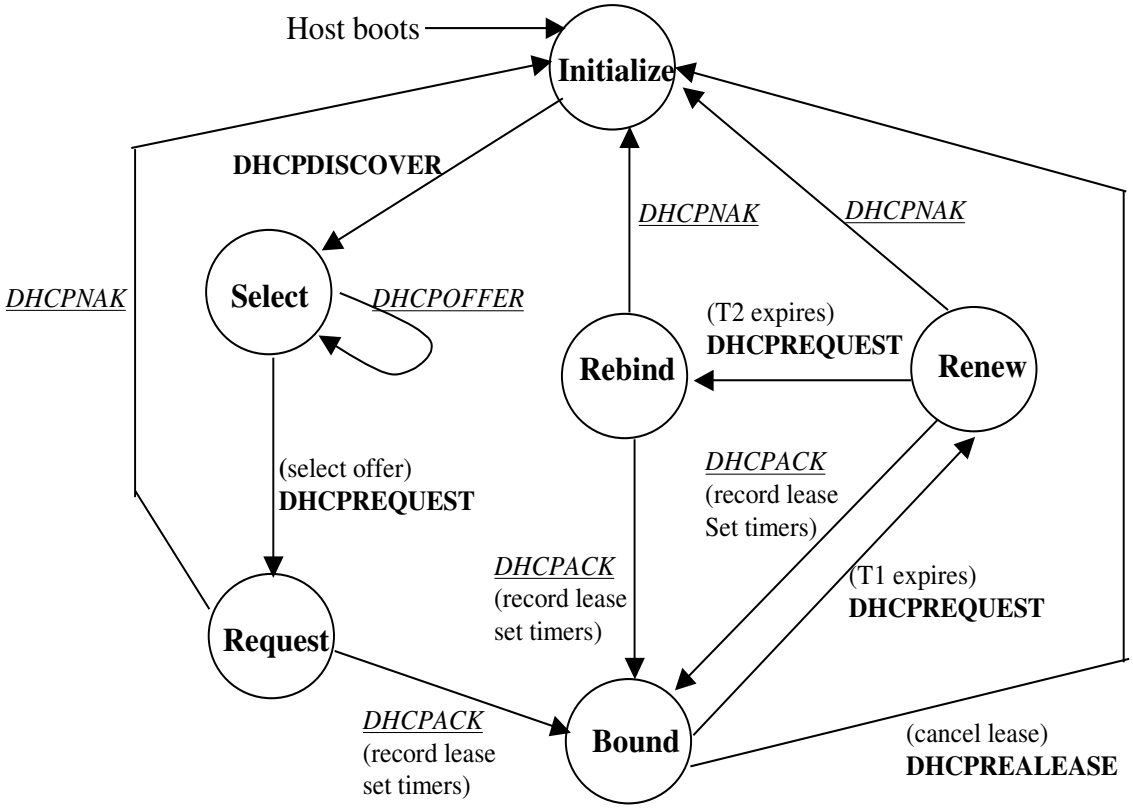
5. The server selected in the DHCPREQUEST message commits the binding for the client to persistent storage and responds with a DHCPACK message containing the configuration parameters for the requesting client. The combination of 'client identifier' or 'chaddr' and assigned network address constitute a unique identifier for the client's lease and are used by both the client and server to identify a lease referred to in any DHCP messages. If the selected server is unable to satisfy the DHCPREQUEST message (e.g., the requested network address has been allocated), the server should respond with a DHCPNAK message.

6. The client receives the DHCPACK message with configuration parameters. At this point, the client is configured. If the client detects that the address is already in use (e.g., through the use of ARP), the client must send a DHCPDECLINE message to the server and

restarts the configuration process. If the client receives a DHCPNAK message, the client restarts the configuration process.

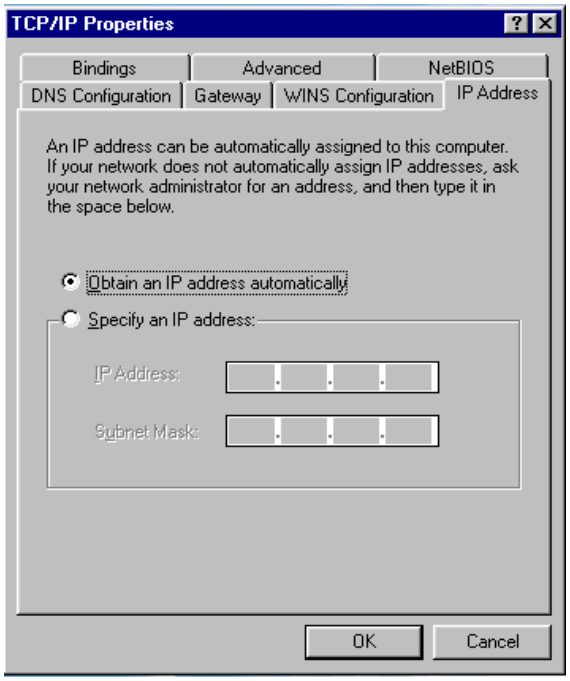
7. The client may choose to relinquish its lease on a network address by sending a DHCPRELEASE message to the server. The client identifies the lease to be released with its 'client identifier', or 'chaddr' and network address in the DHCPRELEASE message. If the client used a 'client identifier' when it obtained the lease, it must use the same 'client identifier' in the DHCPRELEASE message.

The state diagram of address acquisition and lease is as follows:

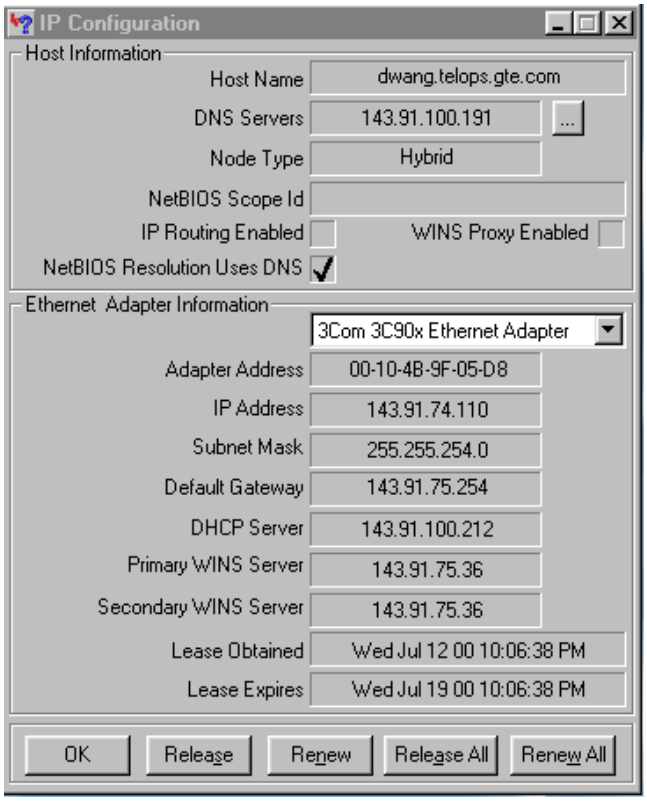


In this diagram, the messages in bold are sent by the client and the messages in italic and underlined are those received by the client. Activities in parentheses are the actions taken by the client.

The following figure shows how to select DHCP from windows network configuration menu in control panel.



The following figure shows the window produced by winipcfg after the host has been configured by DHCP.

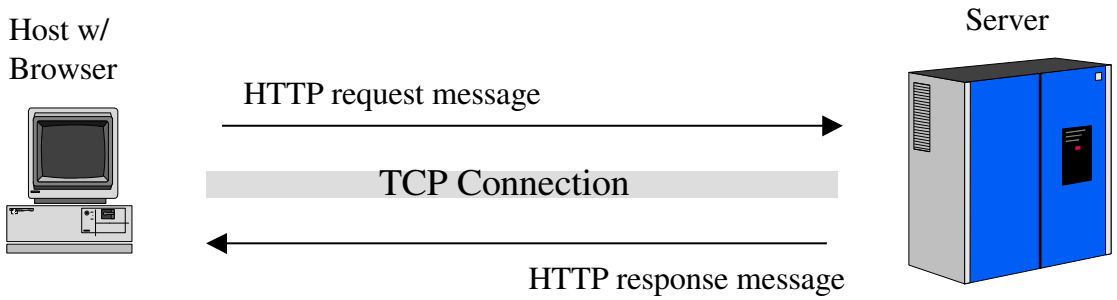


HTTP[16]

The Hypertext Transfer Protocol (HTTP) is the foundation protocol used the World Wide Web (WWW). It is a transaction-oriented client/server protocol. The typical usage of HTTP is between a web browser and a web server.

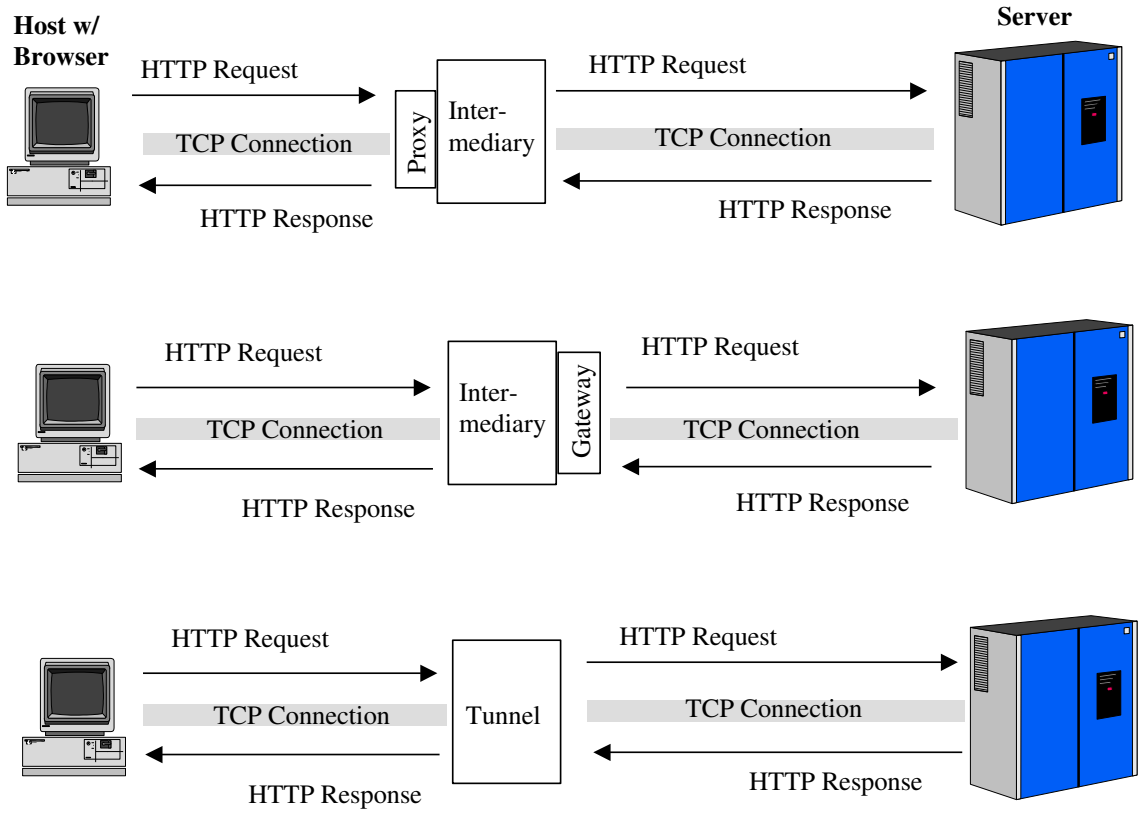
A hypertext means an underlined phrase, in a document, is associated with a pointer to another document. A user can link to the other document by clicking on the phrase.

HTTP started out as a very simple protocol. A client connects to a server via a TCP connection and requests an item. The server responds by sending the requested items and terminates the TCP connection. In the early version of HTTP, the TCP connection is terminated by the server once the response is sent. This works if the response is text only. If the response includes images, the client browser needs to open TCP connections again to request for the images. Thus, many opening and terminations of the same TCP connections are needed. To avoid this, in HTTP 1.1, persistent TCP connections are uses. In a persistent connection, a client can send multiple requests to the server without waiting for each response. The server will return the response in order.



16. R. Fielding, et. al, Hypertext Transfer Protocol – HTTP/1.1, RFC2616, June 1999.

This simple architecture is extended to include intermediary systems, such as proxy and gateway, and tunnel. They are shown here.



The proxy can be a firewall of a network, and the client is part of the network. The proxy acts as a server in interacting with a client, and as a client in interacting with a server. The proxy can also serve as an intermediary if the client and the server are running different version of HTTP. In this case, the proxy implements both versions and performs the mapping.

The gateway is a server that appears to the client as it were an origin server. It acts on behalf of the other servers that may not be able to communicate directly with a client. This gateway could be the firewall of the server network.

The tunnel performs simply as a relay point for the two TCP connections. The HTTP messages are unchanged. An example of a

tunnel is a firewall of a network which neither the client nor the server belong to.

HTTP consists of two types of messages: requests from clients to servers, and responses from servers to clients. The message is text-based. Each line is terminated by CRLF.

The general structure of a request message is as follows:

Request Line
General Header
Request Header
Entity Header
Entity Body

The general structure of a response message is as follows:

Status Line
General Header
Response Header
Entity Header
Entity Body

Where:

- Request Line: identifies the request method and the requested resource.
- Status Line: provides status information about this response. Examples are OK, forbidden.
- General Header: provides information that is applicable to both request and response messages. Examples are cache instructions, TCP connection.
- Request Header: provides information about the request and the client. Examples are image types that the client can handle, if the requested resource has been modified since some date then send it.
- Response Header: provides information about the response.

- Entity Header: contains information about the entity body.
- Entity Body: is the body of the message.

The request line indicates the request method used. HTTP/1.1 defines many request methods. Some of these methods are shown in the following:

- GET: A request to retrieve a resource.
- HEAD: A request to the server to return only the headers, not the entity body.
- POST: A request to the server to accept the attached block of data as a new subordinate to the identified URI.
- PUT: A request to the server to accept the attached block of data and stored it under the supplied URI.
- PATCH: Similar to PUT, except the entity contains a list of differences from the original resource identified in the URI.
- COPY: A request to the server to copy a resource identified by the URI in the Request Line to the location identified by the URI Header field in the Entity Header.
- DELETE: A request to the server to delete the resource identified by the URI in the Request Line.
- MOVE: A request to the server that the resource identified by the URI in the Request Line be moved to the location identified in the URI Header field in the Entity Header.
- OPTION: A request about the options available.
- TRACE: A request to the server to return whatever received as the entity body in the response.

Examples of Request Header Fields are:

- Accept: a list of media types acceptable as a response to the client.
- Accept-charset: a list of character sets acceptable to the client.
- Accept-encoding
- Accept-language
- Host: the Internet host of the resource being requested.
- If-modified-since:
- Range: indicates the range of the resource requested.
- If-range: contains a unique tag that the server sent down with the original entity, which has been cached.

- User-agent: contains information about the user agent originating the request.

The following is an example of a request message sent by a client using Microsoft Internet Explorer:

```
GET index.html HTTP/1.1
Accept: image/gif, image/jpeg, application/msword
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Range: bytes=5025-
If-Modified-Since: Sat, 4 Sep 1999 14:28:10 GMT
If-Range: “0f56d8e2236fd1:35682”
User-Agent: MSIE 4.0
Host: www.xyz.com
Connection: Keep-Alive
```

The message ends with a blank line.

The following is an example of a response message by a server at xyz company:

```
HTTP/1.1 200 OK
Server: xyz
Content-Loction: http://www.xyz.com/index.html
Date: Sun 05 Sep 1999 19:35:50 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Sun 05 Sep 1999 13:25:30 GMT
Etag: “28esdf34fg:53sc”
Content-Length: 23527

<HTM>
<HEAD>
<TITLE> ....
```

Note that the entity body contains a HTML web page.

SIMPLE NETWORK MANAGEMENT PROTOCOL

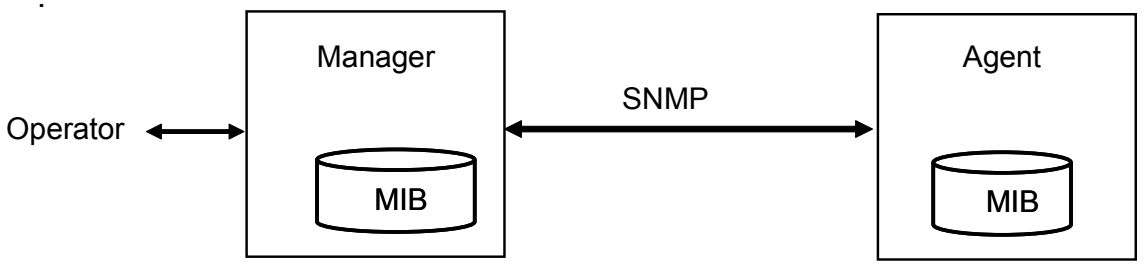
All networks, router networks, ATM networks or frame relay networks, need to be managed. All networks require configuration, performance monitoring, trouble shooting, etc. These activities are called network management.

Simple Network Management Protocol (SNMP)[17] was proposed by IETF to manage the IP network. However, SNMP is widely accepted. It has been used in other types of networks.

SNMP framework is based on the manager/agent model consisting of a manager, an agent, a structure of management information (SMI), a management information base (MIB), and the transport network protocol.

- The manager resides in the management server/system and interfaces with the operator.
- The agent resides in the network element or device being managed. It communicates with the manager.
- SMI defines the general rules for naming the objects, defining object type, range and length, and encoding of objects and values.
- The MIB creates a collection of named objects, their types and their relationship to each other in the managed device.
- The transport network protocol, SNMP, defines the format of packets exchanged between the manager and agents.

.



[17] J. Case, et. al., *Introduction to Version 2 of the Internet Standard Network Management Framework*, RFC 1441, April 1993.
D. Harrington, et. al., *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Framework, RFC 3411*, December 2002.

The MIB is organized in a tree structure with individual variables being represented as leaves on the branches. A long numeric tag or object identifier (OID) is used to distinguish each variable uniquely in the MIB and in SNMP messages. For example, the OID for the tcp object is 1.3.6.1.2.1.6.

SNMP Protocol

SNMP protocol is an application layer protocol running over UDP/IP. There are three versions of SNMP.

- **SNMPv1**[18] was the first SNMP protocol introduced, and it is still widely used. It defines "GetRequest", "GetNextRequest", "GetResponse", "SetRequest", and "Trap" packets.

  Security for SNMPv1 is based on a "community string" that is transmitted with each message. The community string acts as a password. If the Manger includes the correct password in a request to an agent, the agent will send a response. The community string is not encrypted.

- **SNMPv2** [19] introduces new types of packets: "GetBulkRequest", "InformRequest", "Report", and a new "v2trap" operation (same functionality as the v1 "trap"). SNMPv2 utilizes the same community string security as SNMPv1.

- **SNMPv3** [20] is a major step forward in improving security. Several security enhancements are defined in SNMPv3. They include:

  User Authentication: Verification of the identity of the SNMP Entity (Manager or Agent) sending the request. Managers and Agents share knowledge of valid users, and there is a shared secret key defined for each user. When an Entity sends a SNMPv3 message, the secret key is used to create a hash of the message, and this hashed value is included with the message. If the receiving Entity

[18] J.D. Case, M. Fedor, M.L. Schoffstall, J. Davin , "*Simple Network Management Protocol (SNMP*,**"** RFC 1157, May 1990.

[19] J. Case, K. McCloghrie, M. Rose, S. Waldbusser , "*Introduction to Community-based SNMP2,"* RFC 1901, January 1996.

[20] D. Harrington, R. Presuhn, B. Wijnen , "*An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*," RFC 3411, December 2002.

can recreate this hash, then the message is said to be "authenticated" as from a valid user.

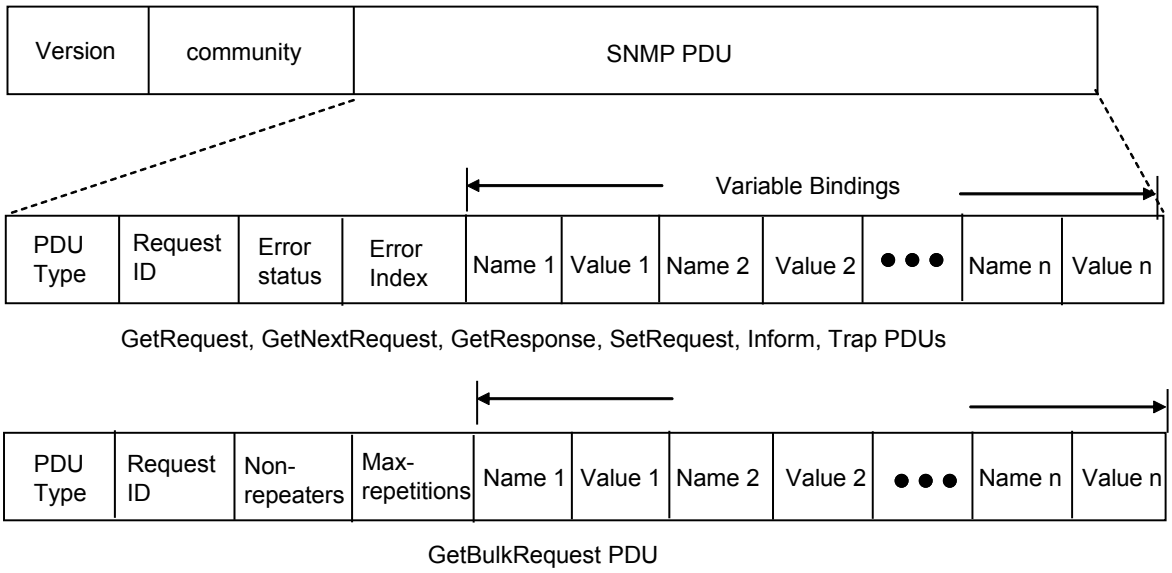
Encryption: Message payload can be optionally encrypted.

View Access Control Model (VACM): With this, agents can be configured to control that can access which MIB Objects under agent management.

The eight types of SNMP protocol data units (PDUs) are:

- GetRequest PDU is sent from the manager to the agent to retrieve the value of a variable or a set of variables.
- GetNextRequest PDU allows the manager to request information for a specific variable.
- GetBulkRequest PDU is sent from the manager to the agent to retrieve a large amount of data.
- SetRequest PDU is sent from the manager to the agent to set a value of a variable in the managed device.
- GetResponse PDU is sent from the agent to the manager in response to the GetRequest, GetNextRequest, or GetBulkRequest PDU received.
- Trap PDU is sent from the agent to the manager to report an event (e.g., a port is down).
- InformRequest PDU is sent from a manager to another manager to get the value of some variables from the agent that the other manager manages.
- Report PDU reports some types of errors. It is not yet in use.

The port numbers used are 162 for trap and 161 for other PDUs. The SNMPv2 PDU formats are as follows:



where

- **Version number**— Version of the SNMP that is being used.
- **Community name**— Serve as a weak form of authentication because devices that do not know the proper community name are precluded from SNMP operations
- **PDU type**—Identifies the type of PDU transmitted
- **Request ID**—An ID that associates SNMP requests with responses.
- **Error status**—Indicates one of a number of errors and error types. Only the response operation sets this field. Other operations set this field to zero.
- **Error index**—Associates an error with a particular object instance. Only the response operation sets this field. Other operations set this field to zero.
- **Variable bindings**—Serves as the data field of the SNMPv2 PDU. Each variable binding associates a particular object instance with its current value (with the exception of Get and GetNext requests, for which the value is ignored).
- **Non repeaters**—Specifies the number of object instances in the variable bindings field that should be retrieved no more than once from the beginning of the request. This field is used when some of the instances are scalar objects with only one variable.

- **Max repetitions**—Defines the maximum number of times that other variables beyond those specified by the Non repeaters field should be retrieved.

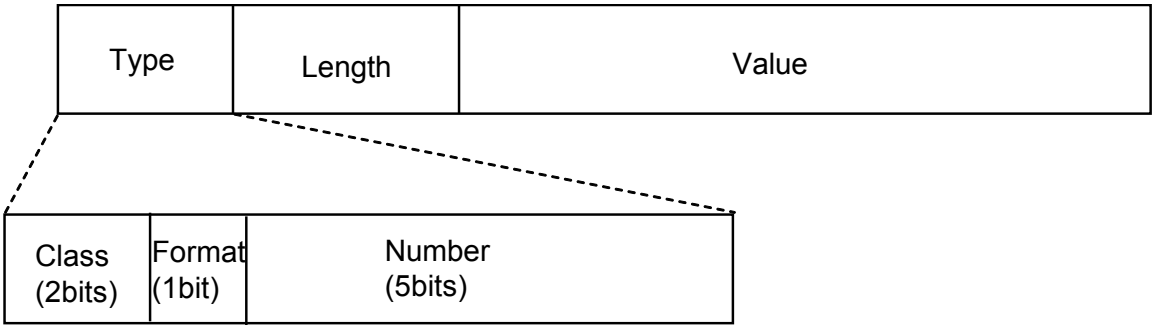
Structure of management Information (SMI)

Each SNMP element manages specific objects with each object having specific characteristics. The structure of management information (SMI) defines the rules for naming the managed objects, defining object type, and encoding of the managed objects and their values.

Collections of related objects are defined in the management information base (MIB) modules. The SMI specifies that all MIB variables are defined using Abstract Syntax Notation One (ASN.1), which describes a notation that human reads and an encoded representation of the same information used in the communication protocol. SMI specifies that Basic Encoding Rule (BER)[21] is used to encode the data to be transmitted between the manager and the agents.

In the SMI, several data types are allowed. The primitive data types are INTEGER, OCTET STRING, NULL, and OBJECT IDENTIFIER. There are also additional user-defined data types that are application specific.

BER specifies that the data is encoded in triplet form: type (or called tag), length, and value, as shown in the following:



Four classes are defined: universal (00), application-wide (01), context-specific (10), and private (11). Format subfield indicates whether the data is simple (0) or structured (1).

[21] ITU X.690, "*Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DE)*," July 1994.

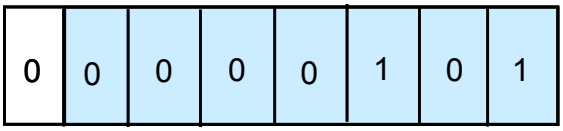
The following table lists the codes of some data types:

Data Type	Class	Format	Number	Type (binary)	Type (hex)
INTEGER	00	0	00010	00000010	02
OCTET STRING	00	0	00100	00000100	04
NULL	00	0	00101	00000101	05
OBJECT IDENTIFIER	00	0	00110	00000110	06
SEQUENCE, SEQURNCE OF	00	1	10000	00110000	30
IPAdress	01	0	00000	01000000	40
Counter	01	0	00001	01000001	41

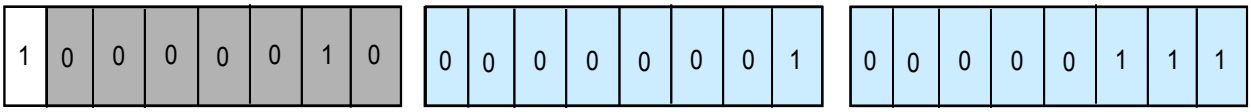
SNMP defines the following for the PDU types:

Data Type	Class	Format	Number	Type (binary)	Type (hex)
GetRequest	10	1	00000	10100000	A0
GetNextRequest	10	1	00001	10100001	A1
Response	10	1	00010	10100010	A2
SetRequest	10	1	00011	10100011	A3
GetBulkRequest	10	1	00101	10100101	A5
InforRequest	10	1	00110	10100110	A6
Trap (SNMPv2)	10	1	00111	10100111	A7
Report	10	1	01000	10101000	A8

The length field is one or more bytes. If it is only one byte, the most significant bit is zero, the remaining seven bits specifies the length of the data. The following shows a single byte indicating the length of 5.



If it is more than one byte, the most significant bit must be 1. The remaining seven bit of the first byte (the shaded area in the following figure) specifies the number of bytes used to define the length. The following shows two bytes are used to indicate the length of 263 bytes for the data field.



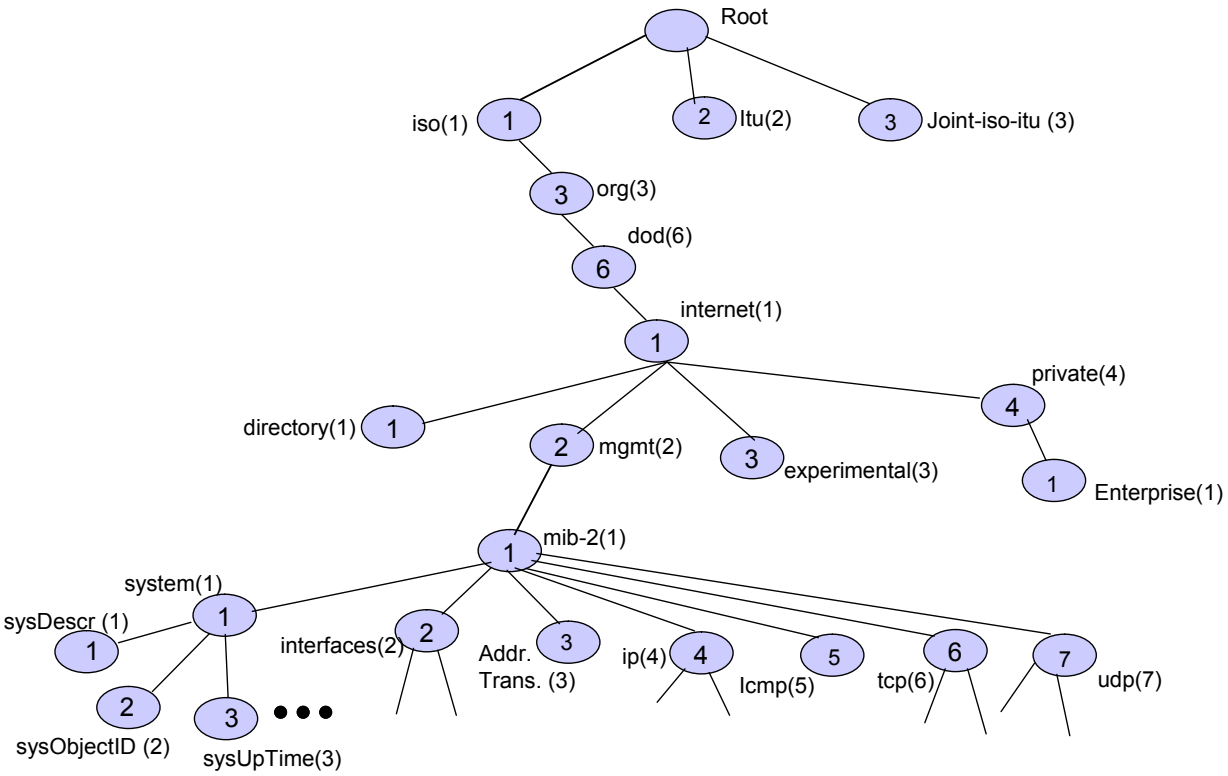
Management Information Base (MIB)

Management Information Base (MIB)[22] is a virtual database used to define the functional and operational aspects of the devices being managed. The database contains an object for each functional aspect of the managed device. These objects are grouped into different modules, or MIB groups.

Each object has a unique object identifier (OID) consisting of numbers separated by decimal points (e.g., 1.3.6.1.2.1.1.3). The OID is a sequence of non-negative integers, where each integer corresponds to a node in a tree. This OID identifies the managed objects and relates its place in the object hierarchy. The figure below shows the object identifier tree defined. The SNMP MIB associates each OID with a

[22] K. McCloghrie, M.T. Rose, "*Management Information Base for network management of TCP/IP-based internets,"* RFC 1156, May 1990.
K. McCloghrie, M. Rose, "*Management Information Base for Network Management of TCP/IP-based internets:MIB-II*," RFC 1213, March 1991.
R. Presuhn, "*Management Information Base (MIB) for the Simple Network Management Protocol (SNMP),"* RFC 3418, December 2002

readable label (i.e., sysUpTime) and various other parameters related to the object. The MIB then serves as a data dictionary that is used to assemble and interpret SNMP PDUs.



In BER, there are two rules concerning the encoding of OID. The first rule applies when encoding the first two numbers in the OID. According to BER, the first two numbers of any OID (x.y) are encoded as one value using the formula (40*x)+y. The first two numbers in an SNMP OID are always 1.3. Therefore, the first two numbers of an SNMP OID are encoded as 43 in decimal or 0x2B in hex, because (40*1)+3 = 43. After the first two numbers are encoded, the subsequent numbers in the OID are each encoded as a byte. However, another special rule is required for large numbers. Large numbers may not be represented by one byte. For example, the number 2680 cannot be encoded using a single byte. The rule for encoding large numbers states that only the lower 7 bits in the byte are used for holding the value (0-127). The highest order bit is used as a flag to let the recipient know that this number spans more than one byte. Therefore, any number over 127

must be encoded using more than one byte. According to this rule, the number 2680 must be encoded 0x94 0x78. Since the most significant bit is set in the first byte (0x94), the recipient knows to use the lower 7 bits from each byte (0x14 and 0x78) and decode the two bytes as (0x14 *128) + 0x78 = 2680.

An example is system group, which has OID of 1.3.6.1.2.1.1 and has the following objects: sysDescr, sysObjectID, sysUpTime, sysContact, etc. The syntax for the sysUpTime object is

sysUpTime **OBJECT-TYPE**
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The time (in hundredths of a second) since the network management portion of the system was last re-initialized."
::= {system 3}

It shows that the sysUpTime object has an integer value, the access is read-only, and it is a mandatory object. This object is in the system group.

Another example of a MIB group is ip, which has the OID of 1.3.6.1.2.1.4. This group is used to store information about the transactions related to IP. The syntax for the ipInHdrErrors object is

ipInHdrErrors **OBJECT-TYPE**
SYNTAX Counter
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The number of input datagrams discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc."

::= {ip4}

Encoded SNMP PDU

The following illustrates an ASN.1 encoded GetRequest PDU sent by the manager to an agent to request the system description (sysDescr, OID=1.3.6.1.2.1.1.1). Note that it is encoded using BER, all the numbers are expressed in hexadecimal, and the shaded rows show the corresponding meaning that human can read.

```
    30          29          02          01          01
SEQUENCE   length=41   INTEGER     Length=1   version=2 (1 for vesion2)

    04          06          07     75     62     6C     69     63
OCTETSTRING  len=6         P      u      b      l      l      c

    A0          1C          02     04     05     AE     56     02
GetRequest   len=28   INTEGER   Len=4        --  request ID (4 char) --

    02          01     00       02          01     00
INTEGER    len=1      Status   INTEGER   Len=1    Error Index

    30          0E          30          0C          06          08
SEQUENCE   len=14   SEQUENCE    Len=12   OBJECTID   Len-8

    2B     06     01     02     01     01     01     00
    1.3    6          1      2      l      l          1      0

    05          00
    NULL       len=0
```

This string of message is encapsulated by UDP header and IP header,