Open networks

When users create a network

Enric Peig Olivé

PID_00164585

Universitat Oberta de Catalunya
www.uoc.edu

Index

1. Internet protocols: IP, TCP and applications

The Internet mainly revolves around two protocols: IP and TCP. To be more precise, the Internet uses four layers, IP and TCP being the main ones:

- **Application layer**. This is where applications are located and is the part that is visible to the user. Applications communicate with each other over the Internet. It uses the services provided by the transport layer to ensure that information flows correctly between the sender and the receiver. Here we can also find mail clients and servers, web browsers, file transfer programs etc. This layer only has meaning at the terminal level. It has nothing to do with the interconnection of computers.

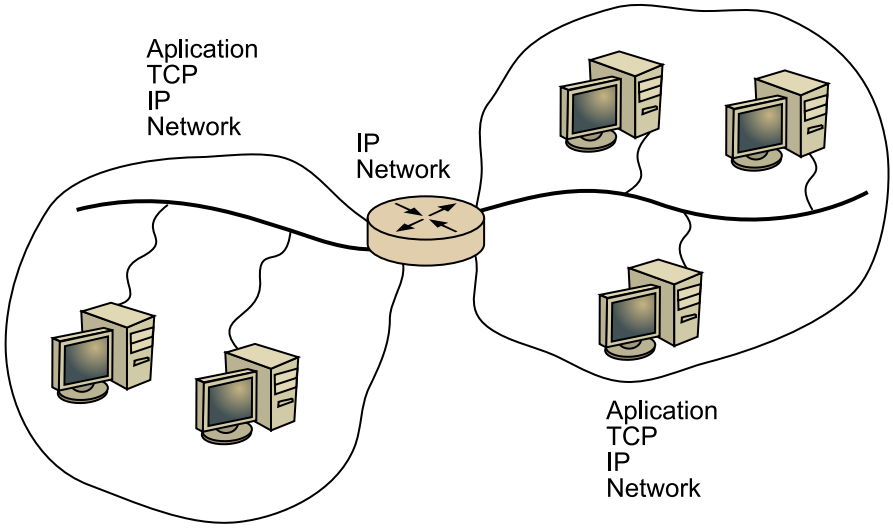
- **Transport layer**. This is in charge of sending data packets to their destination without errors. It is therefore responsible for ensuring that they arrive (they may be lost), and that they arrive correctly (they may have errors). As with the application layer, the transport layer only has meaning for the terminals connected to the Internet lying at each end of the communications channel. It has no meaning for intermediary equipment.

- **IP layer**[1] (network connection). This is what makes the Internet a single unit. It is handled by intermediary equipment (routers) and allows local networks to be connected to form a global network.

- **Network layer**. Everything included under IP. In other words, local networks containing computers which can be clients or servers for Internet applications. If we want to connect a local network to the Internet, we will need to use a router.

(1) This is where we get the term Internet: Internetworking or network interconnection.

The following figure illustrates the four layers:



In this unit, we will take a brief look at the concepts surrounding these layers and the protocols they include.

1.1. Network: IP (Internet Protocol)

IP is a datagram-oriented network connection protocol. It therefore does not use the concept of the virtual circuit so it is unable to recover lost packets or to guarantee that packets arrive in the correct order, as packets may follow different pathways thereby having different amounts of delay, neither is it able to ensure that the rate of arrival is correct to allow the recipient to process the data correctly.

IP is a best effort networking protocol. I.e. it does what it can without any guarantees. When we are using a network, we obviously cannot always be content with the information arriving only when the network has been able to achieve this. This is the point at which the transport layer protocols become involved, as they are responsible for ensuring that the information arrives reliably enough.

The main task of the IP is packet routing. It therefore decides which route a packet should take at any time, assuming that there are multiple pathways between routing devices on the network.

The relevant aspects to the study of this protocol are therefore:

- Address assignment, how we know which is which.
- The IP packet format, how the information being transmitted is structured.
- Routing, or how the packets get from their origin to their destination.

Additional reading

A good description of the protocols used by the Internet can be found in **W. R. Stevens** (1994). *TCP/IP illustrated* (volume 1: "The protocols"). Wilmington: Addison-Wesley.

Versions

IP version 4 has been in use for a long time but in 1994 migration of the whole Internet to IPv6 was approved. IPv6 was a big improvement over version 4 in terms of the number of addresses available and other security and performance issues. The migration was planned to be complete within a reasonable time frame (absolutely all equipment on the Internet needs to be modified). However, this has not happened and only some core equipment works with IPv6. User terminals are still all working with version IPv4. It is for this reason that in this course we will be looking a version 4, the one used by all personal computers.

1.1.1. IP Addresses

Each device has a unique IP address. To be more precise, each address is unique for each of the IP network interfaces of each device. If a device has more than one network interface (a router for example), it will need an IP address for each.

IP addresses are 32 bits long (4 bytes).

To type an address, we write the four bytes separated by dots.

The addres 10100111010100111001100101100100 would be written as:

166.83.153.100

IP numeration follows a hierarchical philosophy. Each address is made up of two parts. One corresponds to the network on which the node resides, known as the *network identifier*, and the other corresponds to the node itself, the *node identifier*.

Making sure no two addresses are the same is done by an organisation called the Internet Network Information Center or InterNIC, which was specifically set up to perform this task. This body is currently responsible for assigning addresses to regional organisations. Addresses are assigned for groups or networks, not individually.

To begin with, they defined three types of network depending on the length of the network identifier. These are known as:

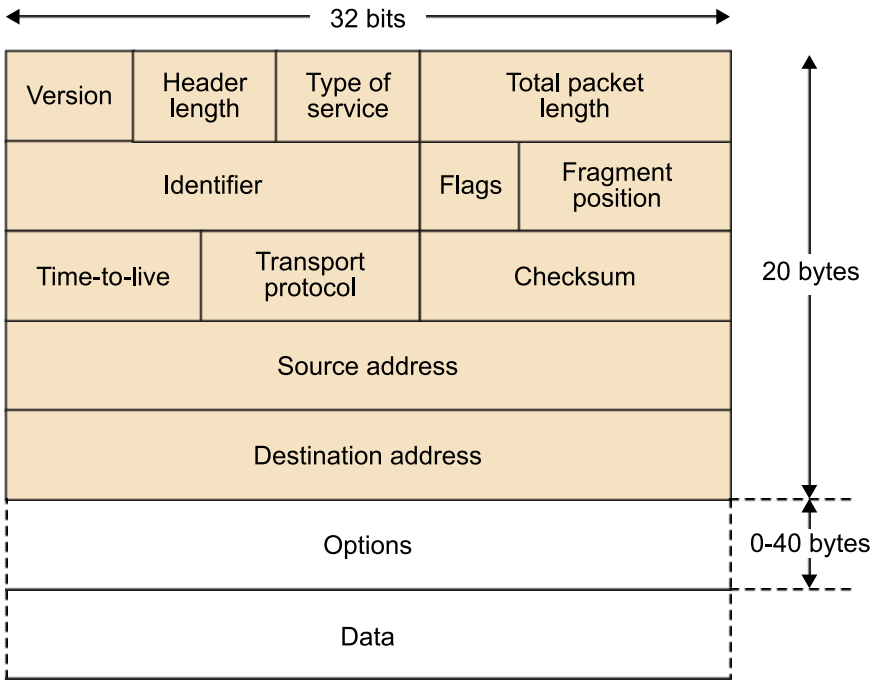
- Class A
- Class B
- Class C

In Class A, for example, the first 8 bits are used to identify the network and the other 8 to identify the node. This system fixes the number of Class A networks and the number of nodes that each network can have.

Later on, a more flexible system was proposed which was known as CIDR (Classless InterDomain Routing), this allowed the number of network bits and the number of node bits to be specified for each address.

1.1.2. IP packet format

IP packets are structured in the following way:



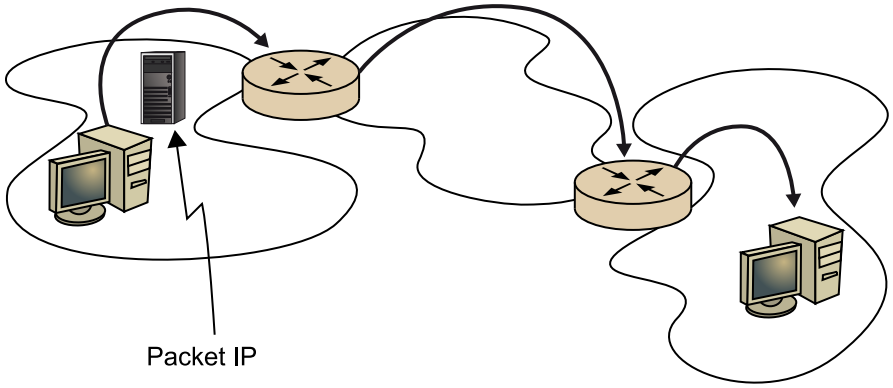
We can see that they have two parts: The header and the data being transported. IP packet headers have 12 fixed fields plus an option field usually used if options need to be specified:

- Version
- Header length
- Type of service
- Total packet length
- Packet identifier
- Flags
- Fragment position
- Time-to-live (TTL)
- Transport protocol
- Checksum
- Source address
- Destination address
- Options

1.1.3. Routing and routers

Every Internet router must be able to decide, in the shortest time possible, where any packet must be sent based only on the destination address, using the information it has about the network and its own position within the Internet. In fact, each router does not decide on the entire route of the packet, only the section of the route it is participating in, the next hop. We can have a varying number of hops between the source and the destination, each hop involving two routers (the transmitter and the receiver of the packet), except the

first and the last, which only involve a transmitter and a receiver respectively. Connections between routers, or rather between the router and the node, use LANs or point-to-point links.



Routing is done using routing tables that contain a limited but sufficient quantity of information to allow connections between all the networks on the Internet. Each device connected to an IP network requires a routing table.

Note

Protocols exist to allow routers to exchange information in order to keep them permanently up-to-date. If necessary, the table can be accessed using a terminal connected directly to the router.

The router decides which route the packet should take using its routing table. Node terminals also require a routing table, if only to find out whether they are communicating with a local node on the LAN (meaning they can communicate directly) or whether the IP packet needs to be sent to a remote node (connected to a different LAN) meaning it should be left in the hands of the router.

This would be the routing table corresponding to one of the nodes in the figure above:

	Address	Mask	Router	Interface
1	137.83.153.103	255.255.255.255	127.0.0.1	*Loopback*
2	127.0.0.0	255.0.0.0	127.0.0.1	*Loopback*
3	137.83.153.0	255.255.255.0	137.83.153.103	ether0
4	255.255.255.255	255.255.255.255	137.83.153.103	ether0
5	0.0.0.0	0.0.0.0	137.83.153.5	ether0

1.2. Transport: TCP and UDP

Applications make use of the network to send information between the two end points. But if the network cannot provide a reliable connection, we will need to find one that can. This is where the transport layer comes in.

A reliable connection is one in which the packets arrive correctly and in the correct order. This has a cost however, above all in terms of time, because if a packet does not arrive, it must be requested again and waited for. As we shall see later on, there are some applications in which this delay may not be desirable (moreover, the delay is variable, as not all situations are the same). The transport layer must therefore provide different levels of reliability and allow applications to choose between them.

In the case of TCP/IP, this is done using two protocols: One provides total reliability (the packets arrive correctly and in the right order) and the other only provides a simple error check. The first is known as TCP (Transmission Control Protocol) and the second is called UDP (User Datagram Protocol).

It is said that TCP is a connection-oriented protocol, while UDP is not connection-oriented.

Another important function of transport protocols is to identify the applications being used. TCP/IP communications are established between applications as well as devices. Therefore, to be able to communicate with a remote application, we need to know the application number as well as the IP address. This number is known as a *port*.

However, the port is defined at the transport level and not at the application level. It will therefore appear in the TCP (or UDP) header in the same way that the IP header contains an identifier for the transport protocol of that packet.

The port identifier has 16 bits, up to 65,536 different applications can be defined on the Internet. When the first applications were specified, it seemed appropriate to give them specific ports and it was decided that the first 1,024 ports out of the 65,536 possible ones be reserved for well-known applications – these ports therefore became known as the *well-known ports*.

A connection point at the application level is therefore identified by the IP/port address pair.

1.2.1. UDP

UDP is a connectionless protocol and therefore does not provide error or flow control, it only uses error checking mechanisms. If the UDP detects an error, it will not deliver the datagram to the application but rather it will discard it.

We must remember that the UDP is using the IP below it which is not a connection-oriented protocol either. A transport layer protocol was therefore designed to allow applications to exploit these types of characteristics and to make them as simple and as straightforward as possible.

See also

Later on, we will look at some of the real-time applications that take advantage of UDP.

It is the simplicity of the UDP that makes it ideal for applications that require low latency transmission (real-time applications for example). UDP is also ideal for systems that cannot implement a system as complicated as TCP.

Another interesting use of UDP is for applications that work in *multicast* or *broadcast* mode (they send information to a group of users or all the users on the network). In this case, they attempt to send information to a large number of recipients without waiting for a response, making it essential to use a simple, straightforward connectionless transport protocol such as UDP.

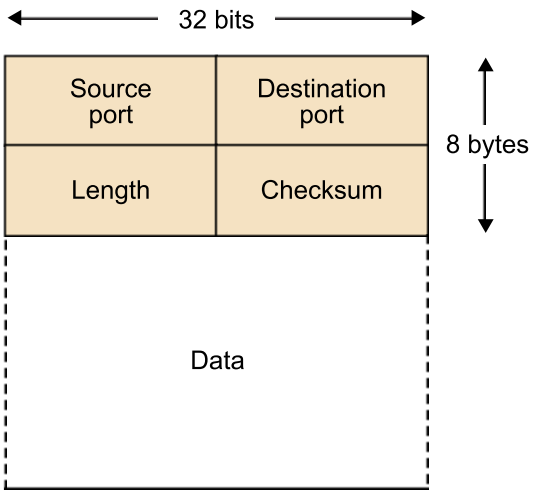
The most important features of UDP are the following:

- It does not guarantee reliability meaning that we do not have the security of knowing that every UDP datagram will reach its destination; it is a *best-effort* protocol. UDP will do everything it can to transfer the datagrams, but will not guarantee that they arrive.

- It does not preserve the sequence of the information provided by the application. As it is in datagram mode and uses a lower protocol such as IP, which is also in datagram mode, the application can receive the information out of sequence. The application must be correctly configured to deal with lost, delayed or out-of-sequence packets.

Format

A UDP datagram is structured into fields similarly to the IP packet:



The header only has four fields:

- Source port
- Destination port
- Packet length
- Checksum

1.2.2. TCP

As we have seen, UDP does not guarantee that information provided to it by an application will be delivered. Nor does it re-order the information if it arrives in a different sequence to that which it was transmitted in. There are some applications that do not tolerate these limitations. To overcome these, the transport layer provides a protocol known as TCP.

The TCP provides reliability and guarantees that all the information arrives in the sequence it was transmitted by the source application. In order to achieve this reliability, the TCP provides a connection-oriented service with error and flow control.

In order to provide the application with a reliable server, the TCP uses the following principles:

- Error free transmission. The TCP must deliver exactly the same information to the destination application that it received from the source application. It is, in fact, an "almost error-free" delivery system as there may be some errors which the TCP error-detection mechanisms cannot detect.

- Guaranteed information delivery. The TCP guarantees that all the information transmitted by the source application is delivered to the destination application. If this is not possible, TCP will notify the application.

- The transmission sequence is guaranteed. The TCP guarantees that the information stream arrives in the same order in which it was transmitted by the source application.

- Duplicate elimination. The TCP ensures that only one copy of a packet is delivered to the destination application. If more than one copy is received due to the functioning of the network or the protocols used below the transport level, the TCP will remove them.

The following properties of the TCP ensure that information is delivered reliably:

- The TCP is stream-oriented: The application organises the data into bit streams which are structured as bytes. The receiver then passes to its application the same stream of data that the TCP received from the source

application. Furthermore, the application has no way of indicating the data transfer limit to the TCP: The TCP decides which bytes to transfer in a segment at all times.

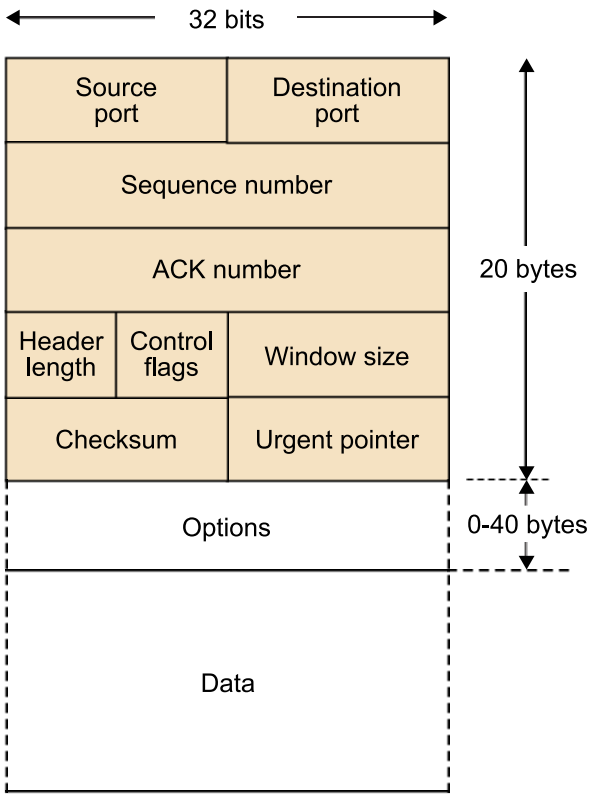
- TCP is a connection-oriented protocol: During the first stage, it establishes a connection, and then it passes through the data transmission stage and the disconnection stage.

- The TCP uses the buffered transfer concept: When the information is received, the TCP divides the data streams (bytes) into sections as it deems appropriate. The TCP will decide on the length of the segments whether the application generates a single byte of information or large streams. In the first case, the TCP can wait until the buffer memory is more full before transferring the information, or it can transfer it immediately (push mechanism). If the streams are very large, the TCP will divide the information into smaller segments before transmitting them.

- The TCP uses a full duplex connection: Information is transferred in all directions. The application will see two independent streams. If the application closes one of the streams, the connection becomes half-duplex. This means that one of the ports (the one that has not closed the connection) can continue sending information, while the other port (the one that has closed the connection) is limited to acknowledging the information. However, this situation would be unusual. Normally, if one port closes the connection, the other will too.

Format

The unit of information used by the TCP is called a ***TCP segment***. The format of the TCP segment is as follows:



TCP packet headers have 9 fixed fields plus an option field usually used if options need to be specified:

- Source port
- Destination port
- Sequence number
- ACK number
- Header length
- Control flags
- Window size
- Checksum
- Urgent pointer
- Options

1.3. Applications: WWW, e-mail, p2p...

Computer networks in general and the Internet in particular are the focus of a new concept in the world of programming: Distributed programming. Distributed programming aims to take advantage of the power and the resources of interconnected computers to carry out tasks in a cooperative fashion.

Additional reading

You can learn more about Internet protocols in D. Comer (1996). "Principles, protocols & architecture". In *Internetworking with TCP/IP* (vol. I): Hertfortshire: Prentice Hall.

A distributed application consists of various programs running on different computers and which communicate over the network that connects them.

It is important to note that each program can do nothing by itself. All of the computers will need to participate for useful results to be achieved.

The various parts of the code will cooperate through the use of a protocol. This protocol can be customised for each application or we can design a standard protocol that can be used by all applications in order to reduce the amount of the required design work. What is more, the existence of a standard protocol will ensure that applications from different manufacturers will be able to work together.

Several paradigms can be used to construct distributed applications but, in terms of the Internet, two are the most important: That which follows the client/server model and that which follows the peer-to-peer model.

1.3.1. Client/server applications

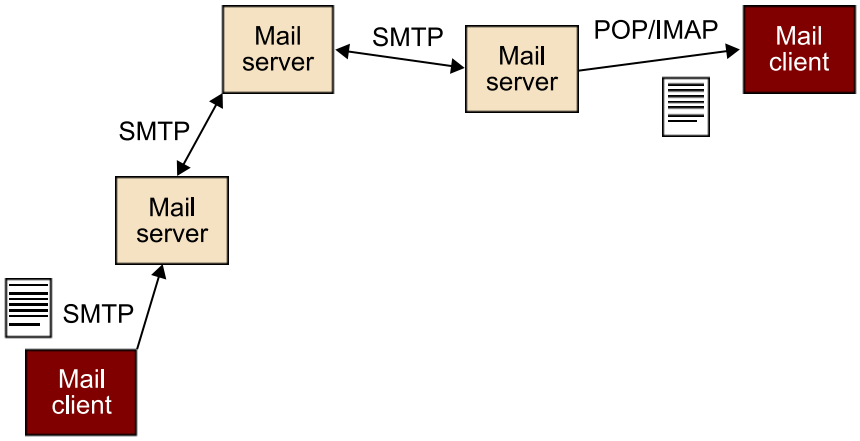
There are two types of components in the client/server model:

- Clients: make requests to the server. Clients will normally initiate communications with the server.

- Servers: provide services. Servers will normally wait to receive requests. Once they have received a request, they process it and return the result to the client.

The classic Internet applications that use this model are e-mail applications, web applications and file transfer applications:

- **E-mail** is based on the philosophy of storage and forwarding. The system consists of a series of servers that are in charge of passing on the mail they receive to their destination.



When we compose a message, it is stored in the server the e-mail client is connected to. The servers connect periodically and pass on messages that are then in turn retransmitted. This continues until they reach the domain of the destination user. The protocol which moves messages over

the Internet is called SMTP (Simple Mail Transfer Protocol). Once it arrives at the destination server, the user will be told that he has mail waiting, he can then download it to his email client using the POP (Post Office Protocol) or the IMAP (Internet Message Access Protocol).
Two of the most important aspects of the e-mail system are the format of the messages, based on RFC 822, and the format of the e-mail address (in the form user@domain).

- The **WWW** (World Wide Web) **service** provides access to structured multimedia content on pages; these are linked together using a hypertext system that allows pages or documents to be browsed in a non-linear and non-sequential fashion by following the links selected by the user and jumping to other parts of the document or to other documents.
  The web service is a good example of an Internet service in which the communication protocol between the client (the browser) and the server (web server) is extremely simple (HTTP – Hypertext Transfer Protocol) but in which the format of the information being represented is much more complicated (HTML, Hypertext Markup Language).
  HTTP defines several commands and their possible responses, which are then sent by the clients to the servers to ask for the content of a page or another resource to be sent.
  HTML is a language that allows us to describe the way in which a page is structured so that the client application (the browser) is able to display it to the user (a page of text, an image, making audio sound through the speakers etc.).

- The **FTP** (File Transfer Protocol) is one of the oldest on the Internet. In fact, it was one of the applications that originally made the Internet useful: it provides the ability to transfer files from one system to another. The first official specifications for the protocol were published in 1985 under RFC 959.
  The protocol allows files to be transferred both from the server to the client and from the client to the server, it also allows a client to order a server to transfer files directly to another server avoiding the need to copy files from the first server to the client in order to send them to the second server. It also provides functions for manipulating the server file system from the client. For example: the creation and deletion of directories, changing file names, deleting them etc.
  Another important characteristic of FTP is that it allows interoperability between different systems by hiding the details of the internal structures of file systems and the way they are organised.

1.3.2. p2p applications

In general terms, we can say that a p2p system is made up of a distributed system in which all the nodes have the same abilities and responsibilities and in which all communication is symmetrical..

See also

We will look again at Skype later on.

File exchange applications such as BitTorrent, eMule and the now-defunct Napster, or Internet telephone systems such as Skype would be the most obvious examples of peer-to-peer applications.

BitTorrent

BitTorrent is a peer-to-peer system used to distribute large volumes of data without the originator of the data having to pay the costs of the resources needed to house the content on a server. These types of solutions can be useful for distributing very popular content. BitTorrent uses servers to manage downloads. These servers store information about the file: Length, name, hashing information and the tracker URL. The tracker knows which peers possess the file (both totally and partially) and allows others to connect with them to download or upload files. When a node wishes to download a file, it sends a message to the tracker, which responds with a random list of nodes that are downloading the same file. BitTorrent cuts up the files into segments (of 256 kB) in order to know what each of the peers have. Each peer who is downloading the file tells the other peers the segments it has. The protocol provides mechanisms that penalise users who download information without then making it available to others. Therefore, when uploading information, a peer will choose another peer from which she has received information.

2. Open networks. When users create a network

The term "open network" can have different meanings in different environments. For the purposes of this course, we will be looking at the following concept:

> An open network is a network that can include any user and that any user can join as they wish.

A user might join an open network as a consumer or a client but also when they wish to be part of the infrastructure of the network or to provide services that may be useful to the other users.

"Open" in this context means the opposite of "private". Private networks are those which belong to companies or institutions who impose conditions and restrictions on joining them. Open networks do not belong to anybody because the devices they use are made available by the members of the community.

"Open" does not mean free. Users may charge a fee for making their resources available to the others (as long as the price is not exorbitant). The users therefore jointly contribute to the development of the network and provide the resources that can produce return, and they use it solely for their own purposes. The network comes about purely from the desire of the users to make it happen, there are no business strategies done by companies, which are usually subjected to market forces and applicable regulations.

Open?

The dilemma is very similar to that posed by the world of free programming: Open means that it is not private, not that it is free.

It is interesting to note that solidarity is a key social component of open networks, as each user is making his or her unused resources available to the rest of the community. The network can be configured using the participant's own devices, perhaps because they do not use all the bandwidth they have and can donate the excess.

In the same way that freeware uses licenses that set out the terms of use, licenses have also been created to establish the conditions under which users may participate in open networks, such as the Wireless Commons Manifesto, for example. On the one hand, this is a declaration of the reasons for the creation of open networks, but it also sets out the terms and conditions for creating and participating in these types of network.

Note

Lately we have seen the appearance of open networks provided by communications companies and local and regional governments in an attempt to achieve full public connec-

tivity. In these cases, the concept of "open" does not refer to the ability of users to form a network. It merely attempts to underline the independence of the network with respect to the services it provides. These are interesting ideas, but we shall not be dealing with them on this course.

In this regard, we should mention hot-spots, internet access points through a local Wi-Fi network provided to the general public either on a free or a paying basis. The idea is spreading fast in big cities and public places such as libraries, civic centres, parks, beaches etc. are being fitted out with Wi-Fi, as well as private businesses such as bars, cafes, and recreation areas, so the public now almost has continuous and ubiquitous access to the Internet. The philosophy of hot-spots does not really fall under the scope of this course. They are not considered to be open networks because the user is merely a consumer and has no control or responsibility for the development of the network.

Open networks can use both cable and wireless infrastructures. But, in practice, it is very difficult to create an open network using a cable infrastructure. The cost of the equipment and the problems associated with laying cables in public places make it virtually impossible. Wireless technologies, on the other hand, have limitless potential. The cost of the equipment is reasonable and some bands within the radio spectrum are considered to be free and commonly owned throughout the world. These bands lie in the 2.4 GHz and 5 GHz ranges.

The 2.4 GHz band and the 5 GHz band

The 2.4 GHz band belongs to a group called ISM (Industrial, Scientific and Medical). They are a set of frequencies that are reserved for the use of short-range radio equipment.

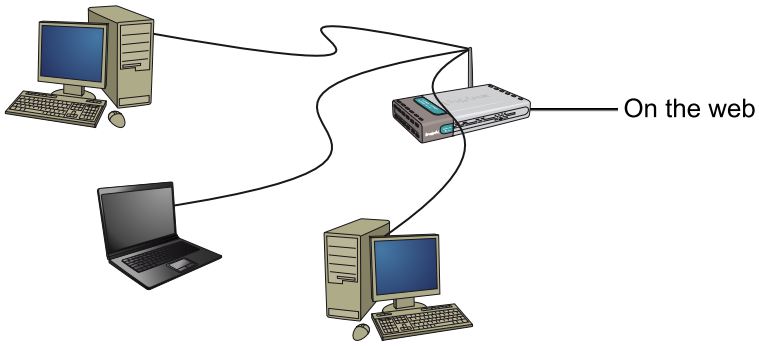
The 5 GHz band belongs to the UNII group (Unlicensed National Information Infrastructure), used for data connections.

Both are freely available to the public, meaning that their use does not require permission from the regulatory authorities (CMT in Spain). However, they do have power and range limitations.

It is because of this access to resources that the term "open network" is often applied to Wi-Fi networks that have been set up in order to serve the community in a cooperative spirit.

The key to setting up an open Wi-Fi network, as we have said, is to use devices called access points (or AP) as routers for circulating packets around the network.

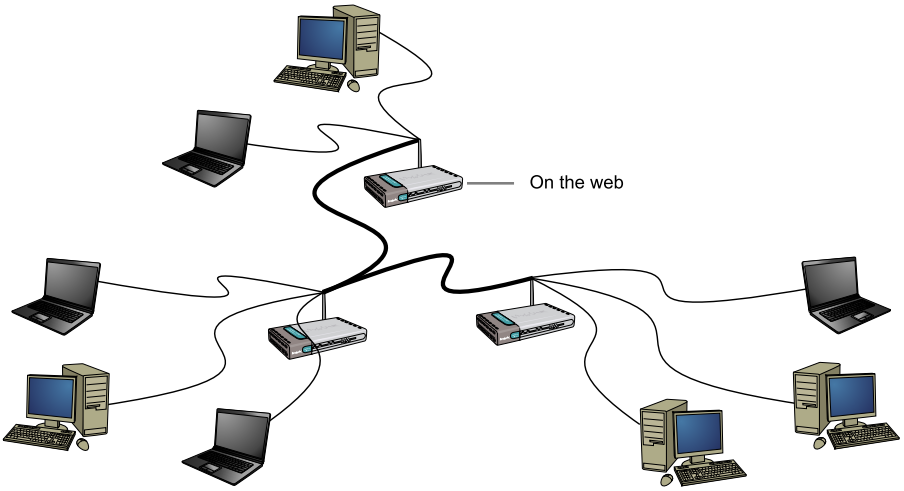
Access points are usually used to provide local area network infrastructure using mobile devices within a confined area (as a hub), and to provide access to cable networks (typically the Internet) if the device has router functionality.



However, if the access point has a good aerial, we can make them communicate with other access points located at a reasonable distance, and if we give them router functionality, we now have a long-range network that goes beyond the LAN, and it is open.

Remember

The band used has power restrictions. Access points often need to be closer than is convenient.



However, as we have already stated, for it to be an open network, it must comply with one basic premise: Any user must be able to join it and contribute.

One classic example of an open and free network is the one that has been created in Catalonia and that has a strong presence in the region of Osona. It is called guifi.net and has been studied around the world. It has created its own manifesto, inspired by the Wireless Commons Manifesto, called *Comuns Sense Fils* (Wireless Commons).

2.1. Routing layouts

The network shown in figure above shows what is known as a *mesh network*, so-named due to the large number of connections between nodes. When many of the devices are mobile, as is the case here, it is called a MANET (mobile ad-hoc network).

A mobile ad-hoc network (MANET) is a set of nodes that communicate with each other through radio links. Each mobile device on the network is free to move within the area, meaning that the network must be reconfigured autonomously and automatically. Each node must behave like a router and pass on the data it receives that is not destined for that device.

These are also known as multi-hop networks, as packets will normally need to make more than one hop to arrive at their destination.

The idea of ad-hoc networks being created thanks to the proliferation of communications devices is giving rise to the appearance of new applications. One example of these are spontaneous social networks. They take advantage of the fact that it is becoming more and more common for mobile phones to have Wi-Fi connectivity. The idea is based on the use of software at a node to detect the presence of mobiles having the same software and to automatically create a network. The possible uses are innumerable: Exchanging messages, files, chatting, streaming etc. We can configure public and private groups and select the level at which groups can participate, as well as the information that is available for sharing. The network requires no special infrastructure because it uses the concept of the MANET: The network is created by the devices because they all have routing capabilities.

Miraveo

Miraveo is a good example of a platform that allows us to create a network of this type.

2.1.1. Routing

The main challenge associated with designing an ad-hoc mobile network is: how to continuously update the information needed for traffic to be routed in the best possible way at all times. With cable networks, updating does not need to be done so often because the working conditions are more stable and changes are often caused by errors or faulty equipment, whereas in the case of mobile networks, whenever one of the nodes on the network moves, which is very often, the network changes, and therefore requires reconfiguration.

Additional reading

E. Royer, C. Toh (April 1999). "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks". *IEEE Personal Communications*, volume 6, issue 2.

This article takes an in-depth look at the routing protocols used in mobile networks.

The huge growth in the use of laptops and Wi-Fi networks has meant that a lot of research has been undertaken in this field and the search for efficient and stable routing protocols began in the middle of the nineties.

A routing protocol is a standard which allows nodes to decide which path the packets they receive should follow. In ad-hoc networks, the nodes are unaware of the topology of the network when they are turned on. They will need to discover it. They do this by announcing themselves and waiting for responses from other nodes. Every node then knows who its neighbours are and how to get to them.

Routing protocols are basically classified into two groups: They can either be proactive or reactive depending on the method used to calculate the routing tables for the devices on the network:

- With **proactive protocols**, routing information is constantly being sent, whether or not there is any data that needs transferring. This strategy provides very fast response times when a new route needs to be established, it also provides good performance when the devices that make up the network are highly mobile. However, the large amount of routing information that is inserted into the network can cause overloading.

- **Reactive protocols** only search for a route when needed by one of the nodes. Whenever it has to initiate communications between two nodes for which it has no routing information, it will send a route discovery message. When a response is received, the newly discovered route is added to the routing tables and communication can take place. The time taken to discover a route can be fairly long when compared to proactive strategies, but better use is made of the resources of the network.

The use of a proactive or reactive protocol will largely depend on the mobility of the nodes, the likelihood of a need to discover new routes and the level to which the network is overloaded.

New protocols are now being proposed which incorporate characteristics of both these types. They are therefore known as hybrids. In fact, the intense activity taking place in this research field is constantly leading to the creation of new proposals for routing protocols and variations on existing ones.

A couple of examples of routing protocols are: OLSR (Optimised Link State Routing Protocol) is one of the most widely used of the proactive protocols, and BATMAN (Better Approach To Mobile Adhoc Networking), which appeared fairly recently and was developed by Freifunk, a German initiative which supports the development of tools for open networks.

2.2. Wireless networks

As we have already mentioned, open networks use wireless technology due to its ease of implementation and low cost, but above all for control of the equipment used. We have also mentioned that the basic element of a wireless network is a device known as an access point, which will normally incorporate hub functionalities to allow the creation of the network and routing capabilities to connect the local wireless network to a longer range network (typically, the Internet).

The standard that covers all specifications, standards and technical details relating to wireless networks is IEEE 802.11. We often talk of the 802.11 family, the first version of which appeared in 1997, which has constantly incorporat-

ed new protocols with new functionalities, new changes, new additions and variations etc. The following table shows some of these protocols and their main specifications:

Protocol 802.11	Release	Frequency	Transmission speed	Interior range (m)	Exterior range (m)
-		2.4 GHz	Up to 2 Mbits/s	20	100
a		5 GHz	Up to 54 Mbits/s	35	120
b		2.4 GHz	Up to 11 Mbits/s	38	140
g		2.4 GHz	Up to 54 Mbits/s	38	140
n		2.4 / 5 GHz	Up to 600 Mbits/s	70	250

Wi-Fi devices will usually be compatible with several standards. At the moment, the most common type of device uses IEEE 802.11b/g. Compatibility is usually desirable.

During the development of the standard, the Wi-Fi Alliance organisation was created. This is a global non-profit association of companies dedicated to wireless networking: manufacturers, service providers etc. Its main goal is to promote the use of Wi-Fi technology and ensure that the devices used are interoperable and of high quality. To do so, they award the "Wi-Fi Certified" seal.

We should state that both the Wi-Fi alliance and the scientific committee of the IEEE, responsible for the 802.11 protocol, are working towards common goals: the development and use of Wi-Fi technologies.

2.2.1. Wireless sensor networks

Wireless networks can also be used to monitor physical and environmental phenomena using sensors. These are known as Wireless Sensor Networks (WSN). The fact that they are wireless allows them to be implemented in hostile environments where cable laying would be difficult (they in fact have military origins, but are becoming increasingly common in civil applications, industrial applications, home automation and medicine etc.).

There are several hardware and software standards in use for the implementation of sensor networks including proprietary architectures, but since the release of the IEEE 802.15.4-2006 standard, most companies are converging towards it.

2.2.2. Access points

There are many manufacturers producing wireless network equipment. But there is one that dominates in the world of open networks: Lynksis. The reason for this is clear: the firmware for their access points is based on the Linux operating system, which allows users to modify, adapt and expand it as they wish. Users will obviously need to have sufficient knowledge to do this. But the company provides support by publishing all kinds of useful technical information. This is the working philosophy within the context of open networks.

Linksys

Linksys was founded in 1988 and was bought out by Cisco systems in 2003. Cisco has maintained the identity of the business line and their devices are sold under the name "Linksys by Cisco".

The main open network unit produced by Linksys is the WRT54G access point.

But it is not the only one. We have a family of devices with differing performances and costs.

2.2.3. Firmware

The Linksys WRT54G is firmly associated with OpenWrt. This is a Linux product that is embedded in devices, especially networking devices. The first version was actually only produced for WRT54G access points, but the rapid growth of wireless networks has meant that other companies have also developed firmware based on Linksys which is fully accessible to users. OpenWrt has incorporated all of these. There are currently more that thirty manufacturers supporting OpenWrt.

OpenWrt provides everything needed to create a local wireless network and a MANET, and therefore everything needed to create a true open network in the participative sense, one that is free and community-based.

2.3. Basics on wireless network security

Security is even more important for wireless networks than it is for cable networks. This is due to the fact that traffic is easier to intercept. For a long time, the Internet had no form of protection as all connections were made by cable. But as it grew and became more popular, the need arose for mechanisms that provide confidentiality and the ability for authentication etc. However, many non-secure connections still exist.

Wireless networks have appeared at a time when there were many security worries, and this led developers to include standard protocols from the outset to provide basic information security and integrity and to allow authentication of users.

When studying these protocols, we also need to remember that security in the wired Internet is only applied from point to point, whereas with wireless networks security needs to be implemented for the wireless link between the node and the access point.

The first attempt was WEP[2] (Wired Equivalency Privacy) which, as the name indicates, attempts to give wireless connections the same privacy as wired ones. The specifications of WEP were included in the IEEE 802.11 standard in 1999. Either due to the short length of passwords or design weaknesses, WEP was soon replaced by WPA (Wi-Fi Protected Access). WPA was far more robust, but was declared obsolete in 2004. However, it is still used because many devices use WPA by default and users who do not have much knowledge about security are not aware of its weaknesses.

[(2)] WEP is often mistakenly interpreted as Wireless Encryption Protocol.

2.3.1. WEP

In terms of integrity, it uses CRC32, or Cyclical Redundancy Checking, widely used in communications protocols. Encryption is done using the RC4 algorithm and two password systems have been proposed for authentication, the shared password and the open system. We will now look at these algorithms in more detail.

The WEP standard provides integrity, encryption and identity authentication.

The CRC32 algorithm generates a binary number based on the message being transmitted. The calculation is repeated at the destination, and if the result is the same as the number received, then the message has not been changed. However, if the calculated number is different from the one received, this will mean that the original message has been altered during transmission (either deliberately or due to errors occurring in propagation). The way in which the

binary number is calculated will mean that a small alteration of the message will produce a large alteration in the number calculated. This prevents small changes from going unnoticed.

Encryption is done using RC4 (Rivest Cipher 4). This algorithm was designed by Ron Rivest at the RSA Security Company in 1987. It is still widely used despite its limitations. This is a classic symmetrical encryption algorithm. This means that it uses the same key for both encryption and decryption, meaning that both the transmitter and the receiver need to know it. If the key were always the same, it could be easily discovered and the contents of the message decrypted. To prevent this, the password is divided into two parts: one fixed and the other changeable. The first version of WEP used a 64 bit encryption key: 40 bits for the fixed part, the WEP key itself, and 24 for the variable part. The variable part is known as the initialisation vector.

RC4

The first version used 64 bits due to the restrictions imposed by the US government on the export of encryption technologies at that time. When the restrictions were abolished, it was later raised to 128 bits (104 for the WEP password and 24 for the initialisation vector), and some devices currently use 256 bits (232+24).

The WEP key is provided by the access point devices and must be entered in the computers we wish to use for communication. To make this easier, it is normally expressed using 10 hexadecimal characters (10 characters x 4 bits per character make up the 40 bits of the key).

Authentication can be done using the open system and shared password methods. In fact, in an open system no authentication is needed. Anyone can associate their terminal to the access point. Once association has been accepted by the access point, the WEP key is used to encrypt the transmission. When using the shared password method, clients must demonstrate that they know the WEP password. The following four steps are followed to do this:

Note

Access points can be configured in open mode: Communications are not encrypted. A WEP key is not needed for association with one of these access points.

- The client sends an authentication request
- The access point returns a text (known as a challenge)
- The client encrypts the text using the WEP key and returns it
- The access point decrypts the text received and, if it is the same as the one sent, authentication is confirmed. If not, access is denied.

From this point on, information can be exchanged, whether it is encrypted or not.

At first sight, it may seem that the first method is much less secure than the second. However, it should be noted that if the packets exchanged during the four steps of the second method are intercepted, the key can easily be recovered. As no information is exchanged during the first authentication process, the key is never exposed to interception.

At any rate, nowadays WEP is no longer able to provide security. There are many widely available tools that can decipher a WEP key in seconds thus allowing a device to associate itself with an access point for which it does not know the key. The only purpose it serves now is to demonstrate that a network is private and that it is not authorised for use by others.

2.3.2. WPA/WPA2

WPA was designed to replace WEP due its shortcomings, but it was a new version called WPA2 that was incorporated into the 802.11i revision of 2004. It is said that WPA is the Wi-Fi Alliance version and WPA2 the IEEE standard version, but actually there are substantial differences between them.

WPA still used RC4, but meant to replace CRC32 with a more robust algorithm known as Michael, and to use a key management system based on a RADIUS authentication server (*Remote Authentication Dial-In User Server*) and the TKIP protocol (Temporal Key Integrity Protocol). With WPA2, RC4 was to be replaced with AES-128 (Advanced Encryption Standard) which is much more robust, and it also includes new authentication mechanisms such as EAP (Extensible Authentication Protocol). So that authentication servers do not need to be used in small networks (domestic and small business networks), it kept a shared key mode that is similar to the one used in WEP, called PSK (Pre-Shared Key). In this mode, we will need to manually enter the key into all the devices on the network.

AES-128

AES-128 will be incorporated to comply with the security requirements imposed by the US government in FIPS140-2.

In terms of backward compatibility, more recent devices (including access points, terminals, laptops, network cards etc.) can use all three security types (WEP, WPA and WPA2), though there are still many older devices in use that do not include WPA2 or only include WEP. This means that correctly configuring the security of a wireless system can often be a headache.

It was not long before WPA was cracked. It presented vulnerabilities similar to those in WEP, mainly due to the security of the keys and the way they were managed. WPA2 is far harder to crack. It is therefore highly recommendable to configure all devices on the network to use this standard.

The following table shows a summary of the main characteristics of the three systems:

	WEP	WPA	WPA2
Year it appeared	1999	2003	2004
Integrity	CRC32	Michael	Michael
Encryption	RC4	RC4/TKIP	AES-128

	WEP	WPA	WPA2
Authentication	Open or shared key Shared key or RADIUS	Shared key or RADIUS	EAP

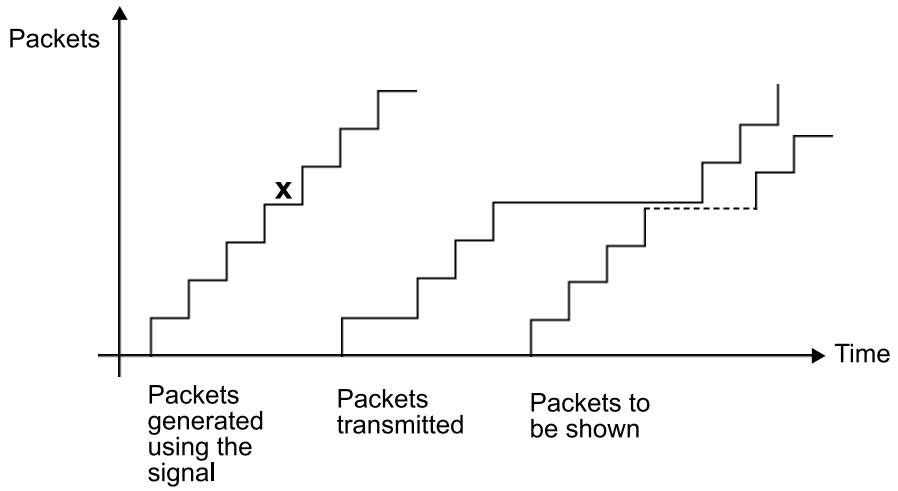
3. Audio and video streaming

3.1. Audio and video on a packet-based network

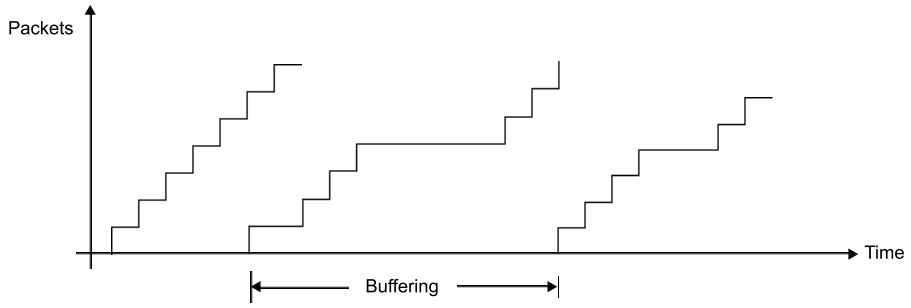
The Internet was not designed for the dissemination of audio and video. It is a network based on the transmission of packets, the protocols which sustain it (TCP, IP) ensure that the packets arrive at their destination but do nothing to make the packets arrive with a specific delay or even to ensure that this delay is the same for each packet. These two aspects, but above all the second, are crucial in audio and video streaming.

We are talking about the transmission of audio and video for immediate reproduction. If multimedia data is sent in a file that is to be saved for reproduction at a later time, the Internet poses no problems. It does not matter if packets are delayed or if this delay varies. From now on, we are only going to look at streaming audio and video that is to be reproduced immediately.

The first thing we can see is that, before we can send audio or video over the Internet, we will need to capture the images or the sound and convert them into packets. When the packets are received, the original images and sound are put back together so they can be displayed to the user. For this recreation to be faithful to the original, we will need the packets to arrive in the same order they were created, the delay must be at an acceptable level and the variability of this delay must be low.



In the figure, we can see that the excessive delay experienced by packet x forces the reproduction to stop until it arrives. One way of minimising this effect is to buffer the information received so that reproduction begins with a certain amount of delay, and this is decided beforehand by estimating the average delay that the packets may experience.



However, we can never be certain that a packet will not experience a longer delay than the one we have set, we are therefore presented with the same problem: reproduction will need to be stopped.

This problem meant that the Internet was not able to stream audio or video for a long time. The usual vehicles for doing this have been television (terrestrial and cable), radio and the telephone network.

Another important factor to bear in mind is bandwidth. The first internet connections were very slow (including the trunk lines, connections between devices and the connection to the user terminal). It became necessary to develop data compression formats that allowed the data to be recreated in real time.

But the challenge was there, and protocols have been developed and strategies put in place to allow the Internet to disseminate multimedia data without being affected by the problems described above. Applications have then been written to take advantage of these protocols. As connection bandwidth has grown, these techniques have become more popular, as well as the development and use of applications.

When we talk about audio and video streaming, we are referring to the transfer of multimedia information from a server to a client that is displayed to the user as soon as it arrives. This is done on the client side by *media players*. Depending on how the server stores this information, we can distinguish between stored content streaming and direct streaming. In the first of these, the multimedia data is stored in a file on a server while in the second it is reproduced immediately. From the point of view of transmission and the problems we are looking at, this difference is irrelevant.

There is also a third type of multimedia network communication: the one including interactivity. Information does not just flow in a single direction (from the server to the client), but in both directions so that a dialogue is

established. Here we are talking about real-time interactive communication and we will look at this in a different module. Here we are going to focus on unidirectional streaming.

3.1.1. Strategies beyond TCP

The IP is a best-effort protocol. This means that, while it does what it can to get packets to their destinations, there are no guarantees that they will arrive or that they will arrive in the right order. It is the TCP which is in charge of doing this and this acts above the IP.

There are several reasons for this separation of duties: The most important of these is that the IP acts on every device (terminal nodes and routers) while the TCP is a point-to-point protocol. It only acts on the terminal nodes. This means that the intermediate devices are simpler as they only have to implement IP.

Another reason for separating the transport layer from the network is that, in some cases, it can be useful to have a simpler transport. For some applications, it is not so critical that all packets arrive. This is the case here. If we are reproducing a video and a packet containing a few images is lost, they can not be retrieved. It will be too late when they arrive. What has happened, has happened, and there is no way around it. In this case, the loss of a packet is not critical.

The loss of a packet

This is obviously critical in the transfer of data files.

We can say the same for a packet that is out of sequence. If we have already shown the following images by the time it arrives, then it does not matter. It behaves as if the packet was not lost.

The same is also valid when a packet arrives with errors. In this case, the TCP will ask for it to be re-transmitted, and to ensure the packet order is correct, they will not be sent to the application layer until the incorrect packet has been received. This is also a problem. We would prefer to skip over the erroneous packet and continue playing the video or sound without having to freeze it.

Therefore, when transmitting audio and video that is to be reproduced as soon as it is received, we are more interested in regular reception than in security and the order of reception. In other words, we do not need the services of the TCP, so it is better to use UDP. This is a transport layer protocol, as is the TCP, but it does not establish a packet ordering mechanism nor are packets retransmitted if they are lost.

But even the UDP is not enough. It does not provide enough functionality. In this case, we need to revert to the application layer For example, we do not need to guarantee that the packets are received in the right order, but we do need to know the order. We have already stated that it is better to discard a

packet that arrives out of sequence than to display it in the wrong place but, to do this, we need to know the sequence. We have already stated that there is no point in requesting the retransmission of an erroneous packet. Instead, we can use a FEC technique (Forward Error Correction) to try to correct the error without requesting retransmission.

In summary, streaming applications generally use a combination of UDP + specialised application protocols instead of using TCP.

One of these protocols is called RTSP; we will now take a closer look at this.

3.2. Real Time Streaming Protocol (RTSP)

The RTSP is defined in RFC 2326, and it is used to establish and control synchronised streams of multimedia information.

The case we are looking at is that of the synchronised transmission of audio and video. The RTSP is not responsible for sending information. Its task is to control delivery. We could say that it remotely controls the transmitter. To transport the data, we could use UDP or RTP[3] (*). There are some applications that even have their own protocols

[(3)]Do not confuse RTSP with RTP. RTP is a protocol used to transport multimedia data, RTSP just issues the commands.

The basic idea is to get the server to send the client or clients small packets of multimedia information so that the content can be displayed as the packets are received without waiting for the whole file. This "live" transmission functionality is fundamental. In this case there is no file.

This synchronised multimedia stream is often called a presentation.

The RTSP allows streams to be transmitted in *unicast* mode and *multicast* mode. In the first of these, the data is transmitted from the source to a single destination. In the second, it is transmitted to multiple destinations.

RTSP is considered to be an application protocol, it is in charge of communicating requests, not data. It is therefore very similar to HTTP. The syntax of the requests and the functionality of the protocol are defined as per HTTP/1.1. It is equally extensive and has defined authentication mechanisms. You can use either TCP or UDP underneath it, however TCP is usually used.

Note

There are no problems associated with using different protocols to transport the data and issue the commands. These are independent of each other.

RTSP uses the concept of sessions instead of connections. Each session has its own identifier but is not linked to a single transport connection. In an RTSP session, we can open and close several transport connections with different protocols (TCP, UDP). Clients and servers pass through different states in each session, depending on the order in which they send and receive.

3.2.1. RTSP commands

The following commands are sent by clients and servers:

SETUP	Used to specify the parameters of the data transport mechanism. If it is sent during a session, it is used to change the parameters.
PLAY	Used to tell the server to start sending data.
PAUSE	Used to tell the server to temporarily stop sending data.
RECORD	Used to initiate data recording in accordance with the description of the presentation.
TEARDOWN	Used to close the communication.
OPTIONS	Allows the client to request information about communications options from the server.
DESCRIBE	Recovers information about the multimedia content identified by the URL that is sent with the command.
ANNOUNC	When sent to the client, this is used to describe a presentation that has been recorded. If it is sent to the server, it is used to update the information in the session description in real time.
GET_PARAMETER	Asks for the value of a session parameter.
SET_PARAMETER	Changes the value of a session parameter.
REDIRECT	Tells the client to connect to a different server.

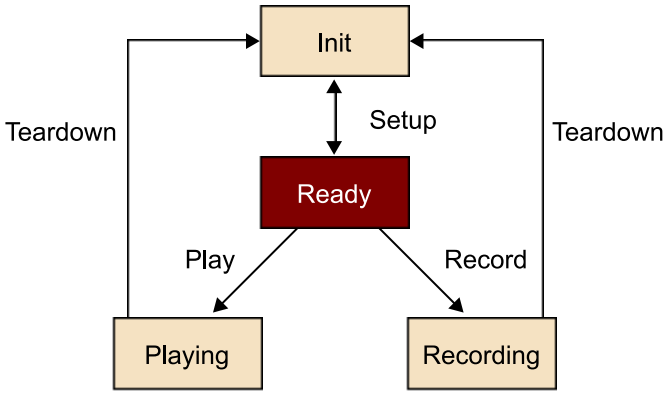
3.2.2. States diagram

Unlike HTTP, which has no states, both the client and the server can have different states within an RTSP session and they exchange commands or requests and the responses to those depending on the state they are in. Also unlike with RTSP, servers can also make requests to clients. Some of these requests can change the state of the participants in the communication, others do not.

The four states in which they can be:

- Init
- Ready
- Playing
- Recording

The commands SETUP, PLAY, PAUSE, RECORD and TEARDOWN can change the state of both the client and the server as is shown in the following figure:



The initial state is *Init*. When a SETUP is sent and a positive response is received, the client and the server pass to *Ready*. From here, they pass to *Playing* or *Recording* depending on whether a PLAY or a RECORD was sent (and a confirmation received). The PAUSE command returns the state to Ready, another PLAY or RECORD command will return the state to Playing or Recording. The TEARDOWN command always returns the state to *Init*.

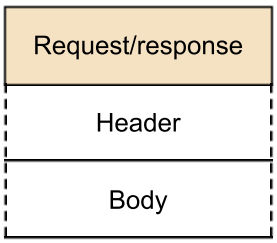
The client and server both follow the same states pattern. When one is in *Ready* the other will be too. Transitions occur when commands are confirmed, followed by one or the other.

3.2.3. Description of the protocol

The RTSP protocol describes the messages that contain the commands and their responses, as well as how they are exchanged. As we have already said, RTSP is very similar to HTTP. Therefore, its messages are also similar.

Messages are written in text and are formatted into three parts as shown in the following image:

- Request/response line
- Request/response header
- Body of the message



The request line has three fields separated by a space:

RTSP_command URL_request RTSP_version

SETUP rtsp://server.com:554/shop/program.rm RTSP/1.0

We should note that the protocol identifier is rtsp:// and the assigned port is 554.

Nota

If the UDP transport protocol is to be used instead of the TCP, the identifier will always be rtspu://.

The response state line also has three fields:

RTSP_version State_code State_descriptio

RTSP/1.0 200 OK

The request header includes several fields in the form:

Name_field: Value_field

There are general fields (which can be either requests or responses), specific request fields, specific response fields and fields related to the message body.

Content-Type: application/sdp

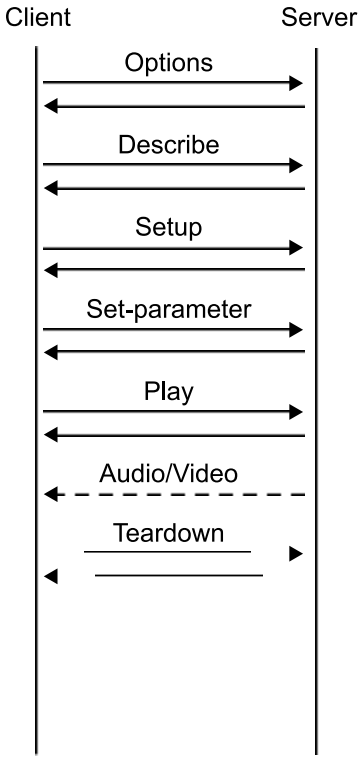
Session: 47112344

Transport: RTP/AVP;unicast

The body of the message carries information relating to the request or the response, if there is any.

Example of an RTSP dialogue

These would be the requests and responses of the RTSP protocol exchanged by a client and a server during normal interaction:



- The client begins the interaction by sending an OPTIONS request. The server responds with various pieces of information including the methods supported.
- The client sends a DESCRIBE request for certain content and the server answers it.
- The client sends a SETUP request for each stream included in the presentation indicating the correct protocols for each stream (for example, audio and video). The client then initialises the applications needed to reproduce the streams in the presentation.
- The client sends a SET_PARAMETER to set an acceptable bandwidth.
- Lastly, the client sends a PLAY command to begin data transmission.
- During the session, the client can verify that the connection is still open by sending the SET_PARAMETER command. Even if the server replies with an error, the client knows that the server is still connected.
- The client sends a TEARDOWN command to close the connection.

3.3. Content delivery networks

When streaming to a considerable number of users concurrently, we may find that the number of nodes the packets pass through will go up and often become overloaded. This is not usually a major problem in standard Internet applications, but it may cause a loss of quality in streaming due to delays, losses etc.

o be able to provide this service effectively, content delivery networks (CDN) have been developed. This is a network of servers that are usually located in privileged positions[4] within an ISP and which all contain duplicate copies of the content being provided. When a client is connected to the content

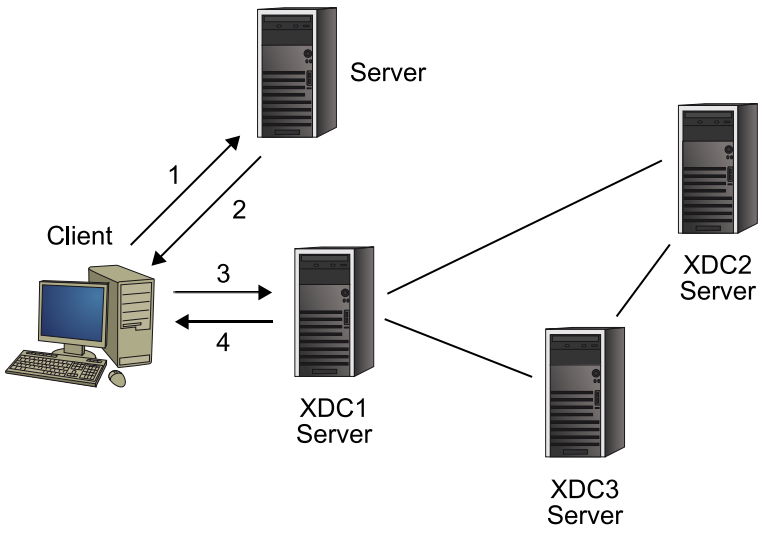
(4) In fact, the CDN nodes are installed in the ISP's data centres and connected directly to trunk lines.

provider's server, this switches the connection to the CDN node that is closest to the client. This optimises the transmission speed and therefore minimises streaming degradation.

Example

Akamai is a well-known example of a CDN.

The following figure illustrates how a CDN is organised:



In practice, CDNs are not just used for streaming multimedia content. The download speed alone is worth the cost associated with housing content on a CDN. Application server platforms are another of their large clients.

3.4. Examples of streaming servers and clients

The list of applications supporting multimedia streaming is long and includes both proprietary and free applications. Among these, we should mention, in terms of servers:

- VideoLAN
- Ampache
- Darwin Streaming Server
- Helix Community
- Icecast
- GNUMP3d

And in terms of clients:

- Amarok
- MPlayer
- Screamer Radio
- VLC Media Player
- XMMS
- Zinf

4. IP telephony

As we have sawn, the use of the Internet for transmitting multimedia content has required the development of new strategies, new protocols and new applications.

We have also looked at the differences between streaming applications and those that allow interactive communication in real time.

When talking about real-time interactive communications, we often hear the term "Voice over IP". This term (abbreviated as VoIP) refers to the group of techniques that allow voice signals to be transported over the Internet. It was created as an alternative to the conventional phone network for holding voice conversations and avoids the costs associated with the phone network, above all for long distance calls.

Additional reading

L. Harte (2006). *Introduction to IP Telephony: Why and How Companies are Upgrading Private Telephone Systems to use VoIP Services*. Althos.

When talking about new technologies in general, we should not confuse applications and protocols. The term *VoIP* is used to designate the concept itself. Transporting the human voice over the Internet. The term "IP Telephony" refers to the telephone service offered to users in general (including a number and other added value services). In terms of protocols, we should mention H.323, for example.

The conventional phone network was developed using the technology available at the beginning of the 20th century using dedicated circuits. This means that during conversation over the phone, a set of resources are designed specifically for it. For this reason, the traditional business model used by these services charges according to time of use and proportionally to the number of resources used. Thus, a long-distance call is a lot more expensive than a local call because more lines and more control boxes are used.

Charging by the minute

One of the arguments used to justify charging by the minute is the deterrent factor. If a person is aware that the longer they talk the more they will pay they are likely to stop sooner, so freeing up the resources they were using for other long-distance calls.

The data networks that began to appear in the middle of the 20th Century used a different philosophy, the transfer of packets. This new technique means that resources (lines and intermediary equipment) do not need to be used exclusively for the duration of a conversation. The idea is not to penalise communications between computers that may go for long periods without exchanging any information at all.

One of the fundamental differences between the two techniques (communication on a circuit and communication using packets) is the delay experienced by the information being transferred. With wired communication, this delay is small and consistent, while in packet transfer, the delay can be long and very unpredictable.

4.1. Architecture and protocols

Several protocol architectures have been designed to provide Voice over IP services. The most commonly used are H.323, SIP and IAX[5]. Proprietary, open and free end-user applications have been developed for all these.

(5)IAX is the Inter-Asterisk eXchange protocol. It arose out of the Asterisk project as a proprietary component, but has now gained global acceptance at the expense of H.323 and SIP.
We will look at Asterisk later on.

In fact, IP telephony is one of the main services provided by the various open networks that have been installed. One of the most relevant aspects of open networks is the cost of maintaining the infrastructure and the services, which is far lower than that incurred by large companies, and it is this same spirit what has given rise to IP telephony.

4.1.1. H.323

H.323 is recommended by the ITU Telecommunication Standardization Sector (ITU-T) for providing general audiovisual services on any packet-based network. It provides signalling and line control, control of signal transport and advanced services such as multipoint conferencing. It is part of the family of recommendations known as H.32x.

Additional reading

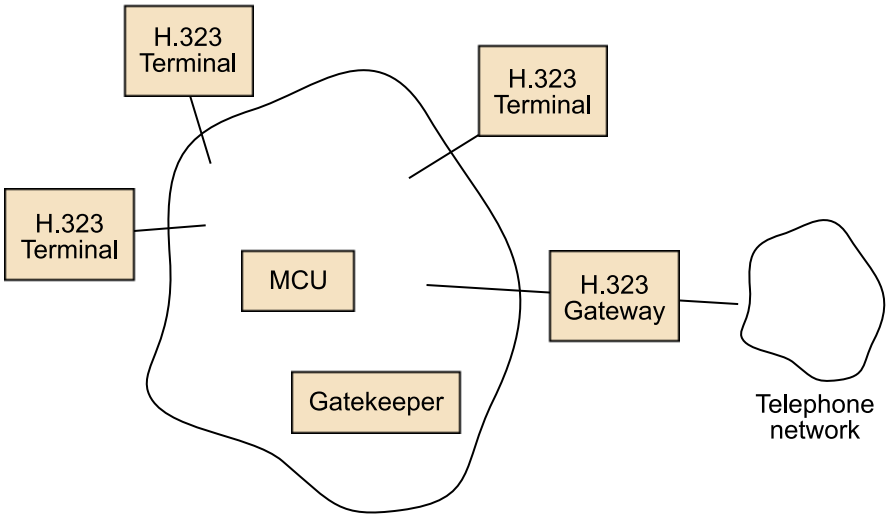
An exhaustive description of H.323 can be found in chapter 10 of L. Davidson, J. Peters (2000). *Voice over IP Fundamentals*. Cisco Press. pp. 229-230.

It was originally created to establish multimedia transport mechanisms over LANs, but evolved rapidly to cover VoIP. What is more, one of its most remarkable features is that it allows a computer connected to the Internet to communicate with a conventional telephone connected to a switched phone network.

The main features of H.323 are:

- Interoperability
- Independence from the network, platform and application.
- Multipoint configuration
- Bandwidth management
- Flexibility

The following figure illustrates the architecture of an H.323 system:



We can distinguish between four types of components which, when inserted into the Internet, provide point-to-point and point-to-multipoint multimedia communications:

- **Terminal**. An H.323 terminal could be a PC executing a specific application or a purpose-built ad-hoc device. The standard stipulates that it must be able to support audio communications and may optionally support video and data communications. This means that we can use simple devices that only support audio communications or more sophisticated devices that can also handle video.

- **Gateway**. The gateway is the element that allows us to connect an H.323 terminal to one which is not. A conventional telephone for example. To achieve this, it is responsible for translating the protocols used to establish and end calls between the two networks and to translate video and audio formats.
  This is optional in an H.323 environment. If we are connecting two H.323 terminals, it is not needed.

- **Gatekeeper**. The gatekeeper can be considered to be the brain of an H.323 network and it provides important services such as addressing, terminal authorisation and authentication, bandwidth management, billing etc. However, it is not needed when establishing communications between terminals. It is optional despite the services it provides. This is a fact. We can also establish calls without gatekeepers.

- **Multipoint control unit**. This allows communications to take place between three or more H.323 terminals. They must all establish a connection with the MCU which is then in charge of managing communications between all of them.

Although we have described gateways, gatekeepers and MCUs as being different components, some devices can provide all these functionalities.

4.1.2. SIP (Session Initiation Protocol)

SIP was created by the IETF to facilitate Internet communications between multimedia devices by taking advantage of pre-existing protocols such as RTP/RTCP and SDP. It provides mechanisms for the initiation, modification and ending of interactive sessions involving all types of multimedia information such as voice, images, online games and virtual reality.

Additional reading

You can find more information about SIP in chapter 11 of the book on H.323: L. Davidson, J. Peters (2000). *Voice over IP Fundamentals*. Cisco Press.

SIP is an application protocol developed along similar lines to HTTP and SMTP. It uses the client/server model meaning that it specifies requests and responses that are included in messages made up of legible text strings. SIP is in charge of the commands. The multimedia information travels in separate packets that are handled by a transport protocol such as RTP.

The primary purpose of SIP is to provide signalling mechanisms for communications established over IP, to imitate the services provided by a switched telephone network up to a certain extent. For example, the concepts of "dialling a number", the ringing sound when someone calls, the ring tone and the engaged tone etc. The mechanisms implemented by SIP have nothing to do with this, but the effects experienced by the user are the same.

SIP defines three types of participants:

- **User agents**. These are the end communication points. They may be clients (UAC, User Agent Clients), if they issue requests, or servers (UAS, User Agent Servers) if they respond to requests. Devices or applications that are SIP user agents must implement a UAC and a UAS.

- **Registry servers**. For SIP, all users will have an identifier or logic address in the form user@domain, that is fixed and does not change, but a connection can be initiated or received from any computer. We therefore need a mechanism to link the logic address with the physical address of the communications device. This is where registry servers come in. When a user wishes to establish communications, the client user agent will send a REGISTER request to a registry server that includes the logic address and the physical address of the device. The server establishes a connection for a certain period of time after which it expires and must be renewed.

- **Redirection or proxy servers**. These units allow SIP messages (request or response) to reach their destination. Proxy servers do the same job as IP protocol routers: They transmit messages from the source device to the destination device. Redirection servers tell the source device the address

of the proxy server. Servers can often act as redirectors or proxy servers depending on the task that needs to be performed.

A server will cover a domain and will receive requests from the users that make up this domain. It then finds the physical address of the destination user and locates a proxy server in order to send the request to.

SIP commands

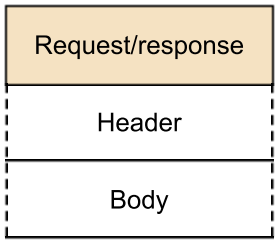
The following commands are sent by clients:

REGISTER	Used by user agents to state your IP address and the URL you wish to call
INVITE	Used to establish a session between two user agents
ACK	It confirms the establishment of a session
CANCEL	It cancels a pending request
BYE	It ends a session
OPTIONS	It allows you to obtain information about the capabilities of an agent without establishing a session.

Description of the protocol

The protocol is very similar to RTSP. SIP messages are text and are formatted into three parts as shown in the following image:

- Request/response line
- Request/response header
- Body of the message



The request line has three fields separated by a space:

```
SIP_command URL_request SIP_version
```

```
INVITE sip:usuari1@domain_alfa SIP/2.0
```

The response state line also has three fields:

```
SIP_version State_code State_description
```

SIP/2.0 200 OK

The header fields, both for requests and responses, are used to specify parameters relating to a particular function of the protocol. The protocol defines some of these fields and their possible values. It also allows new header fields to be defined.

4.2. Examples of IP telephony applications

There are various applications and systems that provide IP telephony services. Some of these were developed in the free programming world and others are proprietary:

- **Skype**. It is the best known example of a proprietary VoIP environment. It was created in 2003 by the people who wrote Kazaa, Janus Friis, a Dane, and Niklas Zennström, who is Swedish. The application is free, but the environment is closed. It also allows calls to be made from a PC to fixed lines for a modest price. Calls between PCs using the software are free.
  The application allows for the exchange of voice, image, video and text information, and provides additional services such as audio and video conferencing, an answer machine, call forwarding etc.

- **Asterisk**. The best-known application in the world of free software. It was created in 1999 by an American IT student called Mark Spencer because the company he had created needed a telephone exchange and he did not have enough money to buy one. He soon redirected the activities of the company to support the application exclusively, using the GPL philosophy.
  Asterisk provides many of the functionalities that can be found in standard exchanges such as voicemail, call forwarding, three-way calls etc. It also allows users to expand the functionalities with their own modules, which can be written in any programming language (however the native application is written in C). It supports all VoIP protocols such as H.323, SIP, AIX and MGCP.
  The system is also very scalable. It can be used for small systems such as internal telephony for small businesses, but it can also be used for call-centres, where the functionality and performance requirements are very high.

4.3. VoIP for mobile phones

VoIP was designed to be a computer-based application that could replace the use of telephones to make calls. When VoIP was first created, mobile phones did not exist. It was therefore only designed to replace landlines.

The appearance of mobile telephony and, above all, the fact that these devices can be connected to the Internet, has meant that VoIP technology has migrated in this direction, saving large amounts of money for voice calls.

However, powerful mobile phones are needed, as they have to be able to execute applications that implement SIP, SDP and RTP for voice transfer. The appearance of smartphones (mobiles that have similar abilities to PCs and a reasonable processor, a large amount of memory and an operating system that is adapted to this environment) have made it possible to develop VoIP technology for mobile phones.

The SIP standard is relatively new and is in undergoing constant evolution. Extensions are being developed that allow the behaviour of these types of devices to be customised. Some manufacturers are also producing mobiles that already have these technologies integrated.

The future seems to be pointing towards the abandonment of both traditional and mobile switched telephone networks.