CHAPTER 1

The TCP/IP protocols

In this first chapter we will make a deep study of the basic protocols that permit the performance of Internet. The family of protocols TCP/IP is the cornerstone on which the rest of services (WWW are supported, FTP, SSH…) that we can currently find in Internet, and therefore its study is fundamental part for the rest of work.

The address system, the format used in its bedsides and the method of performance of the most important protocols will be introduced than this family:

- The protocol of flow control (ICMP).

- The non reliable protocol of transmission of data without connection (UDP).

- The reliable protocol of transmission of data with connection (TCP).

The routing process will finally be explained of the IP datagrams as well as its transit for the local network until the addressee computer.

This study will only concentrate on version 4 of the protocol IP, since although currently it is working in version 6 is [Ver00] the latter still experimental.

1.1 IP Networks

We will define the networks IP as those networks that use the protocols TCP/IP for their performance. Internet is a network IP.

The families of protocols TCP/IP "*permit communication between different types of computers regardless of the manufacturer, network to which connected and operative system used are."* [Ric98-1]

The networks IP are characterized by constructing by following a strict scheme of layers. Each layer is the person responsible for the each one of the different facets of communication. In this way, the family of protocols TCP/IP can be defined as a combination of four layers (see figure 1-1) according to the model OSI [Ric98-1].

APPLICATIONS Layer	WWW, SSH, FTP...
TRANSPORT Layer	TCP, UDP, ICMP...
NETWORK Layer	IP
LINK Layer	IEEE 802.2, 802.3...

FIG. 1-1: Four layers structure.

In this scheme, the superior layer attains only services provided by the layer situated just in the level below it. In this way, we free a layer of the rest of inferior layers, which allows us to have a modular scheme.

- **The link layer either real network layer**, also named the data layer or network interface layer, includes the mechanisms that they allow the operative system to send and receive information through the network to which it is physically connected (Ethernet, XDSI…).

- **The network layer or layer**, also named Internet layer, is the responsible to move the packages of information through the different networks to arrive at its destination. In this layer we find the protocols of lower level, emphasising the IP (Internet Protocol).

- **The transport layer** is the responsible to provide a flow of data between two computers. This flow of data may be reliable (Transmission Control Protocol, TCP) or not reliable (User Datagram Protocol, UDP).

- **The application layer** is the representative to handle the particular details relating to the different applications that the user will enjoy (WWW, TELNET, FTP…).

This system permits independence among the different layers and obliges that communication between two computers is made through a communication between the layers of the same level of both computers (see figure 1-2).

Communication in Internet happens through the exchange of packages of information among the different computers. These packages of information (also named datagrams) travel by the different computers that are connected to Internet until they reach its aim or they are discarded for some reason.

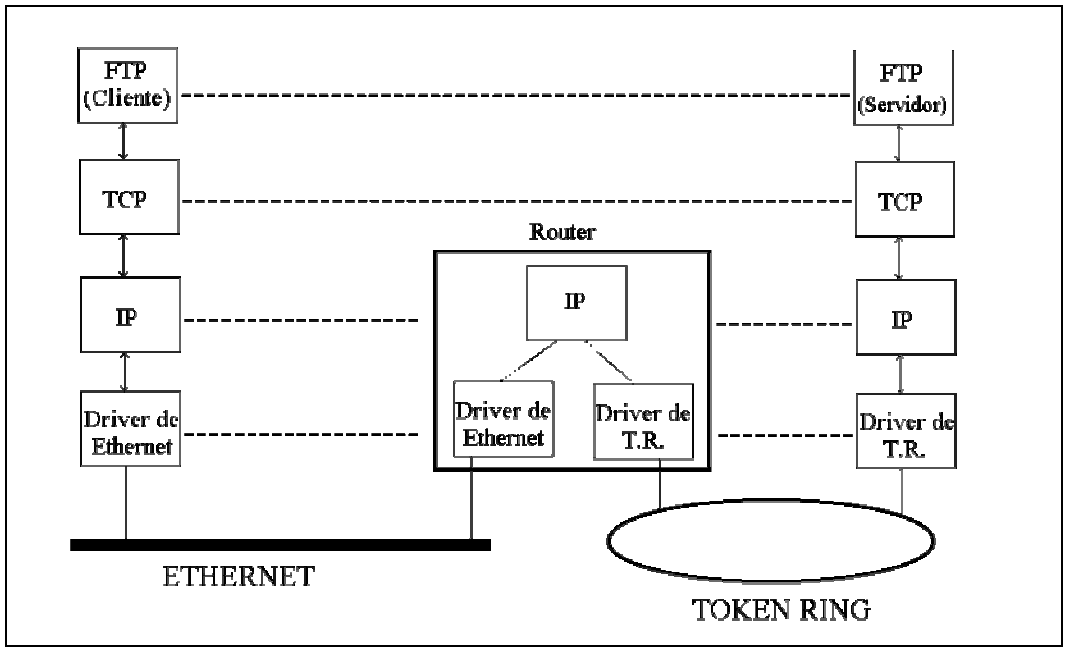

FIG. 1-2: Scheme of two computers connected to Internet.

In this way, in the communication of two computers by Internet we can differentiate two types of duties that can play the computers by which the packages of information are passed on:

1. **Transmitting computer / receiver (end-system or end-host):** The origin or destination computer in the communication process.

2. **Intermediate computer (intermediate-system, router or gateway):** Would be all the computers or systems travelled by datagrams or packages of information until the destination of communication will be reached (or origin in the case of answer).

The IP protocol has a numeration system that permits differentiating the each and every one of the connected computers. In version 4 of the protocols TCP/IP, these addresses have to satisfy two basic requirements (see figure 1-3):

1. They must be unique. It cannot be two computers with the same address.

2. The address is 32 bits (4 bytes) numbers. These addresses are represented through four decimal numbers separated by a point.

CLASS	RANGE		
A	0.0.0.0	To	127.255.255.255
B	128.0.0.0	To	191.255.255.255
C	192.0.0.0	To	223.255.255.255
D	224.0.0.0	To	239.255.255.255
E	240.0.0.0	To	247.255.255.255

FIG. 1-3: IP address classes found in Internet.

This type of address system, allows a large flexibility to us when defining networks that we will connect afterwards to Internet. So, an A class address would be ideal for very large networks, since 128 networks (2^7) of 16.777.216 (2^24) computers are allowed in this scheme.

B class permit 16.384 (2^14) networks with 65.535 computers, and a C class permits 2.097.152 (2^21) networks of 256 computers.

D class (multicast) and E class (reserved) are used for different possibilities as that of having computers in different networks and that were seen as if they were in the latter (Ex. 2 computers in the UAB University, 1 in UPC University and 10 in the MIT. All of them received the same information, as in a multi-conference).

However these peculiarities go beyond the purposes of this work, and the reading of bibliography is recommended [Ric98-1] for more information.

Having defined the address system of networks and computers in Internet, we will mention the existence of DNS services (Domain Name Server).

Due to the fact that it is easier to recall a name (Calculus Centre of the Autonomous University of Barcelona, cc.uab.es) than a numerical address (158.109.0.4), the name servers or DNS were created. These services translate the given name (cc.uab.es) into their corresponding numerical address (158.109.0.4).

1.2 IP protocol version 4

The protocol IP (Internet Protocol) is the fundamental part in the TCP/IP system and its specification is collected in [RFC791]. The data unit of the protocol IP is the datagram (see figure 1-4), whose maximum size is of 65535 bytes (64K).

The protocol IP facilitates a system without any connection method (*connectionless*) and non reliable delivery of datagrams between two computers connected to Internet.

IP gives a service of delivery based on the best attempt (*best effort*). This implies that when there is some anomalous performance of Internet, as a collapsed router, an error very simple treatment system is considered. This error control mechanism is regulated by the protocol ICMP (*Internet Control Message Protocol*).

In our case, the collapsed router would discard the datagram and would send an ICMP error message to the origin computer without taking care of datagram retransmission, which does not imply reliability.

Furthermore, it does not keep any type of information referring to the state of the connections. Each datagram it is directed independently, which it converts into a protocol without connection.

Due to these particular characteristics, it can happen that some datagrams could be lost and/or that they do not arrive in correct order. In this way, any reliability that is needed will have to be made by the superior layers (TCP…).

The structure of an IP datagram is divided into blocks of 32 bits (4 bytes). The IP datagram is passed by sending first bit 0, then bit 1, 2, 3… and so on even finishing the datagram transmission.

This order is named "**network byte order**". It is very important to know this order of transmission of the information, since the different computers have different memory storage ways for bits.

The **little endian** format, consists of storing the bits in inverse order to the byte network order (used in Intel processors for example), while the other possibility is named **big endian** (used for example in the Motorola processors).

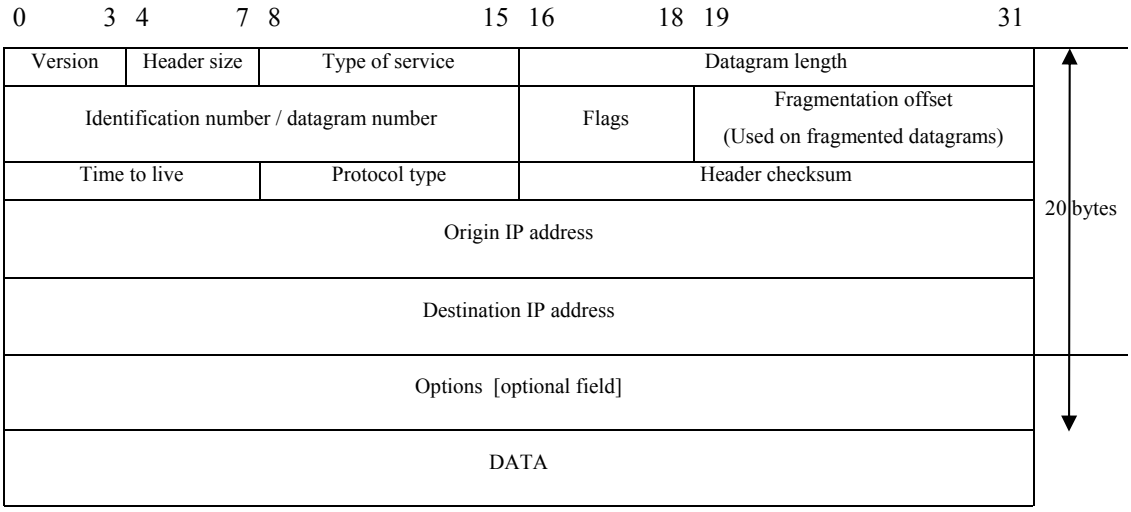

FIG. 1-4: IP v4 datagram schema.

The **version field** (4 bits), serves to identify to that specific version (RFC) refers the format of the datagram. This information is only used by the routers and layer IP of origin and end of the datagram. This permits the coexistence of different versions of the protocol IP in a transparently way to the user. The current version is the 4 (also known as IPv4).

Header Length, are 4 bits (2^4 = 16 positions, 0...15) that indicates the number of words of 32 bits that occupies the header. These 4 bits of maximum size us limit to a maximum size of header of 60 bytes (15 * 32 bits = 60 bytes). However, the usual value of this field is 5.5 * 32 bits = 20 bytes).

The field of the type of service is made up of 8 bits. First 3 bits have an obsolete function and they are not considered currently. The 4 following bits define the type of service (see figure 2-12) and the last bit is not used currently and must have value 0.

Only 1 of the 4 bits of the type of service can be active at the same time. The type of service determines the policy to continue in the sending of the datagram by Internet. The possible options are:

1. Minimize delay.
2. Maximize throughput.
3. Maximize reliability.
4. Minimize monetary COST.

Application type	Minimize delay	Maximize throughput	Maximize reliability	Diminish cost	Hexadecimal value
TELNET	1	0	0	0	0x10
FTP	0	1	0	0	0x08
SMTP	0	1	0	0	0x08
DNS (UDP)	1	0	0	0	0x10
DNS (TCP)	0	0	0	0	0x00
ICMP	0	0	0	0	0x00
BOOTP	0	0	0	0	0x00

FIG. 1-5: Typical values of the type service according to the implementation.

The datagram Length, is a number of 16 bits (2^16 = 65536, 0... 65535) that indicates the total length of the datagram. This value is very important, since it allows us to know that memory size we must reserve for the reception of the datagram. Furthermore, it indicates the number to us of bytes to read, which allows a simple error control

mechanism. In this way, if the value is incorrect, the number of bytes read will be at most of 65535, by noting the error. Furthermore us it limits the number of bytes to send in a datagram (Maximum Transfer Unit, MTU) to 65535 – 20 (typical size of the header) = 65515 bytes.

If the length of datagram is larger than the maximum size of any travelled network (Ex. datagram of 32000 bytes goes through Ethernet network. Maximum size accepted for this kind of LAN is 1500 bytes), the datagram is fragmented into N small pieces.

Identifications Field is a number of 16 bits that in case of a fragmentation of a datagram indicates its position into the original datagram. This allows us to rebuild the original datagram in the destination computer. This value indicates that a datagram can be fragmented in a maximum of 65535 fragments.

The flags are 3 bits long field. Firstly they permit indicating if the received datagram is a fragment one, bit M (More) activated. The second specifies if the datagram does not have to be fragmented, bit DF (Don't fragment) and the third is not used currently, assigning a zero value. [San99].

The Fragmentation Offset, indicates the position in bytes that the data in the original datagram occupy. It has only sense if the fragment belongs to a greater datagram that been fragmented. This field has a maximum of 13 bits (2^13 = 8192, as it indicates relocation to us in bytes 8192 * 8 bits = 65536). In this way, we can always rebuild the original datagram with the fragments.

Time to Live, is an 8 bits field that indicates maximum time that the datagram will be valid and be able to be passed over the network. This permits a control mechanism to avoid datagrams that circulate eternally by the network (for example in the case of loops).

This field is set in the origin to a value (maximum 2^8 = 256) and is decremented in a unit whenever it goes through a router. In this way, if a loop is produced and/or it does not reach its destination in a maximum of 255 "jumps", it is discarded. In this happens, an ICMP error datagram is sent to the computer of origin to inform of its loss.

The type of protocol field is a value that indicates to which protocol the datagram (TCP, UDP, ICMP…) belongs. It is necessary due to the fact that all the Internet services use IP as transport, which makes necessary a discrimination mechanism among the different protocols.

The header checksum field is a control mechanism that affects only the header of the IP datagram. The rest of protocols TCP, UDP, IGMP… have their own header and checksum. Its function is to provide a simply error control mechanism. In this way, if an error in the checksum of a datagram IP is found, it is simply discarded and no message of error is generated. This implies that of the superior layers must control the flow of the datagrams to ensure that the latter arrive correctly at the destination, either using a reliable protocol (TCP) or implementing internally some type of control.

Origin and destination IP address are formed by two numbers of 32 bits. These addresses are corresponded to a distribution according to the figure 1-3.

1.3 The ICMP protocol

The ICMP protocol (Internet Control Message Protocol) is a simple protocol that is encapsulated into IP datagrams (see figure 1-6) and controls the flow of communication as well as the communication of errors [RFC792].

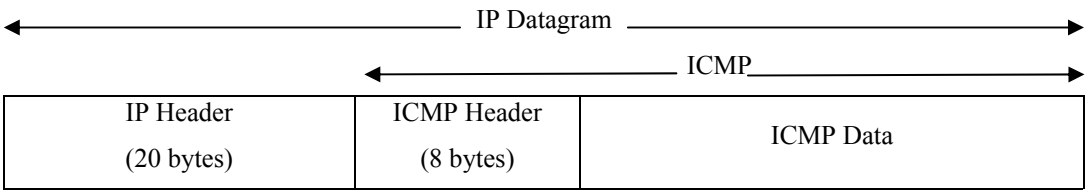

FIG. 1-6: ICMP message structure.

The header of the ICMP protocol has a size of 8 bytes that contains several fields (see figure 1-7) that permit the identification of the message.

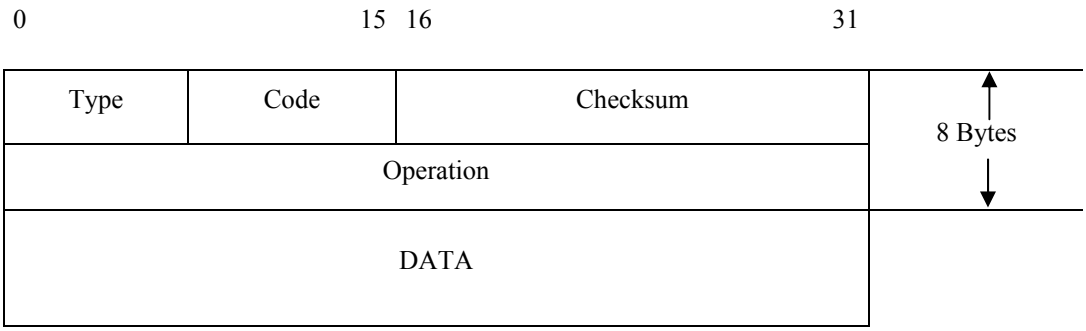

FIG. 1-7: ICMP header scheme.

The Type indicates the nature of the sent message, since the protocol permits specifying a wide variety of errors or messages of flow control of communications (see figure 1-8).

The Code field indicates the number of error within the type of error indicated in the "type" field. That is to say, it groups the messages in types and within each type it specifies the specific code to which we refer (see figure 1-8).

The Checksum permits simply checking the integrity of the sent message. This feature permits detecting possible errors in the shipment, transport or reception of the message of control ICMP.

The Operation field depends directly on the content of the type field and code field permitting the inclusion of extra information's referring to the code of error.

TYPE	CODE
0	Echo Reply Codes: 0 No Code
1	Unassigned
2	Unassigned
3	Destination Unreachable Codes: 0 Net Unreachable 1 Host Unreachable 2 Protocol Unreachable 3 Port Unreachable 4 Fragmentation Needed and Don't Fragment was Set 5 Source Route Failed 6 Destination Network Unknown 7 Destination Host Unknown 8 Source Host Isolated 9 Communication with network is Administratively Prohibited 10 Communication with Host Administratively Prohibited 11 Destination Network Unreachable for TOS 12 Destination Host Unreachable for TOS
4	Source Quench Codes: 0 No Code
5	Redirect Codes: 0 Redirect for the Network 1 Redirect Datagram for the Host 2 Redirect for the TOS and Network 3 Redirect for TOS and Oct
6	Alternate Host Address Codes: 0 Alternate Address for Oct
7	Unassigned
8	Echo Codes: 0 No Code
9	Router Advertisement Codes: 0 No Code
10	Router Selection Codes: 0 No Code
11	Time Exceeded Codes: 0 Time to Live exceeded in Transit 1 Fragment Reassembly Time Exceeded
12	Parameter Problem Codes: 0 Pointer indicates the error 1 Missing a Required Option 2 Bad Length
13	Timestamp Codes: 0 No Code
14	Timestamp Reply Codes: 0 No Code
15	Information Request Codes 0 No Code
16	Information Reply Codes 0 No Code
17	Address Mask Request Codes 0 No Code
18	Address Mask Reply Codes: 0 No Code
19	Reserved (for Security)
20-29	Reserved (for Robustness Experiment)
30	Traceroute
31	Datagram Conversion Error
32	Mobile Host Redirect
33	IPv6 Where-Are-You
34	IPv6 I-Am-Here
35	Mobile Registration Request
36	Mobile Registration Reply

FIG. 1-8: ICMP type and code messages table.

1.4 The UDP protocol

The UDP protocol (User Datagram Protocol) might be defined as a simple protocol and datagram oriented [Ric98-1]. Its definition is picked up in [RFC7680] published by Postel in 1980.

In this way, each data transmission corresponds to a single sending of an independent datagram of the rest of datagrams of the same communication (see figure 1-9).

In this way, following the rules of routing of datagrams by Internet, the delivery to the destination is not ensured by the protocol. What is more, neither it is so at least ensured that the datagrams arrive in the order in which they were sent.

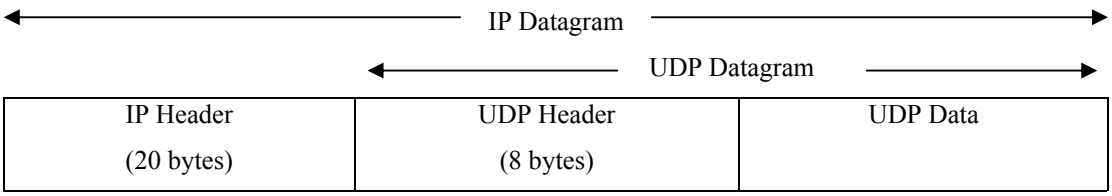

FIG. 1-9: UDP datagram structure.

Due to the fact that UDP uses IP for its transport for Internet, in case of being necessary (by different sizes of MTU, for example) this it will be fragmented and none of these fragments (like the original datagram) provide any type of safety or reliability in the delivery.

Due to the simplicity of this protocol, the design of its bedside (figure 1-10), is much simpler than that of IP.

The port number is used in communications between two computers to differentiate the different existing connections. If we have several communications from our computer (for example a TELNET to the cc.uab.es looking email and a file transfer or

FTP to blues.uab.es at the same time) on receiving an IP datagram must know to which of the two connections belongs.

If we assigning a port number to a communication we can know to which connection belongs every datagram. If this port number is 16 bits wide, we can deduce that the maximum number of connections that a computer can have simultaneously in use is 65535 (2^16). In the bibliography [WWW58] a complete list of services associated with the ports TCP or UDP can be obtained.

The length field of UDP datagram refers to the size of the datagram in bytes, and includes the header (8 bytes) plus the data that it transports. The minimal length value is 8 bytes (so, the protocol permits sending a datagram UDP with 0 bytes).

This field is superfluous, since it uses IP for its transport, and the latter incorporates already a field for the length of the data (see figure 2-11) that would be the length of the datagram IP except the size of the header.

The checksum field as in IP protocol, serves as a control method as the data, checking that they have not been altered. This checksum covers the header UDP and the sent data. It is necessary due to the fact that the checksum of the protocol IP covers the IP header only and not the data that it transports. If an error in the checksum is detected, the datagram is discarded without any type of news.

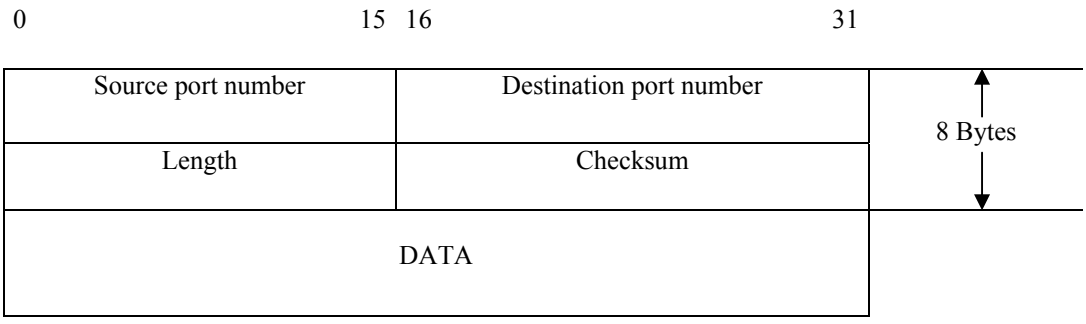

FIG. 1-10: UDP header schema.

1.5 The TCP protocol

The TCP protocol (*Transmission Control Protocol*) might be defined as a protocol having an aim to connection and reliability. It's focused to a flow of bytes [Ric98-1]. Its definition is picked up in [RFC793] published by Postel in 1981.

Although the protocol TCP like UDP uses IP services for its transport for Internet (see figure 1-11), it is a protocol having a connection procedure. This means that both applications surrounded in communication (usually a client and a server), must establish a communication previously before being able to exchange data.

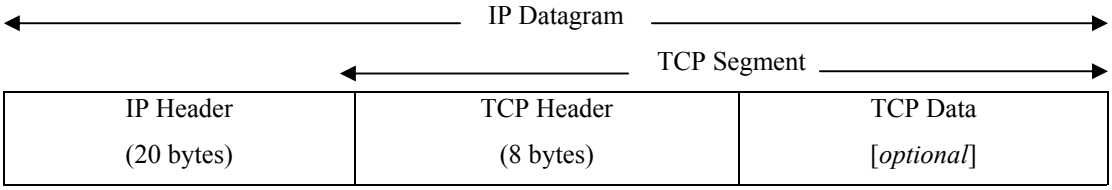

FIG. 1-11: TCP segment structure.

TCP is also a reliable protocol. The reliability provided by this protocol is given principally by the following aspects:

1. The data sent are regrouped by the protocol in portions named *segments* [Ric98-1]. The size of these segments is assigned by the protocol itself and can be different. This differentiates from UDP, where each generated portion of data corresponds to a datagram.

2. When a TCP connection receives a complete segment, the receiver sends an answer of confirmation (Acknowledge) to the transmitter confirming the number of received correct bytes. In this way, the transmitter commits the bytes and can continue to send new bytes.

3. When a segment is sent a timer is started. In this way, if in a fixed space of time a confirmation (Acknowledge) of the sent data is not received, the date is retransmitted.

4. TCP incorporates a checksum to check the validity of the received data. If an erroneous segment is received (failure of checksum for example), a confirmation is not sent. In this way, the transmitter sends the data (bytes) another time.

5. The IP protocol does not guarantee the order of arrival of the datagrams. The TCP protocol uses numbers of sequence to assure the reception in order, avoiding changes of order and/or lies of the received bytes.

6. TCP is a protocol that implements a data flow control. In this way the data transmission can be adjusted in each segment, avoiding the collapse of the receiver. This collapse would be possible if the transmitter sent data without expecting the confirmation (ACK) of the already sent bytes.

The header of the segment TCP (figure 1-12), is substantially more complex than that of UDP due to the fact that communication is drawn up and must provide reliability. This implies a series of additional information that must be kept to be able to meet the state of communication at any moment.

The number of origin port and destination serve to differentiate a communication in a computer of the others. It complies with the same function as in the datagram UDP.

The set formed by the IP address and the number of port is named **socket**. This name used soon in the specification of the interface of programming of Berkeley (API).

Similarly, the group of the two sets that define a connection between two computers is named socket pair. In [WWW58] a complete list of services associated with the ports TCP and UDP can be obtained.

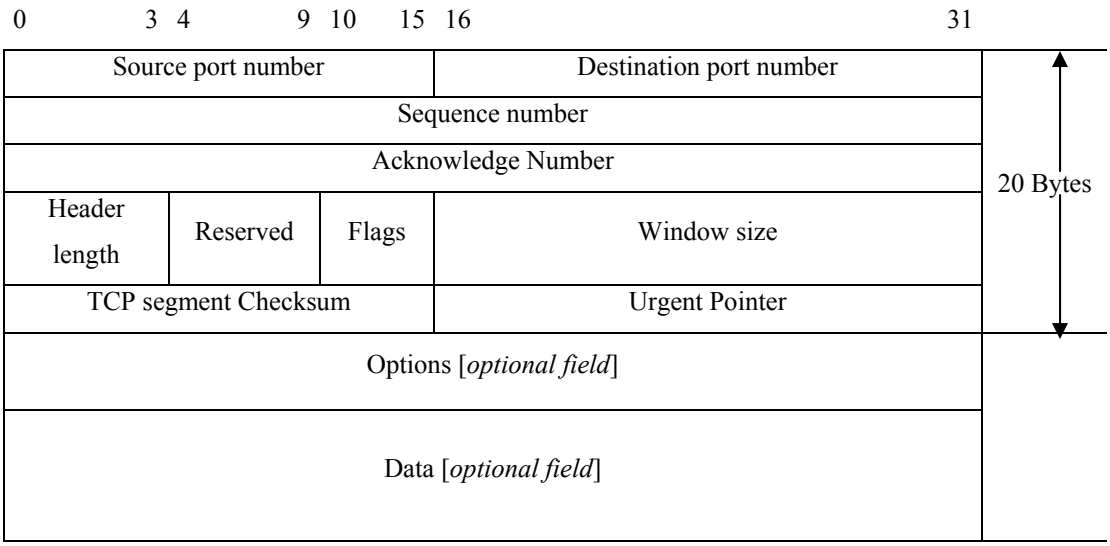

FIG. 1-12: TCP header.

The Sequence number identifies the specific byte of the flow of data that is currently sent of the transmitter to the receiver. In this way, TCP numbers the bytes of the communication in a consecutive way from the number of initial sequence. When a communication is established, transmitter and receiver choose a common sequence number, which allows control mechanisms to assure that the data arrive in the right order. It is a number of 32 bits, so we can send 2^32-1 bytes before it starts again (reset).

The acknowledge number is the number of sequence plus one. In this way it is specified to the transmitter that sent data until this number of sequence minus one are correct. We can see the importance of the election at the beginning of the communication of a common sequence number.

The header length specifies in a 32 word bits (4 bytes) the size of the header of the segment TCP including the possible options. In this way the maximum size is 15 * 4 = 60 bytes. However, the usual thing is to have a size of 20 bytes (simple header without options).

The flags are representation numbers to specify the different states of communication. It can be simultaneously activated. In figure 1-13 we can see the different meaning of the existing flags.

FLAG	Meaning
URG	Urgent pointer field has been set.
ACK	Acknowledge fields specified.
PSH	Data must be pushed to the application as fast as possible.
RST	Connection RESET.
SYN	Communication beginning. Seek for a common sequence number,
FIN	Sender ends (FINish) communication.

FIG. 1-13: TCP header flags field.

The Window size is the number of bytes from the number specified in the field of confirmation, that the receiver is ready to accept. The maximum size is of (2^16) 65535 bytes. In this way, the protocol TCP permits the regulation of the flow of data between the transmitter and the receiver.

The checksum of a TCP segment is like UDP or IP checksum. Its function is to control the possible errors that happen in the transmission. This checksum includes the TCP header and the data. In case of an error, the datagram/segment is discarded and the protocol is the representative to assure the retransmission of the lost or erroneous segments.

The urgent pointer is valid if the flag URG is activated alone. It consists of a positive value that must be added to the number of sequence by specifying a position advanced where we can send urgent data.

The options field allow us to specify optionally extra characteristics to communication. An example of the options is MSS (Maximum Segment Size), the maximum size of data that the transmitter wishes to receive [Ric98-1]. This option is indicated at the beginning of communication (flag SYN activated).

The data field are optional. This means that we can send simply TCP headers with several options. This characteristic is used on beginning of communication (establishment) or in the sending of confirmations. In this way, we diminish the overhead since we only send/we receive what is necessary to establish or confirm communication.

1.5.1 TCP connection establishment

TCP is a protocol designed to connection. This implies that a previous step has to be made before being able to exchange data. This step is that of establishment of connection.

This step is essential to be able to guarantee the reliability of the protocol, since it is in this previous step where the obtained number of sequence will permit managing of communication. The method chosen to establish the connection is named protocol of 3 steps (*three way handshake*, see figure 1-14).

In this scheme we can differentiate an active client that begins the connection (it activates open) and a passive server that only answers (passive open) to the requests of connection. The three steps developed in the request of connection are:

1. The client (computer that wishes to begin communication) selects an uncertain number of sequence (x in figure 1-14). Activates the SYN flag in the options field and sends finally a segment TCP to the computer destination.

2. The server (computer with which is wanted to establish a communication) receives the request and stores the number of sequence x. It chooses an uncertain number (y in the figure 1-14) that will use as a number of sequence and activates the flags SYN and ACK.

   Finally it sends a segment with the chosen number of sequence and a confirmation of the value received plus one (ACK x+1 in figure 1-14).

3. The client stores the number of sequence (y in figure 1-14). It activates the flag of ACK and finally sends a confirmation of the number received plus one (ACK y+1 in figure 1-14).

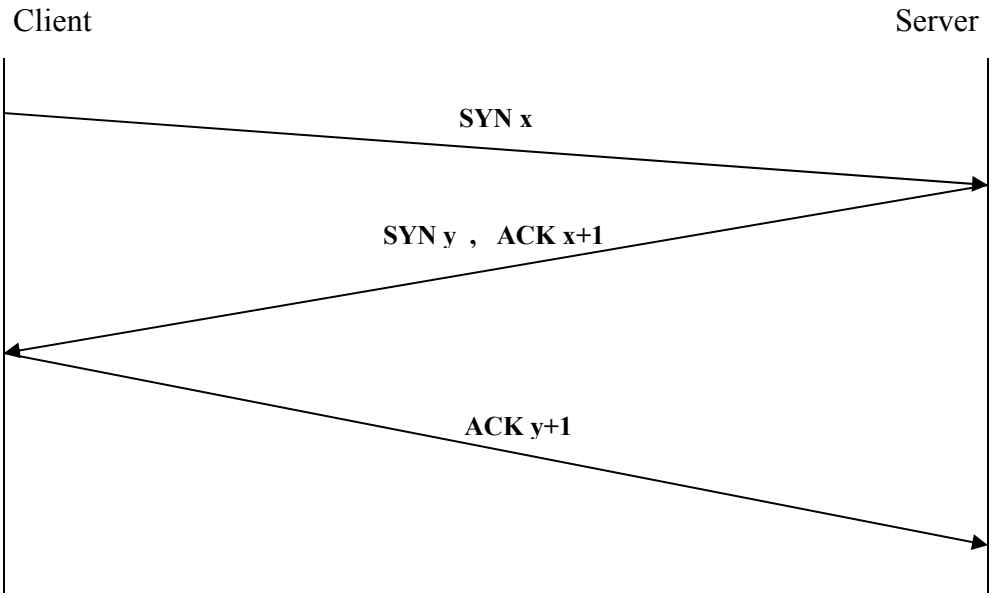

FIG. 1-14: TCP *three way handshake* protocol.

In case this third step is not made, after a certain time (between 70 and 130 second depending of operative system) the connection will be freed.

This characteristic happens because TCP connection is initialized it also initializes a timer. When it finish (time-out) allows us to free the connection and the associated resources.

1.5.2 TCP connection ending

As we have seen at the previous point, we need three actions to begin a TCP connection. To finish a connection four actions are needed due to the fact that communication is full duplex (the data may be sent and/or received independently and in any direction of communication). This obliges that each sense of communication must be finished independently (see figure 1-15).

Due to the possibility that any of both ends implied in communication may send and/or receive data, we have the chance that any of both ends finishes communication sending an END signal towards the other end point. This obliges us to make two half-closes one towards each sense.

TCP provides the capacity for both ends of the connection to finish its data output allowing it still to receive data of the other end. This capacity is named half-close. In the case of finishing only one direction of communication (client to server for example), the client can continue to receive data of the server sending only recognition of data (acknowledge, ACK). In this way when the server sends a signal of END the TCP connection it will be totally finished.

The sent of an END signal indicates only that more exchange of data in that direction will not happen. In this way it is possible that a TCP connection keep on sending data after receiving a signal END. This possibility is very little used in the applications that use TCP currently.

In a half-close process we can differentiate clearly an active end (it activates close) and a passive end (passive close).

The active end is the one that sends the END signal to finish that direction of communication, while the passive end accept the request and only sends a recognition (acknowledge) data.

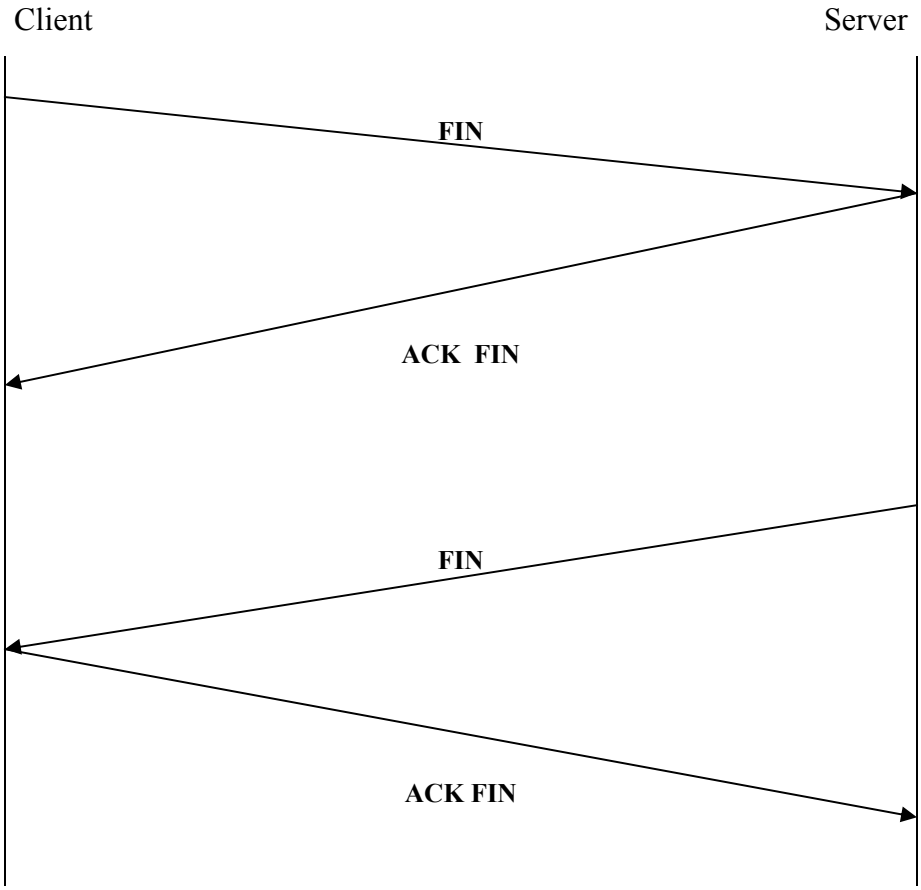

FIG. 1-15: TCP full closing schema.

In this case for finishing a TCP connection we must finish each one of the both directions. This obliges that alternatively the client and the server adopt the active and passive roles in a process that is made up from 4 phases:

1. (*The client adopts the active role*) the client decides to finish communication in his direction. Sending the server a signal of ending (END) with a sequence number.

2. (*The server adopts the passive role*) the server receives the latter signal and answers with a recognition (acknowledge, ACK) signal. Sending the number of sequence received plus one.

   It is necessary to indicate that the ending (END) like the signal of request of connection (SYN) consumes a number of sequence. This is the ending of the first half-close.

3. (*The server adopts an active role*) the server decides to finish the connection in his direction and sends a signal of ending (END) of connection to the client.

4. (*The client adopts a passive role*) the client accepts the request to finish the connection answering with an ACK and sending the number of sequence received plus one. This is the ending of the second half-close and the ending of the TCP connection.

1.6 Datagram routing

The routing process refers to how the different IP datagrams circulate and select the way towards their destination.

Due to the construction of Internet and its fault tolerant scheme, communications ways are not fixed or static. There are different ways for a package to reach its destination without the obligation of using the same route always. In this way, we have that each package is directed independently towards its destination.

The computers entrusted to receive and distribute the datagrams towards its destination are named routers. The latter routers take charge of reading the destination address of the datagram and selecting their way of exit.

In figure 1-16 we can see the guidance process, where a router receives an IP datagram and it leads until the following router.

We can see how this communication happens through unit jumps (hops) of router to router until it reaches its destination or the package is discarded (either because a route to the destination of the information does not exist, or because the life time or TTL has been consumed).

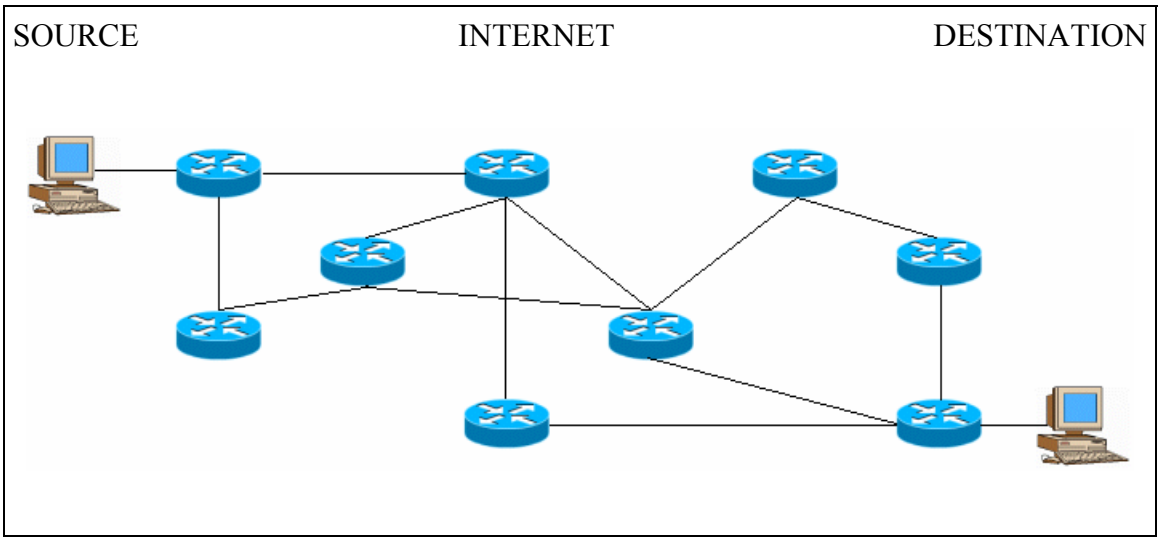

FIG. 1-16: Internet routing schema.

Once the datagram arrives to the final router, it is propagated by the local network until its destination. These process is achieved in a different way because in local area network (LAN) the addressing schema is based on MAC address (Medium Access Control) [WWW25][WWW26][WWW32].

The MAC address is unique for each device of network and is printed in the hardware of the network card. It consists of a 48 bits number that specifies the manufacturer and a unique serial number (see figure 1-17).

This address it is usually expressed as 12 hexadecimal digits (0- f) that they are divided into 24 bits (6 hexadecimal digits) for the number of organisation OUI (Organization Unique Identifier) and 24 bits for the address of number of series.

Once the final router receives the IP package for a computer that is connected in its local network (LAN), it makes a request to the whole local network so that the correct computer can be identified to be able to send the datagram IP to it.

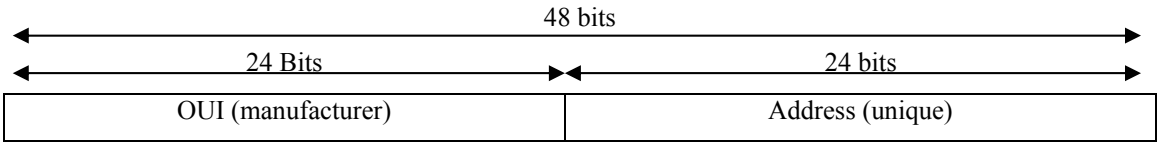

FIG. 1-17: MAC address schema.

To facilitate this process there are two protocols for the obtaining of the MAC addresses of a given IP address (**ARP**) and another for the obtaining of the IP address of a given MAC address (**RARP**) [WWW12][WWW13][RFC903].

- **ARP** (Address Resolution Protocol)[WWW27]: Obtaining of the MAC address of a given IP address. On receiving a package IP for X.Y.Z.T, the router asks the whole network (via broadcast) what is the MAC for this IP address?

- **RARP** (Reverse ARP)[WWW28]: Obtaining of the IP address corresponding to a given MAC address. Used for example in the machines that use DCHP for the obtaining of an automatic IP address.

The use of the protocols ARP and RARP propagates messages to all the computers connected in the local network, since the address must be identified among all the possible candidates. The computers that they do not answer are unaware of these requests.

In the IP protocol a way to identifying the network exists too. With a given IP address we must simply replace the bits of the mask of network by zeros (see figure 1-3).

Similarly, IP networks also has a broadcasting system [RFC919] that permits simultaneous communication with all the computers of the same network, we have simply to replace the bits of the mask of network by ones (see figure 1-3).

1.7 Summary

In this chapter a presentation of the TCP/IP protocols family has been made. This study focuses on version 4.

IP (*Internet Protocol*), that is a non reliable protocol and not connection oriented. Its main function is to provide a data transport mechanism by Internet.

UDP (*User Datagram Protocol*) is a simple protocol datagram oriented that uses the services provided by the IP protocol.

ICMP (*Internet Control Message Protocol*) is an encapsulated that takes care of the control of the flow of communication and the errors that they may occur.

TCP (*Transmission Control Protocol*) is a reliable protocol, connection oriented and focused on byte exchange that uses services of the protocol IP. This protocol establishes a previous connection with the destination (through the three way handshake) that allows it to regulate the flow of bytes sent and its correct reception by resending the incorrectly received data or lost packets. It needs a finally disconnection phase of the receiver (through the half close system) before giving up a finished communication.

Also the routing process of datagrams into the Internet has been commented. From the sender to the receiver, datagrams travel through Internet until they reach their final destination. When the local network (LAN) is reached the router uses different protocols (ARP and RARP) to deliver the data to the final destination.