Imperial College
London

NOTES

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

Network and Communications

Author:
Your Name (CID: your college-id number)

Date: May 18, 2018

1 Introduction

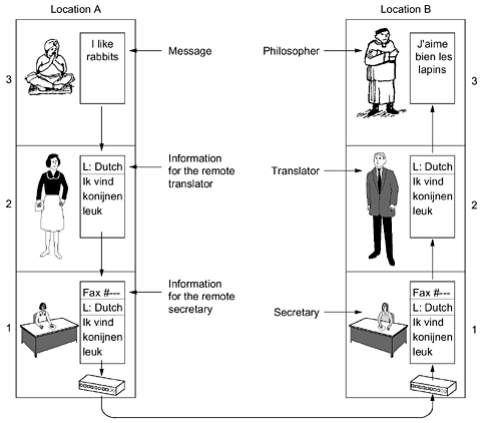
1.1 Protocol Hierarchies

Most networks are organised as a stack of **layers** or **levels**, each one built upon the one below it.

Layer n on one machine carries on a conversation with layer n on another machine. The rules and conventions used in this conversation are collectively known as the layer n **protocol**.

In reality, no data are directly transferred from layer n on one machine to layer n on another machine. Instead, each layer passes data and control information to the layer immediately below it, until the lowest layer is reached. Below layer 1 is the **physical medium** through which actual actual communication occurs.

Between each pair of adjacent layers is an **interface**. The interface defines which primitive operations and **services** the lower layer makes available to the upper one.



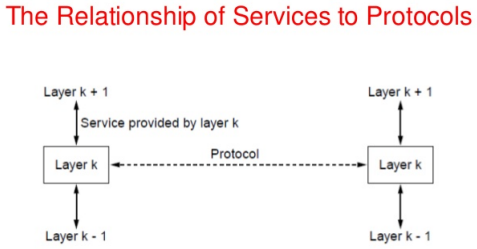
Analogy: Imagine two philosophers (layer 3), one of whom speaks English and one of whom speaks French. Since they have no common language, they each engage a translator (layer 2), each of whom in turn contacts a secretary (layer 1).

Philosopher 1 wishes to convey a message to philosopher 2. To do so, he passes the message in English across the 2/3 interface to his translator. The translators have agreed on a neutral language known to both of them, Dutch, so the message is converted to Dutch. The choice of language is the layer 2 protocol.

The translator then gives the message to a secretary for transmission, for example by fax. When the message arrives, it is translated in to French and passed across the 2/3 interface to philosopher 2.

Note that each protocol is completely independent of the other ones as long as the interfaces are not changed. The translators can switch from Dutch to Finnish. Similarly, the secretaries can switch from fax to email without disturbing other layers.

1.2 The Relationship of Services to Protocols



The relationship between a service and a protocol.

A **service** is a set of operations that a layer provides to the layer above it. A service relates to an interface between two layers, with the lower layer being the service provider and the upper layer being the service user.

A **protocol** is a set of rules governing the format and meaning of the packets, or messages that are exchanged by peer entities with a layer.

1.3 Design Issues for the Layers

Every layer needs a mechanism for identifying senders and receivers. As a consequence of having multiple destinations, some form of **addressing** is needed in order to specify a specific destination.

Error control is important because physical communication circuits are not perfect. Both ends of the connection must agree on which error-detecting and error-correcting method is being used. In addition, the receiver must have some way of telling the sender which messages have been correctly received and which have not.

An issue that occurs at every level is how to keep a fast sender from overflowing a slow receiver with data, this subject is called **flow control**

Multiplexing and Demultiplexing is the subject to support multiple communications in parallel.

When there are multiple paths between source and destination, a route must be chosen. This topic is called **routing**

1.4 Connection-Oriented and Connectionless Services

Layers can offer two different types of service to the layers above them: **connection-oriented** and **connectionless**.

Connection-oriented service is modelled after the telephone system. To use a connection-oriented network service, the service user first establishes a connection, uses the connection, and then releases the connection.

In contrast, **connectionless service** is modelled after the postal system. Each message carries the full destination address, and each one is routed through the system independent of all the others.

Each service can be characterised by a **quality of service**. Some services are reliable in the sense that they never lose data. Usually, a reliable service is implemented by having the receiver acknowledge the receipt of each message so the sender is sure that it arrived. The acknowledgement process introduces overhead and delays, which are often worth it but are sometimes undesirable.

Connection-Oriented Versus Connectionless Service

	Service	Example
Connection-oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Movie download
	Unreliable connection	Voice over IP
Connection-less	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Text messaging
	Request-reply	Database query

Six different types of service.

In **reliable message stream**, the message boundaries are preserved. When two 1024-byte messages are sent, they arrive as two distinct 1024-byte messages. This can be useful if pages of a book are sent over a network, it might be important to preserve the message boundaries.

In **reliable byte stream**, the connection is simply a stream of bytes, with no message boundaries. When 2048-byte arrives at the receiver, there is no way to tell if there were sent as one 2048-byte or 2 1024-bytes. They is useful when downloading movies when message boundaries are not relevant.

Digitised voice traffic uses **unreliable connection** because it is preferable for telephone users to hear a bit of noise on the line from time to time than to experience a delay waiting for acknowledgements.

An electronic junk-mail sender probably does not want to go to the trouble of setting up and later tearing down a connection to send one item. Nor is 100 percent reliable delivery essential, so may just need **unreliable datagram**

In some situations, the convenience of not having to establish a connection is desired, but reliability is essential. The **acknowledged datagram service** can be provided for these applications.

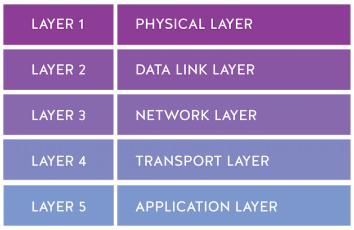
In a **request-reply service** the sender transmits a single datagram containing a request; the reply contains the answer.

1.5 Reference Models

The protocols associated with the OSI model are rarely used anymore, the model itself is actually quite general and still valid, and the features discussed at each layer are still very important.

The TCP/IP model itself is not of much use but the protocols are widely used.

1.5.1 The OSI Reference Model



In the OSI model, data flows down the transmit layers, over the physical link, and then up through the receive layers.

During transmission, each layer adds a header to the data that directs and indentifies the packet. This process is called encapsulation. The header and data together form the data packet for the next layer that, in turn, adds its header and so on.

The combined encapsulated packet is then transmitted and received. The receiving computer reverses the process, de-encapsulating the data at each layer with the header information directing the operations. Then, the application finally uses the data.

The **application layer** contains a variety of protocols that are commonly needed by users. This layer works with the application software to provide communications functions as required. It verifies the availability of a communications partner and the resources to support any data transfer.

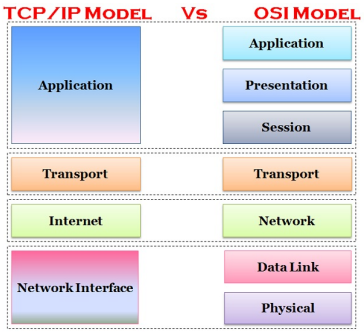
The **transport layer** provides quality of service (QoS) functions and ensures the complete delivery of the data.

The **network layer** handles packet routing via logical addressing and switching functions.

The **data link layer** transforms a raw transmission facility into a line that appears free of undetected transmission errors to the network layer. It accomplishes this task by having the sender break up the input data into data frames.

The **physical layer** is concerned with transmitting raw bits over a communication channel.

1.5.2 The TCP/IP Reference Model



TCP/IP also is a layered protocol but does not use all of the OSI layers, though the layers are equivalent in operation and function

The seven layers of the OSI model somewhat correspond with the four layers that make up the TCP/IP protocol.

The **transport layer** is designed to allow peer entities on the source and destination hosts to carry on a conversation. Two end-to-end transport protocols have been defined here: **TCP(Transmission Control Protocol)** and **UDP (User Datagram Protocol**.

TCP is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet. TCP also handles flow control to make sure fast sender cannot swamp a slow receiver with more messages than it can handle.

UDP is an unreliable, connectionless protocol. It is widely used for one-shot, client-server-type request-reply queries and applications in which prompt delivery is more important than accurate delivery.

The **internet layer** defines an official packet format and protocol called **IP (Internet protocol)**. The job of the internet layer is to deliver IP packets where they are supposed to go.

1.5.3 Repeaters, Hubs, Bridges, Switches, Routers and Gateways

There is a variety of ways to get frames and packets from one cable segment to another. Repeaters, hubs, bridges, switches, routers and gateways are in common use but they differ in subtle and not-so-subtle ways.

At the bottom at the physical layer, we find the **repeaters**. These are analog devices that are connected to two cable segments. A signal appearing on one of them is amplified and put out on the other. Repeaters do not understand frames, packets or headers. They understand volt. Next we come to **hubs**. A hub has a number of input lines that it joins electrically. Frames arriving on any of the line are sent out on all the others. If two frames arrive at the same time, they will collide. Hubs differ from repeaters in that they do not amplify the incoming signals. Like repeaters, hubs do

not examine addresses or use them in any way.

The data link layer is where is find **bridges** and **switches**. A bridge connects two or more LANs. When a frame arrives, software in the bridge extracts the destination address from the frame header and looks it up in a table to see where to send the frame. This address is the 48-bit MAC address. Switches are similar to bridges in that both route on frame addresses. The main difference is that a switch is most often used to connect individual computers.

The network layer uses **routers**. When a packet comes into a router the frame header and trailer are stripped off and the packet located in the frame's payload field is passed to the routing software. This software uses the packet header to choose an output line. For an IP packet, the packet header will contain 32-bit(IPv4) or 128-bit(IPv6) address.

In the transport layer we find **gateways**. These connect two computers that use different connection-oriented transport protocols. For example, a computing using TCP/IP protocol needs to take to a computer using ATM transport protocol. The transport gateway can copy the packets from one connection to the other, reformatting them as need be.

2 Application Layer

2.1 DNS (Domain Name System)

Although programs theoretically could refer to hosts, mailboxes, and other resources by their IP addresses, these addresses are hard for people to remember. To solve this problem, the **DNS** was invented.

The essence of DNS is the invention of a hierarchical, domain-based naming scheme and a distributed database system for implementing this name scheme.

The DNS works as follows. To map a name onto an IP address, an application program calls a **resolver**, passing it the name as a parameter. The resolver sends a UDP packet to the local DNS server, which then looks up the name and returns the IP address to the resolver, which then returns it to the caller. With the IP address, the program can then establish a TCP connection.

DNS is a connectionless protocol. Both queries and replies have the same format. DNS uses UDP (port 53) because it always only involves two network packets (request and response), setting up and tearing down a TCP connection every time would be wasteful.

2.1.1 DNS Name Space

The internet is divided into over 200 **Top Level Domains**, where each domain covers many hosts. Each domain is partition into subdomains, and these are further partitioned, and so on.

All these domains can be represented by a tree. The leaves of the tree represent domains that have no subdomains. A leaf domain may contain a single host, or it may represent a company that contains thousands of hosts.

The Top Level Domain com in two flavours: generic and countries. The original generic domains were com (commercial), edu (education institutions), gov (the US Federal Government), net (network providers) and org (nonprofit organizations).

Each domain is named by the path upward from it to the root. To create a new domain, permission is required of the domain in which it will included. For example, if a VLSI group is started at Yale and wants to be known as vlsi.cs.yale.edu, it has to get permission from whoever manages cs.yale.edu.

2.1.2 Resource Records

A **resource record** is a five tuple. Including Domain Name, Time to Live, Class, Type and Value.

The **Domain Name** tells the domain to which this record applies.

The **Time to Live** field gives an indication of how stable the record is. Information that is highly stable is assigned a larger value, whereas information that is highly volatile is assigned a small value.

For Internet information, the **Class** is always IN. Other classes are rarely seen.

The **Type** field tells what kind of record this is.

Type A: the **value** holds a 32-bit IP address of the host.

Type NS: the **value** holds the authoritative name server for that domain.

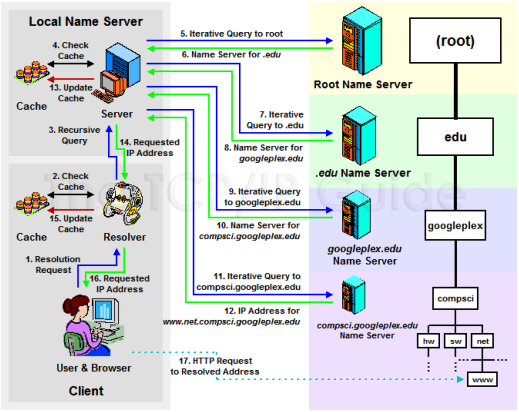
Type CNAME allow aliases to be created. The tuple may be in the form of:

cs.mit.edu — 86400 — IN — CNAME — lcs.mit.edu

To create an alias cs.mit.edu for lcs.mit.edu

Type MX: the **value** is the name of the mail server that handles (incoming) mail for that host or domain

2.1.3 DNS Name Resolution Process



1. You type into your Web **browser** ”www.net.compsci.googleplex.edu”.

2. Web browser passes the name ”www.net.compsci.googleplex.edu” to the local **resolver**

3. The resolver checks its **cache** to see if it already has the address for this name.

4. If not, the resolver generates a recursive query and sends it to **local DNS server**

5. The local DNS server checks its **cache** to see if it already has the address for this name. If it does, it would return the information, marked ”non-authoritative”, to the resolver.

6. Local DNS server generates an iterative request for the name and sends it to a **root name server**. It returns the name and address of the name server for the ”.edu” domain.

7. Local DNS server generates an iterative request and sends it to ”.edu” server. It returns the name and address of the name server for the ”googleplex.edu” domain.

8. Local DNS server generates an iterative request and sends it to the ”googleplex.edu” server. It returns the name and address of the name for the ”compsci.googleplex.edu” domain.

9. Local DNS server generates an iterative request and sends it to the ”compsci.googleplex.edu” server

10. The server ”compsci.googleplex.edu” is authoritative for ”www.net.compsci.googleplex.edu”. It returns the IP address.

11. Local DNS server cache this resolution. Then pass it to resolver

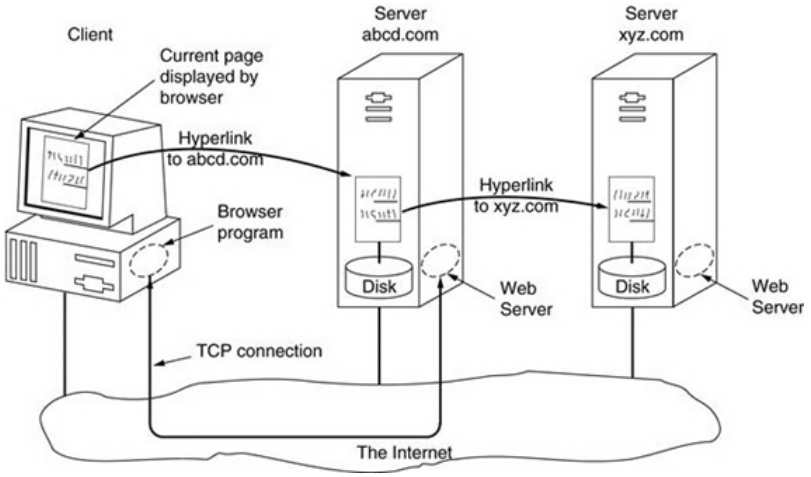
12. Resolver cache this resolution. Then give address to browser.

2.2 World Wide Web

The Web consists of a vast, worldwide collection of **documents** or **web pages**, one website has one or more webpages. A webpage may contain several **objects** such as images, videos and stylesheets.

Each page may contain links to other pages anywhere in the world. Users can follow a link by clicking on it, which then takes them to the page pointed to. The idea of having one page point to another is called **hypertext**. Strings of text that are links to other pages, called **hyperlinks**, are often highlighted.

Pages are viewed with a program called a **browser**. The browser fetches the page requested, interprets the text and formatting commands on it, and displays the page on the screen.



The model above shows how the web works. Here the browser is displaying a web page on the **client** machine. When the user clicks on a line of text that is linked to a page on the abcd.com **server**, the browser follows the hyperlink by sending a message to abcd.com server asking it for the page. When the page arrives, it is displayed. If this page contains a hyperlink to a page on the xyz.com server that is clicked on, the browser then sends a request to that machine for the page, and so on indefinitely.

2.2.1 The Client Side

Embedded hyperlinks needs a way to name any other page on the Web. Pages are names using **URL (Uniform Resource Locators)**. A typical URL is http://www.abcd.com/products.h
The URL has three parts: the name of the protocol (http), the DNS name of the machine (www.abcd.com) and the name of the file (products.html).

Suppose that a user is browsing the Web and finds a link that points to ITU's home page, which is http://www.itu.org/home/index.html. This are the steps that occur when the link is selected

1. The browser determine the URL
2. The browser asks DNS for the IP address of www.itu.org
3. DNS replies with 156.106.192.32
4. The browser makes a TCP connection to port 80 on 156.106.192.32
5. It then sends over a request asking for file /home/index.html

6. The www.itu.org server sends the file /home/index.html
7. The TCP connection is released
8. The browser displays all the text in /home/index.html
9. The browser fetches and displays all images in this file

Not all pages contain HTML. A page may consist of a formatted document in PDF format, an icon in GIF format, a photograph in JPEG format, a song in MP3 format. When a server returns a page, it also returns some additional information including the MIME type of the page. If the MIME type is not one of the built-in, the browser consults its table of MIME types to tell it how to display the page. This includes two possibilities:

A **plug-in** is a code module that the browser fetches from a special directory on the disk and installs as an extension to itself. After the plug-in has done its job, the plug-in is removed from the browser's memory.

A **Helper application** is a complete program that runs as a separate process. Helper applications are large programs that exist independently of the browser, for example Adobe's Acrobat Reader for displaying PDF file.

2.2.2 The Server Side

Modern web servers do more than just accept file names and return files. In fact, the actual processing of each request can get quite complicated.

1. Resolve the name of the Web page requested
2. Authenticate the client. This step is needed for pages that are not available to the general public.
3. Perform access control on the client. This checks to see if there are restrictions on whether the request may be satisfied given the client's identity and location.
4. Perform access control on the Web page. This checks to see if there are any access restrictions associated with the page itself.
5. Check the cache.
6. Fetch the requested page from disk
7. Determine the MIME type to include in the response
8. Return the reply to the client
9. Make an entry in the server log

2.2.3 Caching

Caching is a simple way to improve performance by saving pages that have been requested in case they are used again. The usual procedure is for some process, called a **proxy** to maintain the cache.

A browser can be configured to make all page requests to a proxy instead of to the page's real server. If the proxy has the page, it returns the page immediately. If not, it fetches the page from the server, adds it to the cache for future use, and returns it to the client.

Individual PCs often run proxies so they can quickly look up pages previously visited. On a company LAN, there is often a proxy shared by all the machines on the LAN. Many **ISP (Internet Service Provider)** also run proxies to speed up access for all their customers. So requests first go to the local proxy, if it fails, the local proxy queries the LAN proxy. If that fails, the LAN proxy queries the ISP proxy. The ISP proxy must succeed, either from its cache or from the server itself. This scheme involving multiple caches tried in sequence is called **hierarchical caching**.

Some pages when cached, a user getting a copy from the cache would get **stale** data. There are two approaches to deal with the problem.

The first approach is to base the holding time on the Last-Modified header. If a page was modified an hour ago, it is held in the cache for an hour.

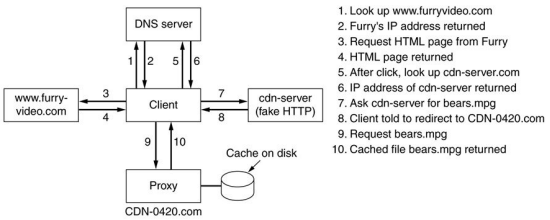
The other approach to use a feature If-Modified-Since request header, which proxy can send to a server. It specifies the page the proxy wants and the time the cached page was last modified. If the page was not modified, the server sends back a "Not Modified" message so the proxy can use the cached page. Otherwise, a new page is returned.

2.2.4 CDN (Content Delivery Networks)

To understand why CDNs are so widely used, you first need to recognise the issue they're designed to solve. Known as **latency**, it's the delay that occurs from the moment you request to load a web page to the moment its content actually appears onscreen.

To minimise the distance between the visitors and your website's server, a CDN stores a cached version of its content in multiple geographical locations. With the content replicated at thousands of sites worldwide, there is clearly great potential for improving performance. However, to make this work, there has to be a way to redirect the client's request to the nearest CDN server, preferably one colocated at the client's ISP.

The Wireless Web



Steps in looking up a URL when a CDN is used.

23

Consider Furry Video's Web page with a URL link to http://cdn-server.com/furryvideo/bears.mpg.

1. When a user types in the URL www.furryvideo.com
2. DNS returns the IP address of Furry Video's own web page
3. The browser request the HTML page from Furry Video
4. Furry Video returns the HTML page to be displayed by the browser
5. When the URL link is clicked on, the browser asks DNS to look up cdn-server.com.
6. DNS then returns the IP address of cdn-server
7. The browser then sends an HTTP request to this IP address, expecting to get back a MPEG file
8. No MPEG file is returned because cdn-server.com does not host any contents. Instead it examines the IP address of the incoming request and loops it up in its database to determine the location of the user. It then uses this information the determine which CDN's content server can give the user the best service. After this, cdn-server.com sends back a response with the file's URL on the CDN content server. For this example, assume it is www.CDN-0420.com/furryvideo/bears.mpg.
9. The browser the request bears.mpg file from CDN-0420.com server
10. The proxy then returns to the browser the cached file bears.mpg

2.2.5 HTTP (HyperText Transfer Protocol)

HTTP specifies what messages clients may send to servers and what responses they get back in return.

The usual way for a browser to contact a server is to establish a TCP connection to port 80 on the server's machine.

In HTTP 1.0, after the connection was established, a single request was sent over and a single response was sent back. Then the TCP connection was released. This can be expensive to operate when a web page contained large number of icons and images as each icons is a different request. In HTTP 1.1, it supports **persistent connections**

where it is possible to establish a TCP connection, send a request and get a response, and then send additional requests and get additional responses.

There are several **methods** in HTTP.

HTTP Methods

Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page's header
PUT	Request to store a Web page
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

The built-in HTTP request methods.

17

Usually the server returns with a status code.

The HyperText Transfer Protocol (3)

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

a) The status code response groups

2.2.6 Statelessness and Cookies

HTTP is a **stateless** protocol. The browser sends a request to a server and gets back a file. Then the server forgets that it has ever seen that particular client. This raises the question of how servers can distinguish different users.

To solve this problem, **cookies** can be used, which is a small file (4KB). When a client requests a web page, the server can supply additional information along with the requested page including a cookie.

A cookie may contain up to five fields. The **Domain** tells where the cookie came from. The **Path** is a path in the server's directory structure that specifies which files my use the cookie. The **Content** takes the form "name = value" and it can be anything the server wants. The **Expires** field specifies when the cookie expires. The **Secure** field can be set to indicate that the browser may only return the cookie to a secure server.

Just before a browser sends a request for a page to some website, it checks the cookies directory to see if any cookies there were placed by the domain the request is going to. If so, all the cookies placed by the domain are included in the request message. When the server gets them, it can interpret them any way it wants to.

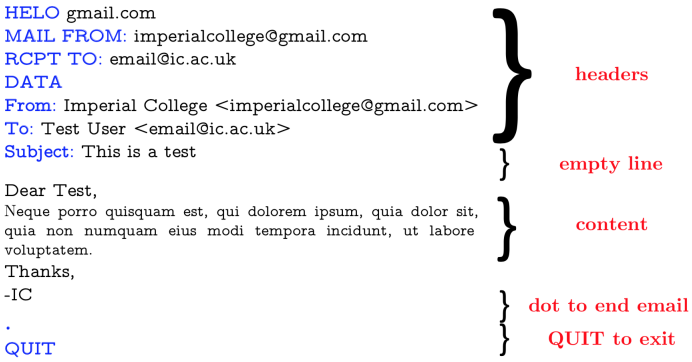
A controversial use of cookies is to secretly collect information about user's browsing habit. An advertising agency can contact major web sites and places banner ads on their page. Instead of giving the site a GIF, it gives them a URL to add to the page. When a user first visits a page containing the ad, the browser fetches and HTML file and sends a request for the image. A GIF file containing an ad is returned, along with a cookie containing a unique user ID. When a user visits another Web page contains an ad from the same advertising agency, the browser will send a request message together with a cookie containing the user's ID. Overtime, the agency can build up a complete profile of the user's browsing habits.

2.3 Electronic Mail

2.3.1 SMTP (Simple Mail Transfer Protocol)

Within the internet, e-mail is delivered by having the source machine establish a TCP connection to port 25 of the destination machine.

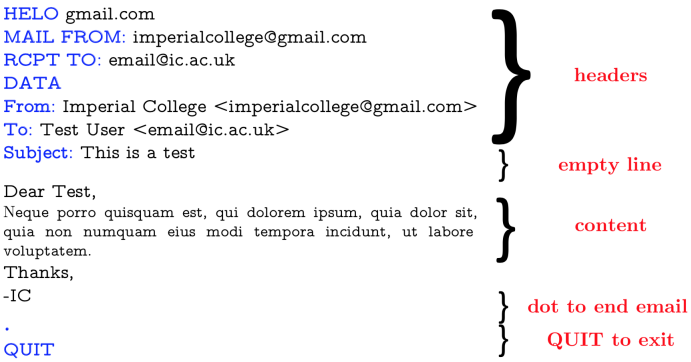
After establishing the TCP connection, the client waits for the server to talk first. The server starts by sending a line of text giving its identity and telling whether it is prepared to receive mail. If it is not, the client releases the connection and tries again later.



2.3.2 POP3

SMTP works if we assumed that all users work on machines that are capable of sending and receiving email. A problem occurs if one end is not currently online, the a TCP connection cannot be established and the email cannot be sent.

One solution is to have a message transfer agent on an ISP machine accept email for its customers and store it in their mailboxes on an ISP machine. The **POP3** protocol allows email to be copied from the ISP the to user.



POP3 begins when the user starts the mail reader. The mail reader calls up the ISP and establishes a TCP connection with the message transfer agent at port 110.

During the authorisation state, the client sends over its username and then its password.

After a successful login, the client can then send over the LIST command, which causes the server to list the contents of the mailbox.

The client can retrieve messages using the RETR command and mark them for deletion with DELE.

When all the messages have been retrieved, the client gives the QUIT command to terminate the transaction. When the server has deleted all the messages, it sends a reply and breaks the TCP connection.

Advantages

1. Mail stored locally, accessible even without internet access
2. Saves server storage space
3. Easier to implement

2.3.3 IMAP

POP3 assumes that the user will clear out eh mailbox on every contact and work offline after that. IMAP assumes that all the email will remain on the server indefinitely in multiple mailboxes.

IMAP establishes TCP connection at port 143

Advantages

1. Mail stored in remove server, accessible from multiple different locations
2. Saves local storage space
3. Mails are automatically backed up

3 Transport Layer

The ultimate goal of the transport layer is to provide efficient, reliable and cost-effective service to its users, normally processes in the application layer.

The existence of the transport layer makes it possible for the transport service to be more reliable than the underlying network service. Lost packets and mangled data can be detected and compensated for by the transport layer.

The transport layer service is very similar to the network layer service. The difference between these two services is whom the services are intended for. Few users write their own transport entities, and thus few users or programs ever see the bare network service. Consequently, the transport service must be convenient and easy to use.

3.1 Berkeley Sockets

To allow users to access the transport service, the transport layer must provide some operations to application programs, that is, a transport service interface. The **Berkeley Sockets Interface** is adopted by all UNIX systems (as well as Windows).

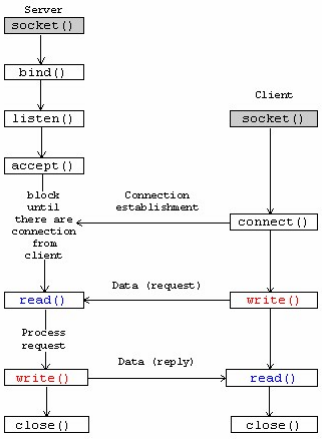
Each socket has a socket number consisting of the IP address of the host and a 16-bit number local to that host, called a **port**. It also includes the transport layer protocol e.g. 146.179.40.24:80 TCP.

A connection is identified by the socket identifiers at both ends, that is, (socket1, socket2). E.g. (146.179.40.24:80 TCP, 192.168.1.1:7155 TCP) . Therefore each socket may be used for multiple connections at the same time.

Berkeley Sockets (2)

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

The socket primitives for TCP



The SOCKET primitive creates a new end point and allocates table space for it within the transport entity.

The BIND primitive attaches a local address to the newly-created socket.

The LISTEN primitive allocates space to queue incoming calls for the case that several clients try to connect at the same time.

The ACCEPT primitive blocks the caller until a connection attempt arrives.

The CONNECT primitive is used by clients to attempt to establish a connection.

When both sides have executed the CLOSE primitive, the connection is released.

In connectionless protocols, such as UDP, there is no need for LISTEN, ACCEPT and CONNECT.

3.2 TCP (Transmission Control Protocol)

TCP was specifically designed to provide a reliable end-to-end byte stream over an unreliable internetwork. TCP service is obtained by both the sender and receiver creating end points, called **sockets**.

All TCP connections are **full duplex** and **point-to-point**. Full duplex means that traffic can go in both directions at the same time. Point-to-point means that each connection has exactly two end points.

The sending and receiving TCP entities exchange data in the form of **segments**. A TCP segment consists of a fixed 20-byte header followed by zero or more data bytes.

The basic protocol used by TCP is the sliding window protocol. A key feature of TCP, is that every byte on a TCP connection has its own 32-bit **sequence number**. When a sender transmits a segment, it also starts a timer. When the segment arrives at the destination, the receiving TCP entity sends back a segment bearing an **acknowledgement number** equal to the next sequence number it expects to receive. If the sender's timer goes off before the acknowledgement is received, the sender transmits the segment again.

3.2.1 The TCP Segment Header

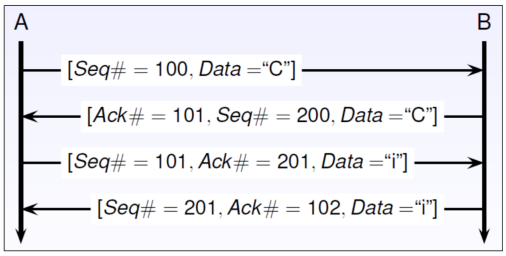
TCP Header

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size															
16	128	Checksum																Urgent pointer (if URG set)															
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

The **Source port** and **Destination port** identify the local end points of the connection.

The **sequence number** in a TCP segment indicates the sequence number of the first byte carried by that segment. Sequence numbers are associated with bytes in the data stream, it is not a segment numbering system. When the TCP connection is set up, a random **Initial Sequence Number (ISN)** is decided upon.

An **acknowledgement number** represents the first sequence number not yet seen by the receiver.



TCP connection consists of a full-duplex link, therefore it has two streams hence two different sequence number.

The **TCP header length** tells how many 32-bit words are contained in the TCP header. This information is needed because the **Options** field is of variable length.

URG flag (1-bit): ”urgent” flag, used to inform the receiver that the sender has marked some data as ”urgent”. The location of this urgent data are marked by the **urgent data pointer field**.

ACK flag (1-bit): signals that the value contained in the acknowledgment number represents a valid acknowledgment.

When an application passes data to TCP, TCP may send it immediately or buffer it in order to collect a larger amount to send at once. To force data out, applications can use the **PUSH flag**, which tells TCP not to delay the transmission.

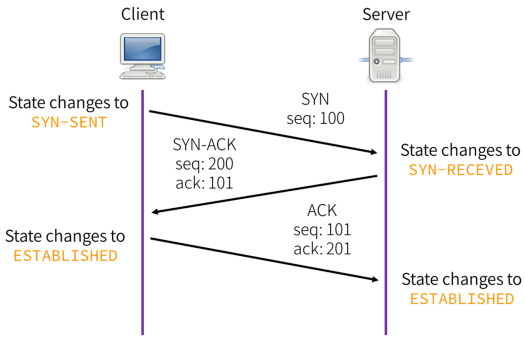
RST flag (1-bit): used during connection setup and shutdown

SYN flag (1-bit): used during connection setup and shutdown

FIN flag (1-bit): used during connection shutdown

The **Checksum** is also provided for extra reliability.

3.2.2 TCP Connection Establishment



Connections are established in TCP by means of the **three-way handshake**. To establish a connection, the server passively waits for an incoming connection by ex-

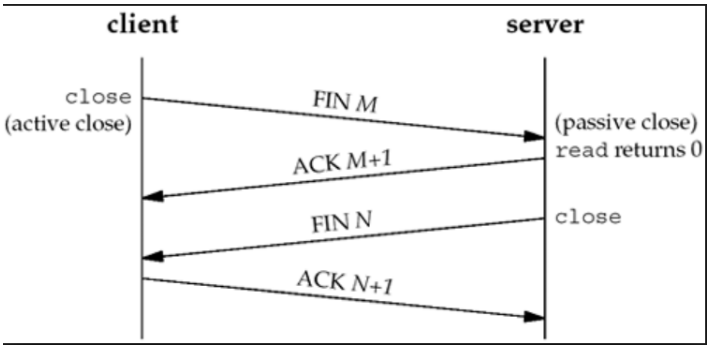
ecuting the LISTEN and ACCEPT primitives. The client executes a CONNECT primitive, which chooses a sequence number and sends a TCP segment with the SYN bit on and ACK bit off. The SYN segment consumes 1 byte of sequence space.

When the segment arrives at the server, it checks to if the port in the Destination Port Field has called LISTEN. If not, it sends a reply with the RST bit to reject the connection.

If it accepts, an acknowledgement segment is sent back together with its own initial sequence number.

In the end, the server acknowledges the sequence number of the client.

3.2.3 TCP Connection Termination



3.2.4 TCP Congestion Control/Flow Control

When the load offered to any network is more than it can handle, congestion builds up. Although the network layer also tries to manage congestion, most of the heavy lifting is done by TCP because the real solution to congestion is to slow down the data rate.

All the Internet TCP algorithms assume that timeous are caused by congestions.

Two potential problems exist. If the sender sends over the size of receiver buffer the buffer will overflow and no more data can be sent until the receiver clear its buffer. Another problem is due to **internal congestion** within the network. These problems are addressed by TCP flow control and congestion control.

Flow control is controlled by the receiving side. It ensures that the sender only sends what the receiver can handle.

Congestion control is a method of ensuring that everyone across a network has a ”fair” amount of access to network resources, at any given time.

Each sender to maintain two windows: a **receiver window**, which the receiver specifies based on its buffer size, and a **congestion window**. Each reflects the number of bytes the sender may transmit. The suitable window size will be the minimum of

receiver window and congestion window.

Calculations

$$Window\ size = min(receiver\ window,\ congestion\ window)$$
$$Maximum\ achievable\ throughput = \frac{Window\ size}{Round\ Trip\ Time}$$

Slow Start

When a connection is established, the sender initialises the congestion window to the size of the **maximum segment size (MSS)** in use of the connection. It then sends down one maximum segment size segment. If this segment is acknowledged, it adds one MSS to the congestion. Then it sends two maximum segment size segments. For each segment acknowledged, the congestion window is incremented by one MSS. When the congestion window is n segments, and all n segments are acknowledged, the congestion window is incremented by the byte count corresponding to n segments. This exponential algorithm is called **slow start**.

Congestion Avoidance

This process continues until a **timeout** occurs. When a timeout occurs, a **threshold** is set to half of the current congestion window, and the congestion window is reset to one MSS. The slow start algorithm is then used until the congestion window size reaches the threshold. Once the threshold is hit, the exponential growth stops and successful transmission grow the congestion window linearly (by one one MSS each burst). This linear growth algorithm is called **Congestion Avoidance**.

If no more timeout occurs, the congestion window will continue to grow to the size of the receiver window. At that point, it will stop growing and remain constant.

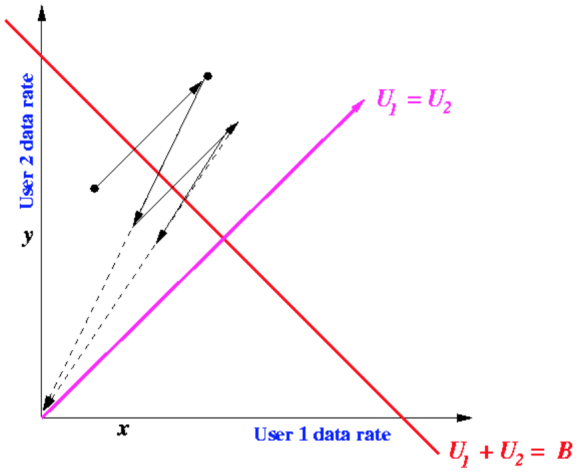
Fast Recovery

Another problem that may occur is package loss, where the receiver cannot received the segment. Three duplicate ACKs (four identical ACKs) are interpreted as a NACK. Both timeouts and NACKs signal a loss, but they say different things about the status of the network. A timeout indicates congestion. NACK suggest that the network is still able to deliver segments along that path. TCP handles NACK by cutting the congestion window by half, the runs Congestion Avoidance. This process is known as **Fast Recovery**.

3.2.5 TCP Congestion Control AIMD

AIMD stands for **Additive Increase/ Multiplicative Decrease** is an algorithm used in TCP congestion control. AIMD increases the congestion window linearly, and reduce the window size exponentially when a congestion occurs.

AIMD is **fair**, this means if K applications sharing a bottleneck with capacity R then each get R/K of the available bandwidth.



Consider the graph shown above. The axis represents the bandwidth each process is receiving.

The line $U1 + U2$ represents the capacity of the network. A point on this line indicates efficiency.

The line $U1 = U2$ represents fairness. A point on this line indicates each process gets a fair share of the bandwidth.

Consider we start at the leftmost point on the graph. The point is below the line of capacity, which means there is under-allocation, so both processes additively increase sending rate. This results in moving in a direction parallel to $U1 = U2$.

Until up to a certain point, the network is overloaded and congestion occurs, two processes will perform multiplicative decrease. Assume both processes decrease their rate by a factor of two, this results in moving towards the origin.

The process continues until it converges to the fairness line $U1 = U2$, reaching the optimal point (The intersection of the two diagonals).

3.2.6 TCP Timer Management

TCP uses a **retransmission timer**. When a segment is sent, a retransmission timer is started. If the segment is acknowledge before the time expires, the timer is stopped. Otherwise, the segment is retransmitted.

To calculate the timeout interval, TCP keeps track of two variables. **RTT (Round Trip Time)**, which is the best current estimate of the round-trip time to the destination. Each transmission, TCP measures how long the acknowledgement took and update the RTT. Another variable is the **deviation of the RTT, D**. While D is not exactly the same as standard deviation, it is good enough to calculate the **timeout interval**.

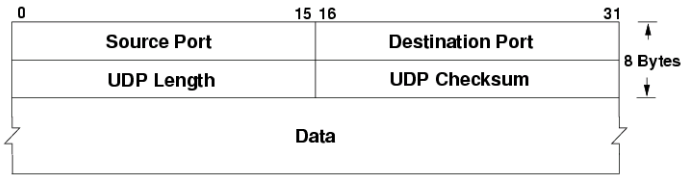
$$Timeout = RTT + 4 \times D$$

3.3 UDP (User Datagram Protocol)

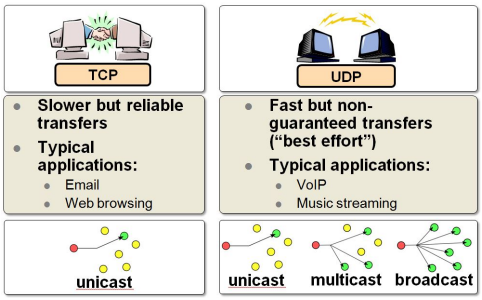
UDP provides a way for applications to send encapsulated IP **datagrams** and send them without having to establish a connection.

UDP does not do flow control, error control or retransmission upon receipt of a bad segment.

3.3.1 UDP Header



3.4 TCP vs UDP



TCP Advantages

1. TCP guarantees the data gets to destination, in-order, without duplication

2. TCP provides congestion control. It will try to use as much of the bandwidth without overwhelming the receiver

3. TCP provides a send buffer and a receive buffer. Applications can read from the receive buffer when they are ready.

TCP Disadvantages

1. TCP cannot be used for multicasting and broadcasting

2. Higher overhead to establish, maintain and teardown connection

3. TCP has no block boundaries

UDP Advantages

1. UDP can support broadcast and multicast transmission

2. UDP does not have to setup connection and do error recovery, therefore it has less delay

3. UDP has smaller packet header overhead

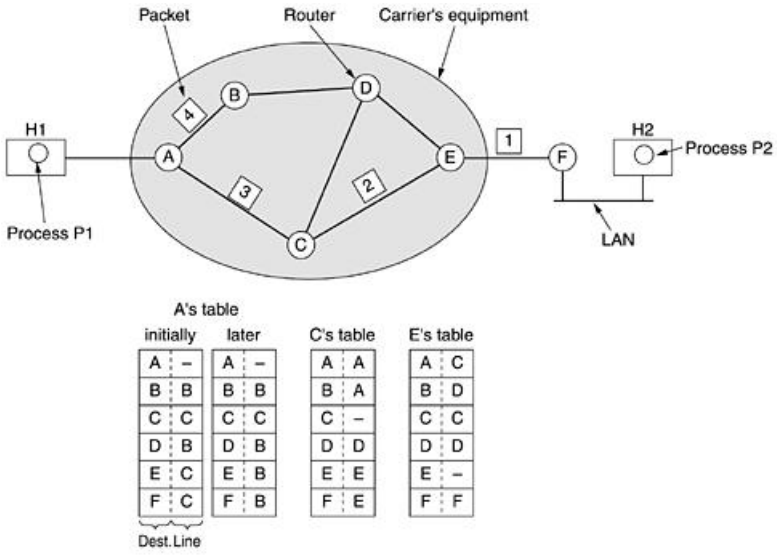
UDP Disadvantages

1. UDP is unreliable. Packets may be loss, maybe in the wrong order, or sent twice.
2. UDP suffers worse packet loss than TCP.

4 Network Layer

The network layer is concerned with getting packets from the source all the way to the destination. Getting to the destination may require making many hops at intermediate **routers** along the way. To achieve its goal, the network layer must know about the topology of the communication subnet (the set of all routers) and choose appropriate paths through it.

4.1 Implementation of Connectionless Service



If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. The packets are frequently called **datagrams** and this network is called **Packet Switched Network**.

Suppose that the precess P1 has a long message for P2. It hands the message to the transport layer with instructions to deliver it to process P2 on host H2. Let us assume that the message is four times longer than the maximum packet size, so the network layer breaks it into four packets 1,2,3,4 and sends each of them to router A.

Every router has an internal table telling it where to send packets for each possible destination. In A's initial table, A should forward the packets to C in order to get to destination F. As the packets arrived at A, packets 1,2,3 were stored briefly and each was forwarded to C.

However something different happened to packet 4. When it got to A it was for-

warded to B instead of C. Perhaps A learned of a traffic jam somewhere along the ACE path and updated its routing table (shown is A's later table).

4.1.1 Advantages

1. Efficient use of the network
2. Can easily get around broken bits of the network

4.1.2 Disadvantages

1. It is unreliable. Packets may arrive in the wrong order, some might even be loss.
2. Higher processing power at end host to reassemble the packets.

4.2 Implementation of Connection Oriented Service

Connection Oriented Services uses a **Circuit Switched Network**. When a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers. The route is used for all traffic flowing over the connection.

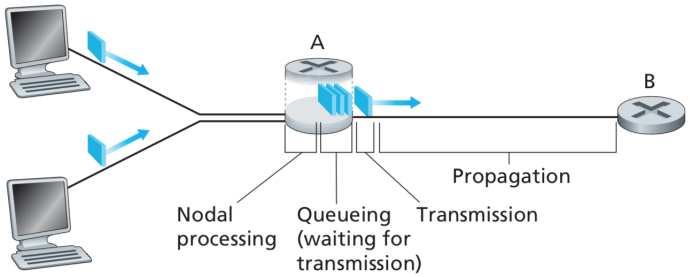
4.2.1 Advantages

1. All packets uses the same path, therefore guarantees data rate
2. Provide reliable connection suitable for long continuous transmission

4.2.2 Disadvantages

1. The dedicated path cannot be used to transmit other data even if its free, therefore inefficient use of the network
2. Higher overhead to setup the connection

4.3 Sources of Delays



$$Nodal\ Delay = Processing\ Delay + Queueing\ Delay + Transmission\ Delay + Propagation\ Delay$$

Processing Delay

The time require to process a packet. For example, checking error bits

Queueing Delay

The time spent waiting in a queue

Transmission Delay

The time to send out/absorb all of the packet bits.

The transmission delay is calculated by:

$$\frac{Packet\ size(bits)}{Link\ Transmission\ Rate(bps)}$$

Propagation Delay

The time for one bit to propagate from source to destination.

The propagation delay is calculated by:

$$\frac{Distance\ between\ source\ and\ destination(m)}{Propagation\ speed\ of\ medium(m/s)}$$

4.4 Routing Algorithms

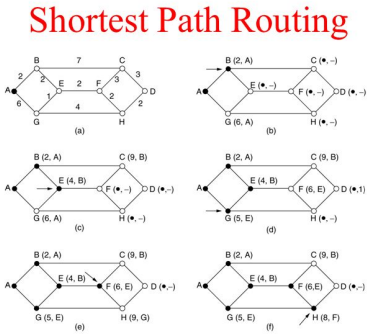
The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on.

One can think of a router as having two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line in use for it in the routing tables. This process is called **forwarding**. The other process is responsible for filling in and updating the routing table.

When a packet arrives at a router. The following steps are applied to forward a packet:

1. The router performs a **checksum** on the frame of data link layer
2. **Checks MAC address** to see if it matches its own MAC address.
3. **Lookup destination IP address** in routing table to find the IP address of the router to forward this packet to.
4. Calls **ARP** protocol to retrieve the MAC address of the destination router.
5. **Add frame** to packet including own MAC address and next router MAC address.
6. **Forward** the packet

4.4.1 Shortest Path Routing



The first 5 steps used in computing the shortest path from A to D.
The arrows indicate the working node.

The idea is to build a graph of the subnet, with each node representing a router and each arc representing a communication line. To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph. There are several algorithms for computing the shortest path between two nodes of a graph. One of which is the **Dijkstra Algorithm**.

4.4.2 Flooding

Another static algorithm is flooding, in which every incoming packet is sent out on every outgoing line except the one it arrived on. Flooding obviously generates vast numbers of duplicate packets, so some measures are taken to damp the process.

One such measure is to have a hop counter contained in the header of each packet, which is decremented in each hop, with the packet being discarded when the counter reaches zero.

An alternative technique is to keep track of which packets have been flooded, to avoid the sending them out a second time.

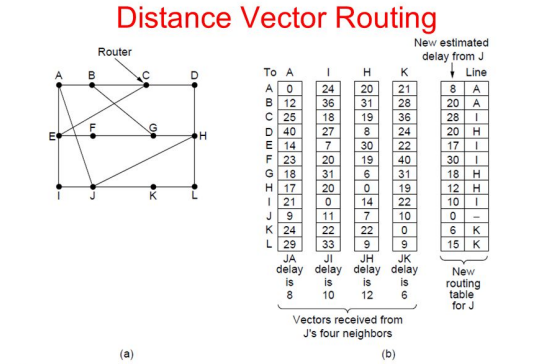
A variation of flooding is **selective flooding**. The routers do not send every incoming packet out on every line, only on those lines that are going approximately in the right direction.

Flooding is not practical in most applications, but it does have some uses in military applications where large number of routers may be blown to bits at any instant.

4.4.3 Distance Vector Routing

Modern computer networks generally use dynamic routing algorithms because static algorithms do not take the current network load into account.

Distance vector algorithms operate by having each router maintain a table giving the best known distance to each destination and which line to use to get there. These tables are updated by exchanging information with the neighbours.



(a) A network.
(b) Input from *A, I, H, K, and the new routing table* for *J*.

The first four columns of part b shows the delay vectors received from the neighbours of router J. Suppose J has measured its delay to its neighbour A, I, H, K as 8, 10, 12, 6 msec, respectively. Consider how J computes its new route to router G. It knows that it can get to A in 8 msec and A claims to be able to get to G in 18 msec, so J knows it can arrive G in 26 msec if it forwards its packets to A. Similarly, it computes to delay to G via I, H and K and finds the best value is 18. So it forward its packets to router H.

There is a problem with distance vector algorithms known as **Count to infinity problem**. Consider a situation in which all the lines and routers are initially up. Routers B, C, D, and E have distances to A of 1, 2, 3 and 4 respectively. Suddenly A goes down. At the first packet exchange, B does not hear anything from A. Fortunately, C says its knows a path to A of length 2. As a result, B thinks it can reach A via C, with a path length of 3. On the second exchange, C notices that each of its neighbours claims to have a path to A of length 3. It picks one at random and make its new distance to A as 4. Gradually all routers work their way up to infinity.

The Count to infinity problem causes a packet to be routed around in a cycle on the internet. IP has several methods to handle this problem:

Maximum Hop Count

IP header has a field Time To Live, in each hop the Time To Live field of a packet is decremented. The packet is discarded when Time To Live field reaches 0. This prevents a packet routing in a loop infinitely.

Split Horizon

A router will not advertise a route back onto the router which it has learnt

Route Poisoning

1. When a router detects that one of its connected routes has failed, the router poison the route by assigning an infinite metric to it

4.4.4 Link State Routing

Distance vector algorithm was replaced by an entirely new algorithm called link state routing.

The idea behind link state routing can be stated in five steps. Each router must do the following:

1. Discover its neighbours and learn their network address. Routers learn their neighbours by sending a special HELLO packet on each point-to-point line.

2. Measure the delay or cost to each of its neighbours. The most direct way to determine this delay is to send over the line a special ECHO packet that the other side is required to send back immediately and measuring the round trip time.

3. Routers constructs a **Link State advertisement** packet with all the information

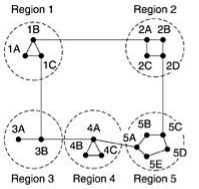
4. Send this packet to all other routers using **flooding**. And also receive information from all routers (not just neighbours).

5. Runs Dijkstra's algorithm locally to find the shortest path.

4.4.5 Hierarchical Routing

As network grow in size, the router routing tables grow proportionally. At a certain point the network may grow to the point where it is no longer feasible for every router to have an entry from every other router.

When hierarchical routing is used, the routers are divided into what we will call **regions**, with each router knowing all the details about how to route packets to destinations within its own region, but knowing nothing about the internal structure of other regions.



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

When routing is done hierarchically, there are entries for all the local routers as before, but all other regions have beed condensed into a single router, so all traffic for region 2 goes via the 1B-2A line.

4.4.6 Broadcast Routing

In some applications, hosts need to send messages to many or all other hosts. Sending a packet to all destinations simultaneously is called **broadcasting**.

One broadcasting method that requires no special features from the subnet is for the source to simply send a distinct packet to each destination.

Another algorithm is **multidestination routing**. Each packet contains a list of destination. When a packet arrives at a router, the router checks all the destinations to determine the set of output lines that will be needed. The router generates a new copy of the packet for each output line to be used and includes in each packet only those destinations that are to use the line. After a sufficient amount of hops, each packet will carry only one destination and can be treated like a normal packet.

The third algorithm makes explicit use of the sink tree for the router initiating the broadcast. If each router knows which of its lines belong to the spanning tree, it can copy an incoming broadcast packet onto all the spanning tree lines except the one it arrived on. Since routers do not know the spanning tree, **reverse path forwarding** is used. The idea is that routers assume that the first copy to arrive at the router is the best route, therefore it will send the copy to all other lines except the one the copy arrives from. After that the router discards all duplicate copies.

4.4.7 Multicast Routing

Sending a message to a group is called **multicasting**. To do multicast routing each router computes a spanning tree covering all other routers. When a process sends a multicast packet to a group, the first router examines its spanning tree and prunes it, removing all lines that do not lead to hosts that are members of the group.

The main issue above is scalability. An alternative design uses **core-based trees** where a single spanning tree per group is computed, with the root near the middle of the group. To send a multicast message, a host sends it to the core, which then does the multicast along the spanning tree.

4.5 Internetworking

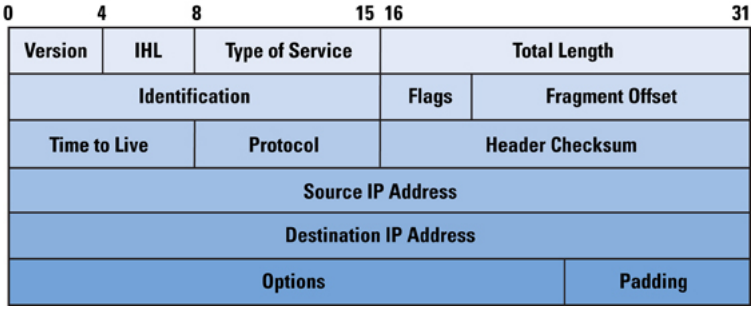
Until now, we have implicitly assumed that there is a single homogeneous network, with each machine using the same protocol in each layer. Unfortunately, this assumption is highly optimistic. Many different network exist, including **PANs (Personal Area Network)**, **LANs (Local Area Network)**, **MANs (Metropolitan Area Network)** and **WANs (Wide Area Network)**. Numerous protocols are in widespread use in every layer. Problem arise when two or more networks are connected to form an **internet**

Many problems can occur at the interfaces between networks. To start with, when packets from a connection-oriented network transit to a connectionless one, they may be reordered. Protocol conversions will often be needed. Address conversions will also be needed. Passing multicast packets through a network that does not support multicasting requires generating separate packets for each destination. The different maximum packet sizes used by different networks can be a major nuisance. Error, flow and congestion control often differ among different networks.

4.6 The IP Protocol

An IP datagram consists of a header part and a text part. The header has a 20-byte fixed part and a variable length optional part

Figure 1: The IP Header



The **Version** field keeps track of which version of the protocol the datagram belongs to.

Since the header length is not constant, the **IHL** is provided to tell how long the header is.

The **Type of service** field is one of the few fields that has changed its meaning over the years. I was and is still intended to distinguish between difference class of service.

The **Total length** includes everything in the datagram, the header and data. The maximum length is 65535 bytes.

The **Identification** field is needed to allow the destination host to determine which datagram a newly arrived fragment belongs to.

Fragment Offset tells where in the current datagram this fragment belongs. All fragment except the last one in a datagram must be a multiple of 8 bytes.

The **Time to live** field is a counter used to limit packet lifetimes. In practice, it is decremented on each hop and when it hits zero the packet is discarded.

The **Protocol** field tells it which transport process to give it to. TCP is one possibility, but there are also UDP and others.

The **Checksum** verifies the header

The **Source address** and **Destination address** indicate the network number and host number.

4.7 Fragmentation

Each network imposes some maximum size on its packets, **Maximum Transmission Unit (MTU)**. The only solution to the problem is to allow **gateways**, multi-protocol routers, to break up packets into **fragments**, sending each fragment as a separate internet packet.

A numbering system used in internetwork protocol defines an elementary fragment size small enough that the elementary fragment can pass through every network (8 Bytes). When a packet is fragmented, all the pieces are equal to the elementary fragment size except the last one, which may be shorter. The internet header must provide the original packet number, **Identification**, and the number of the first elementary fragment contained in the packet, **Fragment offset**. There must also be a bit indicating that the last elementary fragment contained within the internet packet is the last one of the original packet, **More Flag**.

For example, a packet with total length (Including header length) is 1020B arrives a network with MTU of 512B. Obviously this packet has to be fragmented, and all fragments must be multiples of 8B except for the last one.

Identifier	Fragment Offset	More Flag	Header Length	Total Length
789	0	1	20	508
789	61	1	20	508
789	122	0	20	44

In the first packet, the fragment offset is set to 0 because this packet contains data starting at position 0 of the original datagram. Since MTU is 512B and the header is 20B, this leaves 492B for the fragment. Since all fragments must be a multiple of 8, only 488B can be sent as it is the closest number to 492 divisible by 8.

In the second packet, fragment offset starts at 61 because $488/8 = 61$. It can then can another 488B. And the third packet, there's only $1020 - 20 - 488 - 488 = 24B$ left to send, so including the header length, 44B is sent.

4.8 IP Addresses

Every host and router on the Internet has an **IP address**, which encodes its network number and host number. The combination is unique meaning no two machines on the Internet have the same IP address. All IP address are 32 bits long. It is important to note that an IP address does not actually refer to a host. It really refers to a network interface, so if a host is on two networks, it must have two IP addresses. However, in practice most hosts are on one network and thus have one IP address.

IP address were divided into five categories. This allocation has come to be called **classful addressing**. It is no longer used but references to it in the literature are still common.

Address Class	Bit Pattern of First Byte	First Byte Decimal Range	Host Assignment Range in Dotted Decimal
A	0xxxxxxx	1 to 127	1.0.0.1 to 126.255.255.254
B	10xxxxxx	128 to 191	128.0.0.1 to 191.255.255.254
C	110xxxxx	192 to 223	192.0.0.1 to 223.255.255.254
D	1110xxxx	224 to 239	224.0.0.1 to 239.255.255.254
E	11110xxx	240 to 255	240.0.0.1 to 255.255.255.255

Network numbers are managed by a nonprofit corporation called **ICANN (Internet Corporation for Assigned Names and Numbers)** to avoid conflicts. Network addresses are usually written it **dotted decimal notation**. In this format, each of the 4 bytes is written in decimal from 0 to 255. The lowest IP address is 0.0.0.0 and the highest is 255.255.255.255.

IP addresses with 0 as host number is considered as a network identifier, whereas IP addresses with all 1s as host number is the broadcast address for the network. IP address with 0 as network number refers to the current network.

4.8.1 Subnets

All the hosts in a network must have the same network number. This property of IP addressing can cause problems as the networks grow. Getting a second network address would be hard to do since network addresses are scarce. The solution is to allow a network to be split into several parts for internal use but still act like a single network to the outside world, the parts of the network are called **subnets**.

To implement subnetting, the main router needs a **subnet mask** that indicates the split between network + subnet number and host. A subnet mask can be written in dotted decimal notation such as 255.255.252.0. An alternative notation is /22 to indicate that the subnet mask is 22 bits long.

When an IP packet arrives at a external router, its network address is looked up in the routing table and the packet is forwarded to the next router on the interface given in the table. When the IP packet arrives at a subnet router, the subnet mask is applied to lookup if the destination is on their subnet or if they should forward it to another subnet. If the destination is on their subnet, it is forwarded to the host. Subnetting thus creates a **three level hierarchy** consisting network, subnet and host.

4.8.2 CIDR (Classless InterDomain Routing)

The basic idea behind CIDR is to allocate IP addresses in variable sized blocks, without regard to the classes. If an organisation needs 2000 addresses, it is given a block of 2048 addresses on a 2048-byte boundary.

With CIDR, each routing table entry is extended by giving it a 32-bit mask. There is now a routing table for all networks consisting of an array of (IP address, subnet mask, outgoing line) triples. When a packet comes in, its destination IP address is first extracted. Then the routing table is scanned entry by entry, masking the destination address and comparing it to the table entry looking for a match. It is possible that multiple entries match, in which case the longest mask is used.

For example, the routing table may include two entries (network = 123.4.0.0/16, port = 1) and (network = 123.4.20.0/24, port = 2). If a packet comes in addressed to 123.4.20.11, although it matches both entry, 123.4.20.0/24 is the longer match hence it is forwarded to port 2.

4.8.3 Network Address Translation (NAT)

IP addresses are scarce. The problem of running out of IP addresses is not a theoretical problem that might occur at some point in the distant future. It is happening right now. The long term solution is for the whole internet to migrate to IPv6, which has 128 bit addresses. This transition is slowly occurring, in the mean time **Network Address Translation** is used as a quick fix for the short term.

The idea behind NAT is to assign each company a single IP address for Internet traffic. Within the company, every computer gets a unique IP address, which is used for routing intramural traffic. However, when a packet exits the company and goes to the ISP, an address translation takes place.

Three ranges of IP addresses have been declared as private. Companies may use them internally as they wish. The only rule is that no packets containing these addresses may appear on the Internet. The three reserved ranges are:

10.0.0.0 - 10.255.255.255/8

172.16.0.0 - 172.31.255.255/12

192.168.0.0 - 192.168.255.255/16

Within the company premises, every machine has a unique address of the form 10.x.y.z. However, when a packet leaves the company premises, it passes through a **NAT box** that converts the internal IP source address to the company's true IP address, 198.60.41.12 for example. The NAT box is often combined in a single device with a firewall to provide security by carefully controlling what goes in and out of the company.

When a packet enters the company, the **source port** field of the TCP or UDP header is used for mapping. Whenever an outgoing packet enters the NAT box, the TCP source port field is replaced by an index into the NAT box's 65536 entry translation table. This table entry contains the original IP address and the original source port. Then IP and TCP header checksum are recalculated and inserted back to the packet. When a packet arrives at the NAT box from the ISP, the source port in the TCP header is extracted and used as an index into the NAT box's mapping table. The internal IP address and original TCP source port are extracted and inserted into the packet, the header checksums are recalculated and inserted into the packet.

There are problems with NAT:

1) NAT violates architectural IP model, which states that every IP address uniquely identifies single machine worldwide

2) NAT changes Internet from connectionless to connection-oriented network

3) NAT violates fundamental rule of protocol layering: layer k must not make assumptions about layer k + 1. If TCP does not use source port field, NAT will fail

4) NAT cannot easily support new transport protocols

4.9 Internet Control Protocols

In addition to IP, which is used for data transfer, the Internet has several control protocols used in the network layer.

4.9.1 Internet Control Message Protocol (ICMP)

The operation of the Internet is monitored closely by the routers. When something unexpected occurs, the event is reported by the ICMP, which is also used to test the Internet. Some ICMP messages are defined below:

Internet Control Message Protocol

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

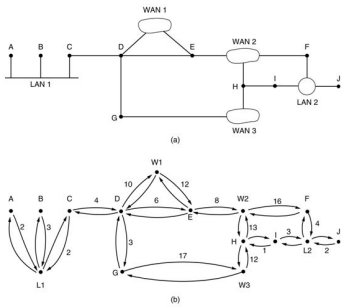
The principal ICMP message types.

4.10 Internet Routing

The internet is made up of a large number of autonomous systems. Each AS is operated by a different organisation and can use its own routing algorithm inside. A routing algorithm within an AS is called an **interior gateway protocol**; an algorithm for routing between ASes is called an **exterior gateway protocol**.

4.10.1 Open Shortest Path First (OSPF)

OSPF – The Interior Gateway Routing Protocol



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

OSPF is a Interior Gateway Protocol. The OSPF routing protocol is also a link state routing protocol that has largely replaced the older **Routing Information Protocol (RIP)** in corporate networks.

Using OSPF, a router that learns of a change to a routing table or detects a change in the network immediately multicasts the information to all other OSPF hosts in the network so they will all have the same routing table information. When routes change, OSPF routers find a new path between endpoints with no loops (which is called ”open”) and that minimises the length of the path.

OSPF bases its path choices on ”link states” that take into account additional network information, including IT-assigned cost metrics that give some paths higher assigned costs.

Many of the ASes in the Internet are themselves large and nontrivial to maintain. OSPF allows them to be divided into numbered **areas**, where an area is a network or a set of contiguous networks. Outside an area, its topology and details are not visible. Every AS has a **backbone** area and all area are connected to the backbone so that it is possible to go from any area to another via the backbone. Within an area, each router has the same link state database and runs the same shortest path algorithm.

4.10.2 Border Gateway Protocol (BGP)

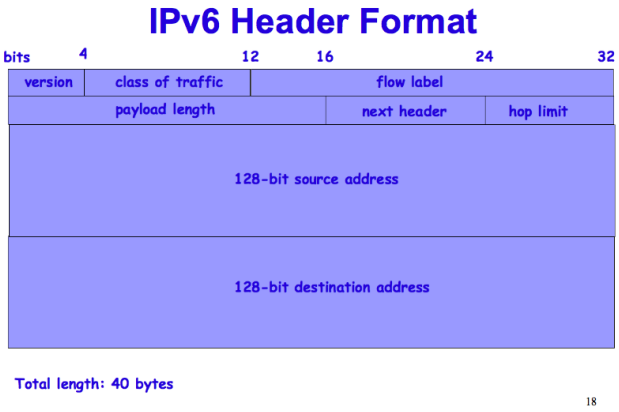
BGP is an Exterior Gateway Protocol. Exterior gateway protocol routers have to worry about politics a great deal therefore polices are typically manually configured into each BGP router.

From the point of view of a BGP router, the world consists of ASes and the lines connecting them. Pairs of BGP routers communicate with each other by establishing TCP connections, this provides reliable communication and hides all the details of the network being passed through.

BGP is fundamentally a distance vector protocol, but quite different from most others. Instead of maintaining just the cost to each destination, each BGP router keeps

track of the path used. Similarly, instead of periodically giving each neighbour its estimated cost to each possible destination, each BGP router tells its neighbours the exact path is it using. This can avoid the count to infinity problem other distance vector protocol might have.

4.11 IPv6



The **Version** field is always 6 for IPv6.

The **Traffic Class** field is used to distinguish between packets with different real-time delivery requirements.

The **Flow Label** is still experimental but will be used to allow a source and destination to set up a pseudo-connection with particular properties and requirements.

The **Payload length** tells how many bytes follow the 40 byte header. This does not include the length of the header, unlike IPv4.

The **Hop limit** is used to keep packets from living forever, same as the **Time to live** field in IPv4. The difference is this field is decremented on each hop, in IPv4 it was time in seconds

The **Source address** and textbfDestination Address are changed to 16 bytes.

The **Fragmentation** fields are removed because IPv6 packets could not be fragmented on the fly during transit across the network. Only the end devices (source and destination) can do fragmentation, so each router could either forward on an IPv6 packet or discard it.

The **Checksum** field is removed to make it more efficient because Transport protocols and the lower layer protocols already provide error-detection features.

5 Data Link Layer

The data link layer has a number of specific functions it can carry out:

1) Providing a well-defined service interface to the network layer

2) Dealing with transmission errors

3) Regulating the flow of data so that slow receivers are not swamped by fast senders

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into **frames** for transmission. Each frame contains a frame header, a payload field for holder the packet, and a frame trailer.

5.1 Services Provided to the Network Layer

The principal service is transferring data from the network layer on the source machine to the network layer on the destination machine. The data link layer can be designed to offer various services. Three reasonable possibilities that are commonly provided are:

1) Un acknowledged connectionless service, which consists of having a source machine send independent frames to the destination machine without having the destination machine acknowledge them. If a frame is lost due to noise on the line, no attempt is made to detect the loss or recover from it.

2) Acknowledged connectionless service. When this service is offered, there are still no logical connections used but each frame sent is individually acknowledged. In this way the sender knows whether a frame has arrived correctly. If is has not arrived within a specified time interval, it can be sent again.

3) Acknowledged connection-oriented service. With this service, the source and destination machines establish a connection before any data are transferred. Each frame sent over the connection is numbered, and the data link layer guarantees that each frame sent is received exactly once in the right order.

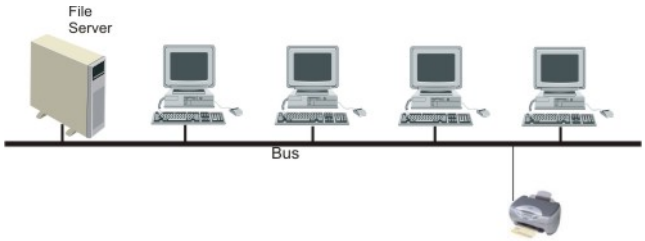
5.2 Framing

The physical layer accepts a raw bit stream and attempt to deliver it to the destination. It is up to the data link layer to detect errors of the bit stream. The usual approach is for the data link layer to break the bit stream up into discrete frames and compute the checksum for each frame. When a frame arrives at the destination, the checksum is recomputed. If the newly recomputed checksum is different from the one contained in the frame, the data link layer deals with the error.

5.3 Network Topologies

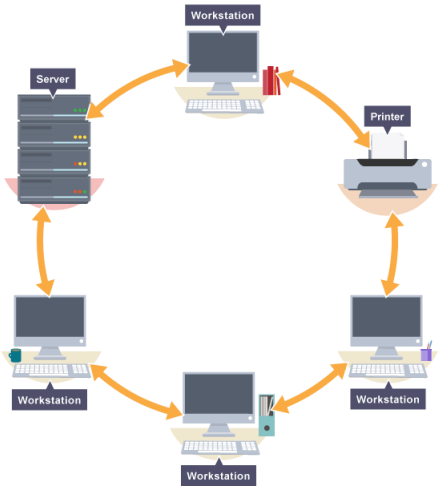
A network topology is the arrangement of a network, including its nodes and connecting lines.

5.3.1 Bus Topology



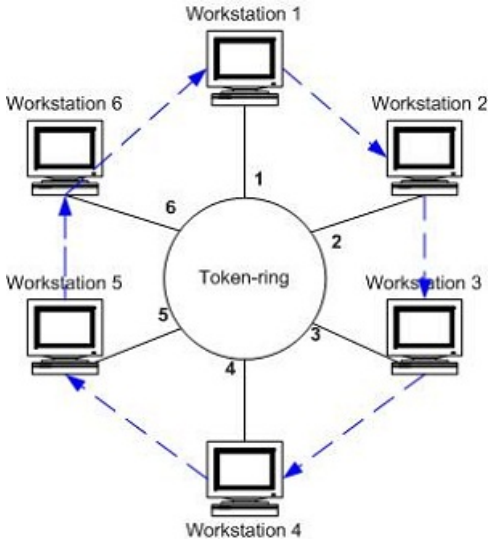
A bus network is an arrangement in a local area network (LAN) in which each node (workstation or other device) is connected to a main cable or link called the bus. A bus network is simple and reliable. If one node fails to operate, all the rest can still communicate with each other. For a major disruption to take place, the bus itself must be broken somewhere. Bus networks are easy to expand. Additional nodes can be added anywhere along the bus.

5.3.2 Ring Topology



A ring network is a local area network (LAN) in which the nodes (workstations or other devices) are connected in a closed loop configuration. Each host is connected to two other devices so each host needs two **Network Interface Card (NIC)**. Each data packet on the network travels in one direction. Each device receives each packet in turn until the destination device receives it. This type of network can transfer data quickly as data only flows in one direction so there won't be any data collisions. However, if the main cable fails or any device is faulty, then the whole network will fail.

5.3.3 Token Ring Topology



A token ring network is a local area network (LAN) in which all computers are connected in a ring or star topology and pass one or more logical tokens from host to host.

This is how token ring works:

Systems in the LAN are arranged in a logical ring; each system receives data frames from its logical predecessor on the ring and sends them to its logical successor. The network may be an actual ring, with cabling connecting each node directly to its neighbours, but more often is a star, with the ring existing only logically in the wiring closet within the ”multiaccess unit” to which all the hosts connect.

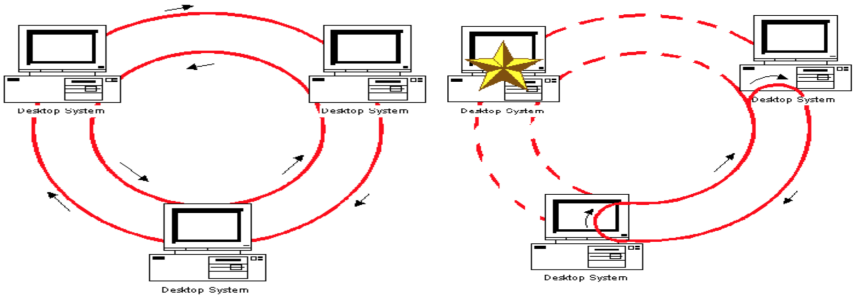
Empty information frames are continuously circulated on the ring, along with frames containing actual data; any node receiving an empty frame and having nothing to send simply forwards the empty frame.

When a computer has a message to send, it waits for an empty frame. It then inserts a token indicating that it is sending data in the frame, the data it wants to transmit into the payload section of the frame and sets a destination identifier on the frame.

When a computer receives a frame containing data (indicated by the token) it knows it cannot transmit data of its own. If this computer is not the sender or receiver, it simply retransmits the frame, sending it to the next host in the ring. If it is the destination for the message, it copies the message from the frame and clears the token to indicate receipt. If it is the sender, it sees that the message has been received, removes the message payload from the frame and sends the empty frame around the ring.

If a host died, the network did not die with it, it just stopped passing the token to the dead host.

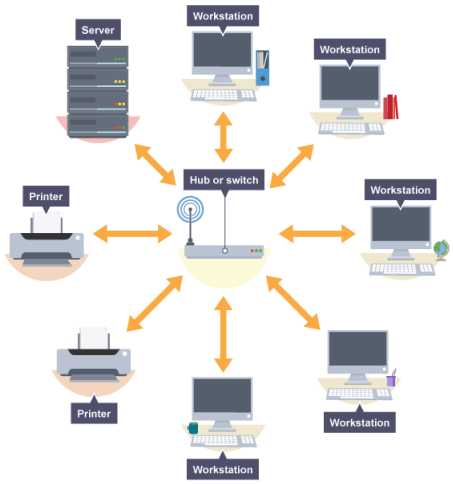
5.3.4 Fiber Distributed Data Interface (FDDI)



The FDDI protocol is based on the token ring protocol. An FDDI network contains two token rings, one for possible backup in case the primary ring fails. The primary ring offers up to 100 Mbps capacity. If the secondary ring is not needed for backup, it can also carry data, extending capacity to 200 Mbps.

If a host dies, it can 'short circuit' two rings together to create a new single ring almost twice as long as original. But all other hosts can still be connected.

5.3.5 Star Topology



A star network is a local area network (LAN) in which all nodes (workstations or other devices) are directly connected to a common central computer. Every workstation is indirectly connected to every other through the central computer. The star network topology works well when workstations are at scattered points. It is easy to add or remove workstations. If any workstation goes down, none of the other workstations will be affected. But if the central computer goes down, the entire network will suffer degraded performance or complete failure.

5.4 Address Resolution Protocol (ARP)

When configuring a new network computer, each system is assigned an Internet Protocol (IP) address for primary identification and communication. A computer also has a unique media access control (MAC) address identity.

MAC Addresses are 'burned' into the Network Interface Card (NIC) and cannot be changed. MAC address is only unique in your network. Duplicate MAC Addresses on the same LAN is a problem. However, duplicate MAC Addresses separated by one or more routers is not a problem since the two devices won't see each other and will use the router to communicate.

However, switches have no idea what IP is. **Address Resolution Protocol (ARP)** is a low-level network protocol for translating network layer addresses into link layer addresses. When an incoming packet arrives at a router for a LAN, the router asks the ARP program to a MAC address that matches the IP address in the ARP cache. If it finds the address, sent to the host. If no entry is found for the IP address, ARP broadcasts a request packet to all the machines on the LAN to see if one machine knows that it has that IP address associated with it. A machine that recognises the IP address as its own returns a reply so indicating. ARP updates the ARP cache and then sends the packet to the MAC address that replied.

5.5 Medium Access Control Sublayer

Networks can be divided into two categories: those using point-to-point connections and those using broadcast channels. In topologies such as Token Ring, hosts are not expected to transmit at the same time. THe protocols used to determine who goes next on a multiaccess channel belong to a sublayer of data link layer called the **MAC (Medium Access Control** sublayer.

5.5.1 Medium Access Strategies

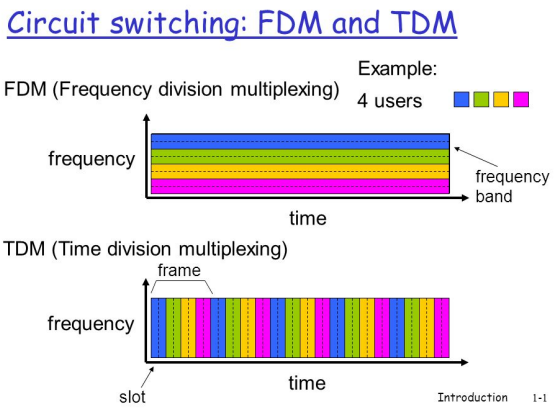
There are different ways to allocate a single broadcast channel among competing users.

1) No control. Simply let stations retransmit after collision.

2) Round-robin. Stations take turns using the channel.

3) Reservations. Station need to obtain "channel reservation" before transmitting.

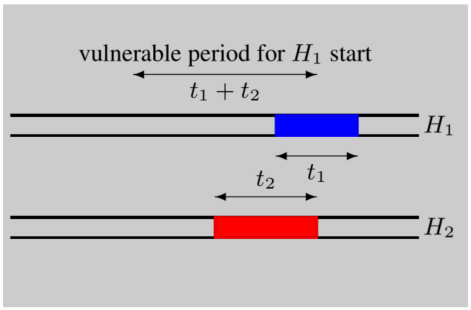
5.5.2 Static Channel Allocation



Frequency-division multiplexing (FDM) is a scheme in which numerous signals are combined for transmission on a single communications line or channel. Each signal is assigned a different frequency (subchannel) within the main channel. FDM presents some problems. If the spectrum is cut up into N regions and fewer than N users are currently interested in communicating, a large piece of spectrum will be wasted. If more than N users want to communicate, some of them will be denied permission for lack of bandwidth.

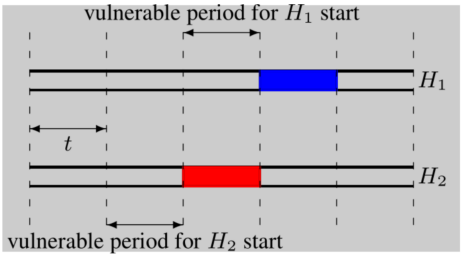
Time-division multiplexing (TDM) is a method of putting multiple data streams in a single signal by separating the signal into many segments, each having a very short duration. Each individual data stream is reassembled at the receiving end based on the timing. Each user is statically allocated every Nth time slot. If a user does not use the allocated slot, it just lies fallow. The transmission rate is limited to R/N where R is the maximum channel rate and N is the number of stations.

5.5.3 Multiple Access Protocols



The basic idea of an **Pure ALOHA** is simple: let users transmit whenever they have data to be sent. The colliding frames will be damaged. However, due to the feedback property of broadcasting, a sender can always find out if a frame was destroyed. If a frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over.

Pure ALOHA suffers low channel efficiency because new frames can easily destroy old frames transmitted.



Slotted ALOHA divides time into discrete intervals, each interval corresponding to one frame. In contrast to Pure ALOHA, a computer is not permitted to send at any time, instead it is required to wait for the beginning of the next slow. One way to achieve synchronisation would be to have one special station emit a pip at the start of each interval, like a clock. This can reduce collision.

Protocols in which stations listen for a carrier and act accordingly are called **Carrier Sense Multiple Access Protocol (CSMA)**.

The first carrier sense protocol is called **1-persistent CSMA**. When a statin has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. Station keeps checking if the channel is free and then transmits immediately.

The second carrier sense protocol is called **Non-persistent CSMA**. Before sending, a station senses the channel. If the channel is already in use, the station does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and than checks again.

The last protocol is **p-persistent CSMA**. When the station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability p. With a probability 1-p, it defers until the next slot.

If two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately. Rather than finish transmitting their frames, they should abruptly stop transmitting as soon as the collision is detected. This protocol is called **CMSA with Collision Detection CSMA/CD**. To avoid poor performance, we can use **binary exponential back-off**. When channel is idle, station may attempt to transmit. If collision occurs, wait either 0 or 1 slots before attempting to transmit again. If another collision occurs, waits 0 or 1 or 2 or 3 slots. After c collisions, choose slot in range 0 to 2c - 1 for next attempt.

6 Physical Layer

The physical layer is the lowest layer depicted in the hierarchy of the OSI model. It defines the mechanical, electrical and timing interfaces to the network.

The purpose of the physical layer is to transport a raw bit stream from one machine to another. Various physical media can be used for the actual transmission. Each one has its own niche in terms of bandwidth, delay, cost and ease of installation and maintenance.

Media are roughly grouped into **guided media** such as copper wire and fibre optics and **unguided media** such as radio and laser.

6.1 Guided Transmission Media

6.1.1 Twisted Pair

One of the oldest and still most common transmission media is **Unshielded twisted pair**. A twisted pair consists of two insulated copper wires, typically about 1mm thick. The wires are twisted together in a helical form, just like a DNA molecule. Twisting the wires can reduce crosstalk or electromagnetic induction between pairs of wires, giving better quality signal over long distances.

Twisted pairs can be used for transmitting either analog or digital signals. The bandwidth depends on the thickness of the wire and the distance travelled.

6.1.2 Coaxial Cable

Another common transmission medium is the **coaxial cable**. It has better shielding than twisted pairs, so it can span longer distances at higher speed.

A coaxial cable consists of a stiff copper wire as the core, surrounded by an insulating material. The insulator is encased by a cylindrical conductor, often as a closely-woven braided mesh. The outer conductor is covered in a protective plastic sheath. The construction and shielding of the coaxial cable give it a good combination of high bandwidth and excellent noise immunity.

6.1.3 Fiber Optics

An optical transmission system has three key components: the light source, the transmission medium and the detector. A pulse of light indicates 1 and the absence of light indicates 0 bit. The transmission medium is an ultra-thin fiber of glass. The detector generates an electrical pulse when light falls on it.

Fiber optics can handle much higher bandwidths than copper. Furthermore, light waves do not suffer from or generate electromagnetic radiation.

6.2 Wireless Transmission

Wireless has advantages for even fixed devices in some circumstances. For example, if running a fiber to a building is difficult due to the terrain (mountains, jungles, swamps), wireless may be better.

The signals are carried in **electromagnetic spectrum**. When electrons move, they create electromagnetic waves that can propagate through space. The number of oscillations per second of a wave is called its **frequency** in **Hz**. The distance between two consecutive maxima is called the **wavelength**. The fundamental relationship between f, λ, c is $c = f\lambda$. When λ is in meters and f is in MHz, $\lambda f = 300$.

Information can be represented by **analogue** or **digital**. In analogue signal, changes in the information are represented in some analogous manner in a physical property of the channel. In digital signal, Information is represented in a discrete set of states, using some coding scheme.

A **modem (modulator-demodulator)** can be used to convert digital signals to analogue signals and vice versa.

6.2.1 Modulation

Modulation is a process through which audio, video, image or text information is added to an electrical or optical carrier signal to be transmitted over a telecommunication or electronic medium.

Modulation enables the transfer of information on an electrical signal to a receiving device that demodulates the signal to extract the blended information. Modulation is achieved by altering the periodic waveform or the carrier. This includes carrying its amplitude, frequency and phase. Modulation has three different types:

Amplitude modulation: the height (i.e., the strength or intensity) of the signal carrier is varied to represent the data being added to the signal.

Frequency modulation: the frequency of the carrier waveform is varied to reflect the frequency of the data.

Phase modulation: Phase of the carrier is modulated to reflect the frequency of the data.

7 Network Security

7.1 Access Control

Access control means that only certain users are allowed access to a resource.

Once early internet system administrators began to understand that they were frequently being attacked, the network firewall was inevitable. Along the way, the firewall has evolved in various ways to produce different types of firewalls.

One thing to note is that all firewalls, regardless of what particular types of firewalls they are, exist to do something that is arguably impossible. They are inserted inline across a network connection and look at all the traffic passing through that point. As they do so, they are tasked with telling what traffic is benign and what is part

of an attack. There cannot exist a computer program that can perfectly predict the outcome of another computer program without running it to see what it does. It is, however, entirely possible to look for known patterns in network packet data that signal attacks that have been seen previously.

7.1.1 Packet filtering firewalls

Packet Filtering mechanisms work in the network layer of the OSI model. In packet filtering, each packet passing through a firewall is compared to a set of rules before it is allowed to pass through. Depending on the packet and the rule, the packet can be either dropped, sent through or a message can be forwarded to the originator. The rules which determine which packets to be sent, and which not to be sent can be based on the source and destination IP address, source and destination port number or the protocol used.

Rule	Dir.	Action	Inside Addr.	Inside Port	Outside Addr.	Outside Port	Description
1	In	Block	*	*	9.9.9.0	*	Don't let these people in
2	In	Allow	*	*	6.6.6.6	*	We trust this host
3	*	Allow	1.1.1.7	300	5.5.5.5	300	Very specific access
4	Out	Allow	1.1.1.1	*	*	*	Allow this inside host access
5	Out	Allow	1.1.1.0	*	4.4.4.3	80	Allow access to this service
6	Out	Block	*	*	*	*	Block anything else

Firewalls use **Access control list ACL**s to filter traffic. By configuring different rules in the ACL you change the behaviour of the firewall. Rules are checked from top to bottom until a match is found.

7.1.2 Circuit-level gateways

The circuit level gateway firewalls work between the transport layer and application layer of the OSI model. They monitor TCP handshaking between the packets to determine if a requested session is legitimate.

When a message is sent out, it uses **Network Address Translation** where the private IP addresses originating from the different clients inside the network are all mapped to the public IP address available through the internet service provider and then sent to the outside world (Internet). So, there is no way for a remote computer or a host to determine the internal private ip addresses of an organization.

7.1.3 Application-level gateways

An **application gateway** is an application program that runs on a firewall system between two networks. Application level firewalls can look in to individual sessions and decide to drop a packet based on information in the application protocol headers or in the application payload. For example, SMTP application proxies can be configured to allow only certain commands like helo, mail from:, rcpt to: etc. to

pass through the firewall and block other commands like expn, vrfy etc.

When a client program establishes a connection to a destination service, it connects to an application gateway, or proxy. In effect, the proxy establishes the connection with the destination behind the firewall and acts on behalf of the client, hiding and protecting individual computers on the network behind the firewall. Once connected, the proxy makes all packet-forwarding decisions. Since all communication is conducted through the proxy server, computers behind the firewall are protected.

7.1.4 Stateful inspection firewalls

Stateful inspection firewalls combine the aspects of the other three types of firewalls. They filter packets at the network layer, transport layer and the application layer.

A stateful inspection firewall also monitors the state of the connection and compiles the information in a state table. Because of this, filtering decisions are based not only on administrator-defined rules (as in static packet filtering) but also on context that has been established by prior packets that have passed through the firewall.

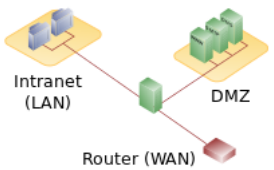
As an added security measure against port scanning, stateful inspection firewalls close off ports until connection to the specific port is requested.

7.1.5 Demilitarized zone (DMZ)

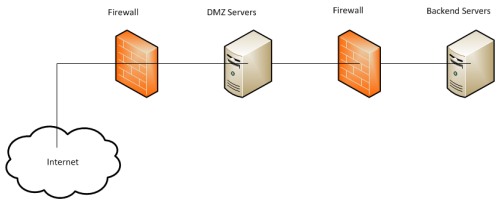
DMZ (demilitarized zone) is a physical or logical sub-network that separates an internal local area network (LAN) from other untrusted networks, usually the Internet.

External-facing servers, resources and services are located in the DMZ so they are accessible from the Internet but the rest of the internal LAN remains unreachable. This provides an additional layer of security to the LAN as it restricts the ability of hackers to directly access internal servers and data via the Internet.

There are many different ways to design a network with a DMZ. Two of the most basic methods are with a single firewall, also known as the three legged model, and with dual firewalls.



A single firewall with at least 3 network interfaces can be used to create a network architecture containing a DMZ. The external network is formed from the ISP to the firewall on the first network interface, the internal network is formed from the second network interface, and the DMZ is formed from the third network interface.



A more secure approach is to use two firewalls to create a DMZ. The first firewall also called the perimeter firewall is configured to allow traffic destined to the DMZ only. The second or internal firewall only allows traffic from the DMZ to the internal network.

7.1.6 Ways to Get Around Firewall

If ssh is allowed, you can use it to tunnel through a firewall

One could potentially ”spoof” a MAC address

One could also attempt to ”spoof” an IP address

7.2 Cryptography

7.2.1 Secret Key Encryption

A secret key algorithm (sometimes called a symmetric algorithm) is a cryptographic algorithm that uses the same key to encrypt and decrypt data.

A very simple example of how a secret key algorithm might work might be substituting the letter in the alphabet prior to the target letter for each one in a message. The resulting text - ”gdkkn” for example - would make no sense to someone who didn’t know the algorithm used (x-1), but would be easily understood by the parties involved in the exchange as ”hello”.

7.2.2 Public Key Encryption

Public-key cryptography, or asymmetric cryptography, is an encryption scheme that uses two mathematically related, but not identical, keys - a public key and a private key. Unlike symmetric key algorithms that rely on one key to both encrypt and decrypt, each key performs a unique function. The public key is used to encrypt and the private key is used to decrypt.

It is computationally infeasible to compute the private key based on the public key. Because of this, public keys can be freely shared, allowing users an easy and convenient method for encrypting content. The private keys can be kept secret, ensuring only the owners of the private keys can decrypt content.

7.2.3 Diffie-Hellman key exchange

1) Bob and Alice agree on a public value g (generator) and a large prime number p

2) Bob chooses a secret value b and Alice chooses a secret value a

3) They each use their secret value to calculate their public value. $B = (g^b \mod p)$ for Bob and $A = (g^a \mod p)$ for Alice. Then they exchange these (public) values.

4) They then use the other's public value to calculate the shared secret key. Bob: $A^b \mod p = g^{ab} \mod p$. Alice: $B^a \mod p = g^{ba} \mod p$. They can use this result/key to communicate securely.