Summary - Computer Networks

Speierer Sébastien

December 14, 2013

Contents

1 Computer networks and the Internet

1.1 The Network Edge

Hosts (or end systems) run applications and are connected together by a network of communication links and packet switches. Host are sometimes divided into two categories: **clients** and **servers**

1.1.1 Access Networks

- **DSL** (digital subscriber line) is obtained from the same local telephone company. It **uses the existing telephone line**. The modem takes digital data and translates it to high-frequency tones for transmission over telephone wires.
- **Cable** uses the existing cable **television infrastructure**.
- **FTTH** (fiber to the home) is a up-and-coming technology that promises even higher speeds.

1.2 The Network Core

The messages exchanged by the end system's applications are broken into smaller chunks of data called **packets**. Between source and destination, each packet travels through communication link and **packet switches**. Most packet switches use **store-and-forward** transmission that means that the packet witch must receive the entire packet before it can begin to transmit the first bit of the packet onto the outbound link.

In the case of sending one packet of length L from source to destination over a patg consisting of N links each of rate R, the **end-to-end delay** is:

$$d_{\text{end-to-end}} = N\frac{L}{R}$$

1.2.1 Queuing Delays and Packet Loss

Each packet switch has an **output queue** for each links attached to it. If an arriving packet finds the link busy, it must wait in the output queue. The **queuing delay** is the time that the packet stay in the output queue. If the arriving packet find the the buffer completely full, **packet full** will occur and the packet will be dropped (sometimes, another packet is dropped instead of the arriving ones).

1.2.2 Forwarding Tables and Routing Protocols

Each router has a **forwarding table** that maps destination addresses to that router's outbound links. When a packet arrives at a router, the router examines the address and searches its forwarding table, using this destination address, to find the appropriate outbound link. Internet has a number of special routing protocols that are used to automatically set the forwarding tables.

1.2.3 Circuit Switching

There are two fundamental approaches to moving data through network of links and switches:

- **circuit switching**: The resources along a path are reserved for the duration of the communication session between the end systems. Since a given transmission rate has been reserved for this sender-to-receiver connection, the sender can transfer the data to the receiver at the guaranteed constant rate. A circuit in a link is implemented with either **frequency-division multiplexing(FDM)** or **time-division multiplexing (TDM)**. → real-time applications
- **packet switching**: The resources are not reserved and so, as a consequence, messages may have to wait for access to a communication link. → simpler, more efficient and less costly.

1.3 A Network of Networks

End systems access the Internet through **Internet Service Providers (ISPs)**. Each ISP is in itself a network of packet switches and communication links.ISPs provide a variety of types of network access to the end systems (cable, DSL, cellular,...). ISP can be a telco, cable company, university, company,... The newtwork of networks means that the ISPs must also be interconnected. There is a hierarchy in the ISP network:

1. **Regional ISP**: In any given region, there may be a regional ISP to which the access ISPs in the region connect.
2. **Tier-1 ISPs**: Then, each regional ISP connects to tier-1 ISPs. There are approximately a dozen tier-1 ISP and the most important are AT&T, Sprint, NTT.

Thus, there is customer-provider relationship at each level of the hierarchy. Note that the tier-1 ISPs do not pay anyone as they are at the top of the hierarchy.

In the today's Internet's network, there are some more elements:

- **Points of presence (PoPs)**: a group of one or more touters (at the same location) in the provider's network where customer ISPs can connect into the provider ISP.
- **Multi-home**: Any ISP (except for tier-1 ISPs) may choose to multi-home, that is, to connect to two or more provider ISPs.
- **Peer**: To reduce their costs, a pair of nearby ISPs at the same level of the hierarchy can peer, that is, they can directly connect their networks together to the "upper ISP".
- **Internet Exchange Point (IXP)**: A third-party company can creat an Internet Exchange Point (typically in a stand-alone building with its own switches), which is a meeting point where multiple ISPs can peer together.
- **Content provider networks**: Google is currently one of the leading examples of such a content provider network. It have a huge network of data centers and it attempts to "bypass" the upper tiers of the Internet by peering with lower-tier ISPs. i.e by directly connecting with them or by connecting with them at IXPs. It can so reduces its payments to upper-tier ISPs and also has greater control of how its services are delivered to end users.

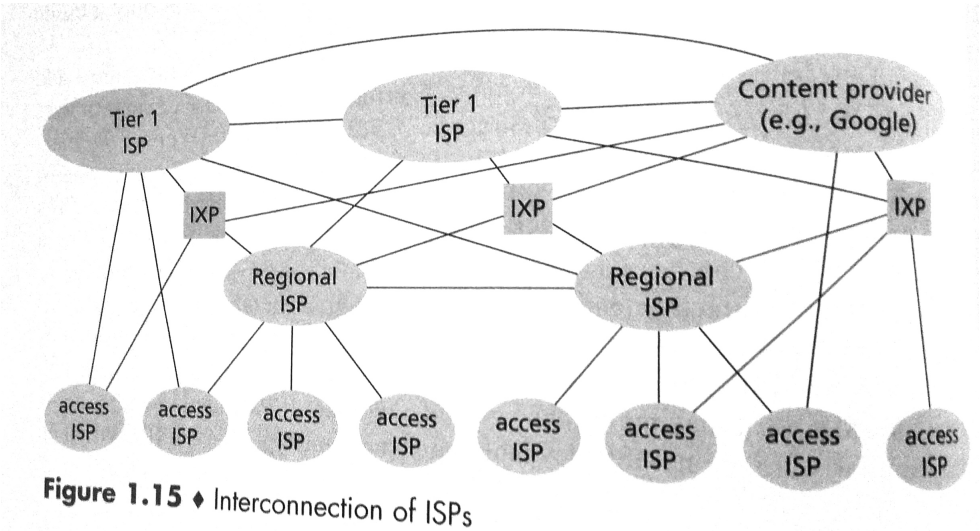

Figure 1.15 ♦ Interconnection of ISPs

1.4 Delay, Loss in Packet-Switched Networks

Along this path, the packet suffers from several types of delays at each node along the path:

1.4.1 Processing Delay

The time required to examine the packet's header and determine where to direct the packet. Contains also the time needed to check for bit-level errors. It's on the order of microseconds or less.

1.4.2 Queuing Delay

Time when the packet waits to be transmitted onto the link. This time depend on the number of earlier-arriving packets. It's on the order of microseconds in practise. The **traffic intensity** often plays an important role in estimating the queuing delay:

$t = La/R \Rightarrow$ R is the rate, a is the average packet rate (packet/sec); L the length of one packet (bits)

1.4.3 Transmission Delay

This is the amount of time required to push all of the packet's bits into the link. It's on the order of micro or milliseconds.

$$d_{trans} = \frac{L}{R} \text{ with R the rate in bits/sec and L the length of the packet in bits}$$

1.4.4 Propagation Delay

The time required to propagate from the beginning of the link to the next router. Depends on the physical medium of the link and the distance. It's on the order of milliseconds.

1.4.5 Packet Loss

The queue preceding a link has finite capacity and so it's not capable of holding an infinite number of packets. A packet can arrive to find a full queue, then the router will drop that packet and the packet will be lost.

1.4.6 End-to-End Delay

Suppose there are $N-1$ routes between the source host and the destination host, the end-to-end delay will be: (here we suppose that the queuing delay is negligible)

$$d_{end-to-end} = N(d_{proc} + d_{trans} + d_{prop})$$

1.4.7 Throughput in Computer Networks

The **instantaneous throughput** at any instant of time is the rate (in bits/sec) at which the destination host is receiving the file.
Lets R_i denote the rate of the link between two node on the path. For a n-link network, the throughput is $\min R_1, R_2, ...R_n$, that is, it is the transmission rate of the **bottleneck link**.

1.5 Protocol Layers and their Service Models

To provide structure to the design of network protocols, network designers organize protocols in **layers** and it has conceptual and structural advantages. Modularity makes it easier to update system components.

1.5.1 Application Layer

It is where network applications and their applocation-layer protocols reside such as HTTP, SMTP, FTP, DNS,...
An application-layer protocol is distributed over multiple end systems, using to exchange packets of information called **message**.

1.5.2 Transport Layer

It transport application-layer's message between application endpoint. There are two transport protocols, TCP and UDP. A transport-layer packet is called a **segment**.

1.5.3 Network Layer

It is responsible for moving network-layer packets known as **datagrams** from one host to another. It get a segment from the transport-layer and provide to deliver this one to the transport layer in the destination host. It include the celebrated **IP Protocols**.

1.5.4 Link Layer

To move a packet from one node (host or router) to the next node in the route, the network layer relies on the services of the link layer. Examples of link-layer protocols include Ethernet, WiFi,... The link-layer packet is called **frames**.

1.5.5 Physical Layer

The job of the physical layer is to move the individual bits within the frame from one node to the next. i.e: twisted-pair copper wire, coaxial cable, fiber,...

1.6 Networks Under Attack

For introduction, just some key words:

- A **malware** can infects our device and do all kinds of devious thigs, including deletin our files, collects our private informations,...
- A **botnet** is a network of thousands of similarly compromised devices (with a malware), which the bad guys control and use fro spam or other kinds of attack.
- A **self-replicating malware** infects one host, and then, from this host, try to infect other hosts.
- A **denial-of-service (DoS) attack** renders a network, host, unusable by legitimate users. i.e: The attacker sends a deluge of packets to the targeted host.
- A passive receiver that records a copy of every packet that flies by is called a **packet sniffer**.
- The ability to inject packets into the Internet with a false source address is known as **IP spoofing**.

2 Application Layer

2.0.1 Client-server architecture

In a client-server architecture, there is an always-on host, called the server, which services requests from many other hosts, called clients. Another characteristic of the client-server architecture is that the server has a fixed, well known address called an IP address.

2.0.2 Processes Communicating

In jargon of operating systems, it is not actually programs but **processes** that communicate. A process sends messages into, and receives messages from, the network through a software interface called **socket**. So a socket is the interface between the application layer and the transport layer. It also referred to as the **Application Proframming Interface (API)** between the application and the network.
In the internet, the host is identified by a 32-bits quantity called IP address. In addition to knowinf the address of the host to which a message is destined, the sending process must also identify the receiving process running in the host and it is the destination **port number** that serves this purpose.

2.1 The Web and HTTP

The **HyperText Transfer Protocol (HTTP)**, the Web's application-layer protocol is at the heart of the Web. It is implemented in two programs: a client program and a server program. HTTP defines how Web clients request Web pages from Web server and how servers transfer Web pages to clients.

HTTP uses **TCP** and it is important to note that the HTTP is said to be a **stateless protocol** because it maintains no information about the client.

In its default mode, HTTP uses **persistent connections** and it means that the server leaves the TCP connection open after sending a response. But it can also be configured to use non-persistent connections.

and here are the steps of the protocol:

1. The HTTP client first initiates a TCP connection with the server.
2. The client sends HTTP request messages into its socket interface:

   ```
   GET /somedir/page.html HTTP/1.1
   Host: www.someschool.edu
   Connection: close
   User-agent: Mozilla/5.0
   Accept-language: fr
   ```

   Notice that GET is not the only method. There exists also POST,PUT,HEAD,DELETE,...
3. Then the server answer with the following message:

   ```
   HTTP/1.1 200 OK
   Connection: close
   Date: Tue, 09 Aug 2011 15:44:04 GMT
   Server: Apache/2.2.3 (CentOS)
   Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT
   Content-Length: 6821
   Content-Type: text/html

   (data data data data ...)
   ```

   There exist also different answers:

   - 200 OK: Request succeeded

- 301 Moved: Requested object has been permanently moved
- 400 Bad Request: Generic error code, the request could not be understood
- 404 Not Found: The requested document does not exist
- 505 HTTP Version Not Supported

2.1.1 Cookies

It is often desirable for a Web site to identify users and for this purpose, HTTP uses cookies. It allow sites to keep track of users. Cookies technology has four components: a cookie request message a cookie file kept on the user's end system and managed by the user's browser and a back-end database at the Web site.
When a new user make a request on the server, this one creates a unique identification number and creates an entry in its back-end database. Then it responds to the client's browser, including a Set-cookie:, which contains the identification number.

Set-cookie: 1678

Then, each time the client requests a Web page on this server, the client's browser puts a cookie header line that includes the identification number in the HTTP request.

Cookie: 1678

In this manner, the server is able to track client's activities on the Web site.

2.1.2 Web caching

A **web cache**, also called a **proxy server**, is a network entity that satisfies HTTP requests on the behalf of an origin Web server. The Web cache has its own disk storage and keeps copies of recently requested objects in this storage. The Web cache will always first verify that its object is up to date. For that, it uses the **conditional GET** by includes ab "If-modified-since: date..." in the HTTP request to the origin server. Then the origin server responds "Not Modified" if it is the case, otherwise, it send the new version of the object. So the Web cache sends:

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
If-modified-since: Wed, 7 Sep 2011 09:23:24
```

And the origin server responds :

```
HTTP/1.1 304 Not Modified
Date: Tue, 15 Oct 2011 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)

(empty)
```

if the file hasn't been modified. So when a browser is requesting a object, here is what happens:

1. The browser establishes a TCP connection with the Web cache and sends the request
2. The Web cache checks if it has a copy of the object. If yes, it sends a conditional GET to the origin server, otherwise, it it just request the file to the origin server.
3. If the object hasn't been modified, it return the object stored in his local storage. Otherwise, it receives the object from the origin server, it stores a copy in its local storage and sends a copy to the client browser.

2.2 File Transfer: FTP

The FTP application-layer protocol allow to transfer file to or from a remote host. In order for the user to access the remote account, the user must provide a user identification and a password.
FTP uses **two parallel TCP** connections to transfer a file, a **control connection** and a **data connection**.
Here are some of the most common commands:

- USER username: Used to send the user identification to the server
- Pass password: Used to send the user password to the server
- LIST: Used to ask the server to send back a list of all the files in the current remote directory.
- RETR filename: Used to retrieve a file from the current directory of the remote host.
- STOR filename: Used to store a file into the current direxTory of the remote host.

And here are the possible replies:

- 331 Username OK, password required
- 125 Data connection already open; transfer starting
- 425 Can't open data connection
- 452 Error writing file

2.3 Electronic Mail in the Internet

In the Internet Mail system, there are three major components: **user agents, mail servers** and the **Simple Mail Transfer Protocol (SMTP)**.
A typical message starts its journey in the sender's user agent, travels to the sender's mail server, and travels to the recipient's mail server, where it is deposited in the recipient's mailbox. It the sender's mail server can not deliver the mail to recipient's mail server then the sender's mail server holds the message in a **message queue** and attempts to transfer the message later.

2.3.1 SMTP

SMTP is the principal application.layer protocol for Internet electronic mail. It uses TCP has two sides: a client side, which executes on the sender's mail server, and a servers side, which executes on the recipient's mail server. Both the client and server sides of SMTP run on every mail server.
Let's walk through a common scenario:

1. A invokes her user agent for e-mail, provides B's e-mail address, composes a message. and instructs the user agent to send the message.
2. A's user agent sends the message to her mail server, where it is placed in a message queue.
3. The client side of SMTP, runninf on A's mail server, sees the message in the message queue. It opens a TCP connection to an SMTP server, running on B's mail server.
4. After some initial SMTP handshaking the SMTP client sends A's message into the TCP connection.
5. At B's mail server, the server side of SMTP receives the message. B's mail server the places the message in B's mailbox.
6. B invokes his user agent to read the message at his convenience.

Now let's look at an example transcript of messages exchanged between an SMTP client (C) and an SMTP server (S):

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr ...Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

And here are the Mail Message Formats header:

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Searching the meaning of life
```

2.3.2 Mail Access Protocols

Once SMTP delivers the message from A's mail server to B's mail server, the message is placed in B's mailbox. There are currently a nomber of popular mail access protocol allow B to access and read his mails in the mailbox on his mail server:

- **POP3**
  POP3 is an extremely simple mail access protocol. First the user agent opens a TCP connection to the mail server on port 110. Then it authenticate the user. Next, the user agent can retrieve message, mark them for deletion or remove deletion marks. At the end, after the client has issued the *quit* command, the mail server deletes the messages that were marked for deletion.
  The connection is done like this (in the case of something goes wrong, the server responds -ERR):

  ```
  telnet mailServer 110
  +OK POP3 server ready
  user bob
  +OK
  pass hungry
  +OK user successfully logged on
  ```

  Then the connection with the mail's server is done, the method we can use are:

  - *list*: list the elements in the mailbox and give there length
  - *retr* x: retrieve the element number x in the mailbox
  - *dele* x: mark the element number x for deletion
  - *quit*: close the connection and the servers will delete the message with the deletion mark.

- **IMAP**
  The POP3 poses a problem for the nomadic user who would prefer to maintain a folder hierarchy on a remote server that can be accessed from any computer. IMAP protocol was invented to solve this and other problems. The IMAP protocol provides commands to allow users to create folders and move messages from one folder to another on the remote server. IMAP also provides commands that allow users to search remote folders for messages matching specific criteria. Another important feature of IMAP is that it permit a user agent to obtain components of messages. i.e: just the header when there is a low-bandwidth connection.

2.4 DNS - The Internet's Directory Service

Internet hosts can be identify in many ways. One identifier for a host is it's **hostname** like "www.google.com". However, hostnames provide little information avout the location within the Internet of the host and for these reasons, hosts are also identified by so-called **IP addresses**.
The **domain name system (DNS)** is a distributed database implemented in a hierarchy of **DNS servers**, and an application-layer protocol that allows hosts to query the distributed database. It is employed by other application-layer protocols to translate user-supplied hostnames to IP addresses and it uses **UDP** as transport-layer protocol.

DNS provides a few other important services in addition to translating hostnames to IP addresses:

- **Host aliasing**: A host with a complicated hostname (called the **canonical hostname**) can have one or more alias names.
- **Mail server aliasing**: DNS can be invokes by a mail application to obtain the canonical hostname for a supplied alias hostname as well as the I address of the host. It permits a company's mail server and Web server to have identical hostnames.
- **Load distribution**: DNS is also used to perform load distribution among replicated servers, such as replicated Web servers. i.e: Busy sites, such as cnn.com, are replicated over multiple servers, with each server running on a different end system and each having a different IP address.

2.4.1 Hierarchy of DNS servers

The client first contacts one of the root servers, which return IP addresses for TLD servers for the top-level domain (as .com, .ch,...). Then this TLD servers returns the IP address of an authoritative server and this one know the IP address for the hostname searched.

- **Root DNS servers**: In the Internet, there are 13 root DNS servers (in fact more than 247 with the replicated servers).
- **Top-level domain (TLD) servers** These servers are responsible for top-level domains such as com, or, ch,fr,...
- **Authoritative DNS servers**: Every organization with publicly accessible hosts on the Internet must provide publicly accessible DNS records that map the names of those hosts to IP addresses. Companies can also pay to have these records stored in an authoritative DNS server of some service provider.
- **Local DNS servers**: Every ISP has a local DNS server. When a host makes a DNS query, the query is sent to the local DNS server, which acts a proxy, forwarding the query into the DNS server hierarchy.

Notice that the query from the requesting host to the local DNS server is **recursive**, and the remaining queries are **iterative** (the local DNS server talk with each of the level of the hierarchy).

In the DNS system, **DNS caching** is very exploited to improve the delay performance. In a query chain, when a DNS server receives a DNS reply, it can cache the mapping in its local memory. Later, if the same query appear, this server would be able to provide the desired IP addresse, even if it is not authoritative for the hostname.

2.4.2 DNS Records and Messages

The DNS servers store **resource records (RRs)**, a four-tuple that contains the foolowing fields:

(Name, Value, Type, TTL)

TTL (time to live) determines when a resource should be removed from a cache. Then, the meaning of Name and Value depend on Type:

- **Type=A**: Name is a hostname and Value is the IP address for the hostname. (That's the standard type)
- **Type=NS**: Name is a domain and Value is the hostname of an authoritative DNS server that knows how to obtain the IP addresses for hosts in the domain. This record is used to route DNS queries further along the query chain.
- **Type=CNAME**: Name is the alias hostname and Value is the canonical hostname.
- **Type=MX**: Name is an alias hostname and Value is the canonical name of a mail server.

In the DNS protocol, both query and reply messages have the same **format**:

- The first 12 bytes is the header section, which indicate if the message is a query or a reply, some more information and indicate the number of elements in each of the following sections.
- The question section contains information about the query that is being made. (Name and Type)
- The answer section contains the resource records for the name that was originally queried.
- The authority section contains record of other authoritative servers.
- The additional section contains other helpful records

If you want to inserting records into the DNS database (your new company's server),you will call a **registrar**, that is a commercial entity that verifies the uniqueness of the domain name, enters the domain name into the DNS database.

2.5 Peer-to-Peer Applications

In a P2P architecture, there is minimal (or no) reliance on dedicated servers in data center. Instead the application exploits direct communication between pairs of intermittently connected hosts, called **peers**. The peers are not owned by the service provider, but are instead desktop and laptops controlled by users, with most of the peers residing in homes, universities, and offices.

2.5.1 P2P File Distribution

In P2P file distribution, each peer can redistribute any portion of the file it has received to any other peers. We called the distribution time, the time it takes to get a copy of the file to all N peers. The minimum distribution time for P2P is (compare to the client-server architecture):

$$D_{P2P} \geq max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{n} u_i}\right\}$$

$$D_{cs} \geq max\left\{\frac{NF}{u_s}, \frac{F}{d_m in}\right\}$$

where F is the size of the file to transfer, d_{min} is the download rate of the peer with the lowest download rate, u_s is the server upload rate (in first, only the server has the file).

2.5.2 BitTorrent

BitTorrent is a popular P2P protocol for file distribution. In BitTorrent, the collection all peers participating in the distribution of a particular file is called a **torrent**. When a peer first joins a torrent, it has no chunks. Over time it accumulates more and more chunks and while it download chunks, it also uploads chunks to other peers.
Each torrent has an infrastructure node called a **tracker**. When a peer joins a torrent, it registers itself with the tracker and periodically informs the tracker that it is still in the torrent. In this manner, the tracker keeps track of the peers that are participating in the torrent.
When a new peer joins the torrent, the tracker randomly selects a subset of peers and give them IP-address to the new peer.
Periodically, the peers ask each neighbour for the list of the chunks they have. To choose which chunk requests first, the peers use a technique called **rarest first**. The peers will always ask for the rarest chunk contains by a neighbour.
On the other side, when a peer receives to much requests, it give priority to the neighbours that are currently supplying her data at the highest rate (take the four best). These four peers are said to be **unchocked** and the peer recalculate that every 10 seconds.
Every 30 seconds, the peer pick one additional neighbour at random and sends it chunk. This is the **optimistically unchocked**. The random neighbour selection also allows new peers to get chunks, so that they can have something to trade.

2.5.3 Distributed Hash Tables (DHTs)

A distributed hash table (DHT) is a simple database in a P2P network. Each peer will only hold a small subset of the totality of the (key, value) pairs (it's just an example).
To optimise this database, we attribute an identifier to each peer (a number for example). Let's also require each key to be an integer in the same range (for that we will use a hash function). Then we will assign each (key,value) pair to the peer whose identifier is the closest successor of the key. We organize the peer in circle within each peer keeps track of its immediate successor and immediate predecessor. It is called an **overlay network**.
In this system, a search of the closest peer of a key is done recursively. We can also add some "shortcuts" to accelerate the search (i.e: each peer keeps track of some other random peers). In a P2P system, peer can come or go without warning. To overcome this problem, let peers keep track of the first and second successor (and first and second predecessor). That verify periodically that the two successor are efficient. Otherwise, they ask the one more successor to the one left. It is called the **peer churn**.

3 Transport Layer

A transport layer protocol provides for **logical communication** between application processes running on different hosts. It is implement in the end systems but not in network routers. The transport layer packets are known as **segments**.

3.1 Multiplexing and Demultiplexing

The transport layer multiplexing and demultiplexing extends the host-to-host delivery service provided by the network layer to a process to process delivery service for applications running on the hosts. On the host, each processes can have more than one sockets. The transport layer in the receiving host does not actually deliver data directly to a process, but instead to an intermediary socket and so, **each socket has a unique identifier**.

Delivering the data in a transport layer segment to the correct socket is called **demultiplexing**.

The job of gathering data chunks at the source host from different sockets, encapsulating each data chunk with header information to create segments and passing the segments to the network layer is called **multiplexing**.

Each segment have special fields that indicate the socket to which the segment is to be delivered. These fields are the **source port number field** and the **destination port number field**.

A **UDP socket** is a two-tuple consisting of a destination IP address and a destination port number.
A **TCP socket** is identified by a four-tuple: (source IP address, source port number, destination IP address, destination port number)

3.2 Connectionless Transport: UDP

UDP(User Datagram Protocol) provides an unreliable, connectionless service to the invoking application. Like IP, UDP is an unreliable service - it does not guarantee that data sent by one process will arrive intact to the destination process.
UDP takes messages from the application process, attaches source and destination port number fields for the multiplexing/demultiplexing service, adds two other small fields, and passes the resulting segment to the network layer. In the other side, if the segment arrives at the receiving host, UDP uses the destination port number to deliver the segment's data to the correct application process.

UDP is said to be **connectionless** because there is no handshaking between sending and receiving transport layer entities before sending a segment.

Another reasons to better suited to UDP:

- **Finer application-level control over what data is sent**, and when.
- **No connection establishment** and so UDP does not introduce any delay to establish a connection.
- **No connection state**: UDP does not maintain connection state and does not track anything and so server can support many more active clients when the application runs over UDP rather than TCP.
- **Small packer header overhead**: 8 bytes instead of 20 with TCP.

The two important disadvantages of the UDP is that it's an **unreliable service** and that it **does not have any congestion control** that means that it does not prevent connection from swamping the links and routers between communicating hosts with an excessive amount of traffic. However, UDP implements a **checksum** to provide error detection at the transport layer, on an end-end basis.

3.3 Principles of Reliable Data Transfer

With a reliable channel, no transferred data bits are corrupted or lost, and all are delivered in the order in which they were sent.

3.3.1 Reliable Data Transfer over a Channel with Bit Errors

In this case, it could appear so bit errors on the packet but we consider that a packet couldn't be lost. This protocol uses both **positive acknowledgements** ("ACK") and **negative acknowledgements** ("NAK"). Fundamentally, three additional protocol capabilities are required in this protocols:

- **Error detection**: For example the checksum using some extra bits.
- **Receiver feedback**: The positive (ACK) and the negative (NAK) replies are examples of such feedback.
- **Retransmission**: A packet that is received in error at the receiver will be retransmitted by the sender.

This protocol is known as a **stop-and-wait** protocols because the sender will not send a new piece of data until it is sure that the receiver has correctly received the current packet.

Because the acknowledge replies can also be corrupt, we will add a new field to the data packet by putting a **sequence number**. It will allow the receiver to know whether the sender is resending the previously transmitted packet or a new one.

The receiver must also include the sequence number in the acknowledgement replies and so a sender that receives two ACKs for the some packet knows that the receiver did not correctly receive the packet following the packet that is being ACKed twice.

3.3.2 Reliable Data Transfer over a Lossy Channel with Bit Errors

Now we consider that the underlying channel can also lose packets.

The sender does not know whether a data packet was lost, an ACK was lost, or if the packet or ACK was simply overly delayed, In all cases, the action is the same: retransmit. This mechanism requires a **countdown timer**.

3.3.3 Pipelined Reliable Data Transfer Protocols

The fact that it is a stop-and-wait protocol decrease the performance. The solution to this problem is that the sender is allowed to send multiple packets without waiting for acknowledgements. For that, the range of the sequence number must be increased to the number of allowed sent packets. Then, the sender and receiver have to buffer packets.

3.3.4 Go-Back-N (GBN)

the sender is allowed to transmit multiple packets without waiting for an acknowledgement, but is constrained to have no more than some maximum allowable number, N, of unacknowledged packets in the pipeline. N is called the **window size**.

The receiver's actions in GBN are also simple. If a packet with sequence number n is received correctly and is in order, the receiver sends an ACK for packet n. Otherwise, the receiver discards the packet and resends an ACK for the most recently received in-order packet.

3.3.5 Selective Repeat

This protocol implements an individual retransmission that will require that the receiver individually acknowledge correctly received packets.

Here is a summary of how works the Selective Repeat protocol:

- **Checksum**: Used to detect biet errors in a transmitted packet.
- **Timeout**: Used to timeout a packet, possibly because the packet or its ACK was lost within the channel. Because timeout can occur when a packet is delayed bit not lost, or when a packet has been received by the receiver but the receiver-to-sender ACK has been lost, duplicate copies of a packet ,may be received by a receiver.
- **Sequence number**: Used for sequential numbering of packets of data flowing from sender to receiver. Gaps in the sequences numbers of received packets allow the receiver to detect a lost packet. Packets with duplicate sequence numbers allow the receiver to detect duplicate copies of a packet.
- **Acknowledgement**: Used by the receiver to tell the sender that a packet or set of packets has been received correctly. Acknowledgements will typically carry the sequence number of the packet or packets being acknowledged. Acknowledgements may be individual or cumulative, depending on the protocol.
- **Negative acknowledgement**: Used by the receiver to tell the sender that a packet has not been received correctly. Negative acknowledgements will typically carry the sequence number of the packet that was not received correctly.
- **Window, pipe lining**: The sender may be restricted to sending only packets with sequence numbers that fall within a given range. By allowing multiple packets to be transmitted but not yet acknowledgement sender utilization can be increased over a stop-.and-wait mode of operation.

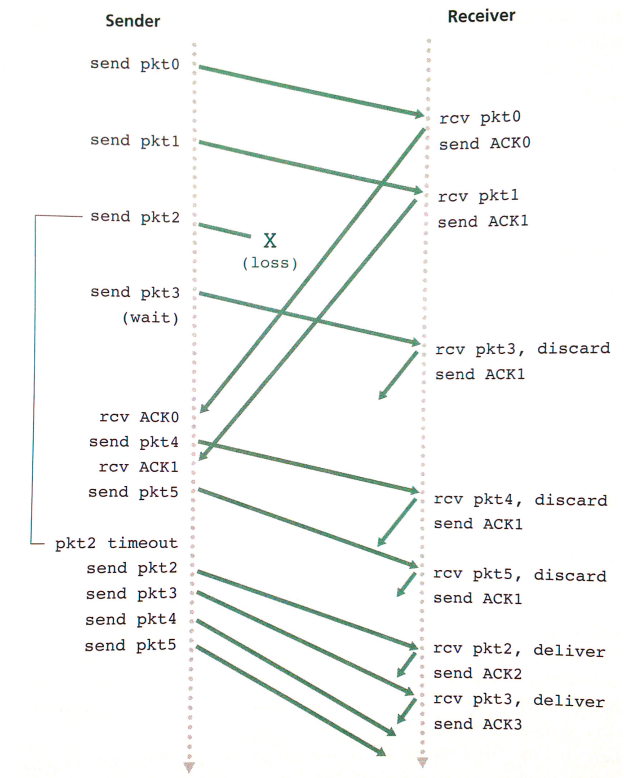

Figure 3.22 ♦ Go-Back-N in operation

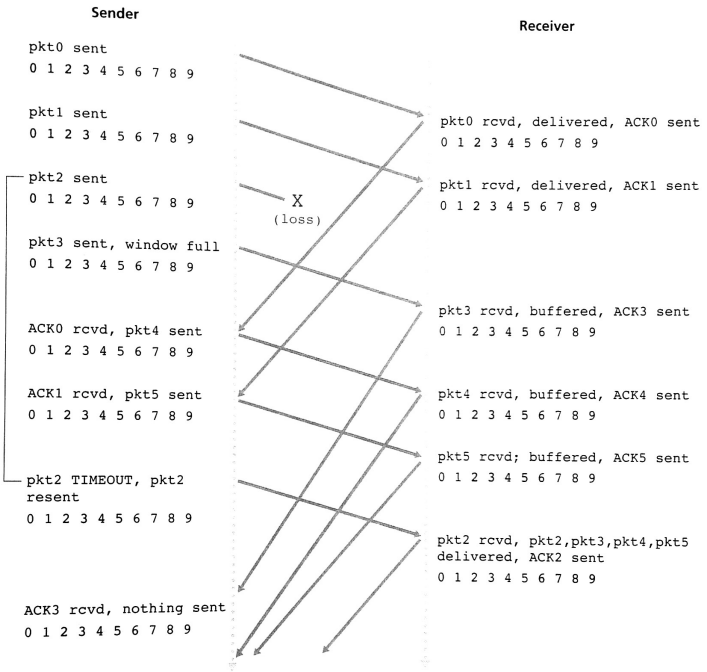

Figure 3.26 ♦ SR operation

3.4 Socket Programming

Socket Programming with UDP:

Here is the code for the client side of the application:

```
from socket import *
serverName = 'hostname'
serverPort = 12000
clientSocket = socket(socket.AF_INET, socket.SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message,(serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print modifiedMessage
clientSocket.close()
```

Here is the code for the server side of the application:

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_DGRAM)
serverSocket.bind('',(serverPort))
print "ready"
while 1:
    message, clientAddress = serverSocket.recvfrom(2048)
    modifiedMessage = message.upper()
    serverSocket.sendto(modifiedMessage, clientAddress)
```

Socket Programming with UDP:

Here is the code for the client side of the application:

```
from socket import *
serverName = 'hostname'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
sentence = "test"
clientSocket.send(sentence)
modifiedSentence = clientSocket.recv(1024)
print modifiedSentence
clientSocket.close()
```

Here is the code for the server side of the application:

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind(('',serverPort))
serverSocket.listen(1)
print "ready"
while 1:
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024)
    capSentence = sentence.upper()
    connectionSocket.send(capSentence)
    connectionSocket.close()
```

3.5 Connection-Oriented Transport: TCP

TCP relies on many principles like error detection, retransmissions,cumulative acknowledgements,timers and header fields for sequence and acknowledgement numbers. TCP is said to be **connection-oriented** because before one application process can begin to send data to another the two processes must first ”handshake” with each other.
TCP connection also provides a **full-duplex service**: flow in the two direction a the same time.

3.5.1 A three-way handshake

1. The client first sends a special TCP segment
2. The server responds with a second special TCP segment
3. Finally the client responds again with a third special segment.

3.5.2 Buffer

TCP has a **send buffer** and a **receive buffer** that allow it to wait before transmit the segment next (on the upper/lower layer).
The maximum amount of data that can be grabbed and placed in a segment is limited by the **maximum segment size (MSS)**.

3.5.3 TCP Segment Structure

The TCP segment consists of header fields and a data field. The header contains the following things:

- source and destination port numbers
- checksum field
- The 32-bit sequence number field and the 32-bit acknowledgement number field
- The 16-bit receive window field used for flow control
- Header length field and the option field
- The 6-bits flag field. It contains the ACK bit, RST ,SYN,FIN (these three are used for connection setup),URG (urgent),PSH (should pass this data to the upper layer immediately)

3.5.4 Sequence Numbers and Acknowledgement Numbers

The **sequence number for a segment** is therefore the byte-stream number of the first byte in the segment.
The **acknowledgement number** that Host A puts in its segment is the sequence number of the next byte Host A is expecting from Host B. Thus, TCP is said to provide cumulative acknowledgements.

3.5.5 Round-Trip Time Estimation and Timeout

After sending element, TCP waits for an ACK until the timeout, but how defines this timeout? Obviously, it should be greater that the RTT. Moreover, since the RTT is not constant, TCP takes samples of RTT and compute the average. Then it add some ”security time” because this average is only an estimation and it depends of the variability of the RTT. Finally the timeout interval is given by:

$$TimeoutInterval = EstimatedRTT + 4 \cdot DevRTT$$

3.5.6 Reliable Data Transfer

There are three major events related to data transmission and retransmission in the TCP sender:

- **Data received from application above:** TCP receives data from the application, encapsulates the data in a segment, and passes the segment to IP. Note that if the timer is already not running for some other segment, TCP starts the timer when the segment is passed to IP.
- **Timer timeout:** TCP responds to the timeout event by retransmitting the segment that caused the timeout.
- **ACK receipt:** On the occurrence of this event, TCP compares the ACK value y with the sequence number of the oldest unacknowledged byte (called $Sendbase$). TCP uses cumulative acknowledgements, so that a acknowledges the receipt of all bytes before byte number y. If $y > Sendbase$, so the sender updates its $Sendbase$ variable and restarts the timer (if there currently are any not-yet-acknowledged segment).

Each time TCP retransmits, it sets the next timeout interval to twice the previous value. However, whenever the timer is started after data received from application above or ACK receipt, the timeout interval is derived from the most recent values of the estimation RTT (see above). In the case that three duplicate ACKs are received, the TCP sender performs a **fast retransmit**, retransmitting the missing segment before that segment's timer expires.

We can say that TCP is an hybrid between the Go-Back-N and the Selective Repeat protocol. Like the GBN, it sends cumulative ACKs and does not ACK individual out-of-order segments. Like the SR, it retransmit only one segment with the oldest un-ACKed sequence number.

3.5.7 Flow control

TCP provides a flow.control service to its applications to eliminate the possibility of the sender overflowing the receivers buffer. It maintain a variable called the receive window that is used to give the sender an idea of how much free buffer space is available at the receiver. Host B tells Host A how much spare room it has in the connection buffer by placing its current value of $rwnd$ in the receive window field of every segment it sends to A.

$$rwnd = RcvBuffer - (LastByteRcvd - LastByteRead)$$

In the case of a zeroWindow, the sender continue to send segment with one data byte. These segments will be acknowledged by the receiver (give maybe a new window size).

3.5.8 Connection Management

The TCP in the client proceeds to establish a TCP connection with the TCP in the server in the following manner:

- The client-side TCP first sends a special TCP segment to the server-side TCP that contains no application-layer data. The SYN bit of this segment is set to 1 and the client randomly chooses an initial sequence number and puts this number in the sequence number field.
- When the server receive it, it allocates the TCP buffers and variables. Then the server sends another segment with the SYN bit to 1, its random sequence number and the first ACK. This is the **SYNACK segment**.
- Receiving the SYNACK segment, the sender allocates buggers and variables. Then it sends another last segment, acknowledges the server's connection-granted with the ACK value (seqNum+1) and the SIN bit is 0.

To end a connection, the client TCP sends a special TCP segments to the server. This special segment has the FIN bit set to 1. When the server receives this segment, is sends the client an ACK segment in return. The server then sends its own shutdown segment, which has the FIN bit set to 1. Finally, the client acknowledges the server's shutdown segment.
There is two famous kinds of attack against the TCP Connection management. First, there is the **SYN flood attack**: it's a classic Denial of Service in which the attacker send a large number of TCP SYN segment, without completing the third handshake step. With this deluge of SYN segments, the servers connection resources become exhausted as they are allocated (buffers and variables). Servers use the SYN cookies to protect them-self against this attack. They use cookies and creates a fully open connection only after the third handshake segment.

3.5.9 Principles of Congestion Control

The approach taken by TCP is to have each sender limit the rate at which it sends traffic into its connection as a function of perceived network congestion. The TCP congestion-control mechanism operating at the sender keeps track of an additional variable, the **congestion window** (*cwnd*) which imposes a constraint of the rate at which a TCP sender can send traffic into the network. The dropped datagram results in a loss event at the sender (timeout or the receipt of three duplicate ACKs) which is taken by the sender to be an indication of congestion on the sender-to-receiver path.

When a TCP connection begins, the value of *cwnd* is typically initialized to a small value (1 MSS). When it receives acknowledgement, the TCP sender increases *cwnd* by one MSS for each of the acknowledged segments. This process results in a doubling of the sending rate every RTT.
If there is a loss event indicated by a timeout, the TCP sender sets the value of the *cwnd* to 1 and it maintain the value $cwnd/2$ ($ssthresh$). Thus, when the value of *cwnd* equals $ssthresh$, TCP increases *cwnd* more cautiously (1 MSS every RTT).
If three duplicate ACKs are detected, (fast retransmit), TCP enters the fast recovery state: the value of *cwnd* is increased by 1 MSS for every duplicate ACK received for the missing segments.
In summary for the TCP Congestion:

- Exponential increase = **slow start**:
  - when sender times out, it reset window to MSS
  - sets congestion threshold to last window size/2
- Linear increase = **congestion avoidance**:
  - window reaches congestion threshold
  - sender receives 3 duplicate ACKs

4 The Network Layer

The role of the network layer is to move packets from a sending host to a receiving host. It provides a single service, known as **best-effort service**. It means that timing between packets is not guaranteed to be preserved, packets are not guaranteed to be received in the order in which they were sent, nor is the eventual delivery of transmitted packets guaranteed.

Forwarding involves the transfer of a packet from an incoming link to an outgoing link within a single router.

Routing involves all of a network's routes, whose collective interactions via routing protocols determine the paths that packets take on their trips from source to destination node.

Every router has a **forwarding table**. A router forwards a packet by examining the value of a field in the arriving packet's header and index it into the right outgoing link using the forwarding table.

4.0.10 Virtual Circuit

Virtual-circuit (VC) networks: computer networks that probide only a connection service at the network layer.

While the Internet is a datagram network, many alternative network architectures are VC. It consist of a path between the source and destination hosts. In a VC network, the network's routers must maintain **connection state information** for the ongoing connections so routers along the pat between the two end systems are involved in VC setup.

4.0.11 Datagram Networks

Datagram networks: computer networks that provide only a connectionless service at the network layer.

Each time an end system wants to send a packet, it stamps the packet with the address of the destination end system and then pops the packet into the network. As a packet is transmitted from source to destination, it passes through a series of routers. Each of these routers uses the packet's destination address to forward the packet.

4.1 Inside a Router

Four router components can be identified:

- **Input ports**: It performs the physical layer function of terminating an incoming physical link at a router. It is also here that the forwarding table is consulted to determine the router output port to which an arriving packet will be forwarded via the switching fabric. The input ports contains the packet buffer.
- **Switching fabric**: The switching fabric connects the route's input ports to its output ports.
- **Output ports**: An output port stores packets received from the switching fabric and transmits these packets on the outgoing link by performing the necessary link-layer and physical-layer functions.
- **Routing processor**: The routing processor executes the routing protocols, maintains routing tables and attached link state information, and computes the forwarding table for the router.

A router's input ports, output ports, and switching fabric together implement the forwarding function while the routing processor implements the routing function.

4.1.1 Switching

The switching fabric is at the very heart of a router, as it is through this fabric that the packets are actually switched from an input port to an output port. It can be accomplished in different ways:

- **Switching via a bus**: The simplest router are traditional computers, with switching between input and output ports being done under direct control of the CPU. When a packet arrives, it is copied into processor memory. Then the routing processor extracted the destination address from the header, looked up the appropriate output port, and copied the packet to the output port's buffers. Many modern routers switch via memory.
- **Switching via an interconnection network**: In this approach, an input port transfers a packet directly to the output port over a shared buy, without intervention by the routing processor. The packet is received by all output ports, but only the port that matches the label will keep the packet. (The router add a prefix (label) with the number of the output for example) Switching via a bus is often sufficient for routers that operate in small local area and enterprise networks.
- **Switching via an interconnection network**: One way to overcome the bandwidth limitation of a single,shared bus is to use a more sophisticated interconnection network. (horizontal and vertical bus with crosspoint) This crossbar network are capable of forwarding multiple packets in parallel, but only one packet can be sent over any given output at a time.

With these different architectures, it is obvious that packet queues may form at both the input ports and the output ports. A consequence of output port queuing is that a **packet scheduler** at the output port must choose one packet among those queued for transmission.

Active queue management (AQM): It's the name of the packet-dropping and -marking policies algorithms

Random Early Detection (RED): One of the most widely studied and implemented AQM algorithm.

Head-of-the-line (HOL) blocking: a queued packet in an input queue must wait for transfer through the fabric because it is blocked by another packet at the head of the line. It is shows that due to HOL blocking, the input queue will grow to unbounded length under certain assumptions as soon as the packet arrival rate on the input links reaches only 58 percent of their capacity.

4.2 The Internet Protocol (IP): Forwarding and Addressing in the Internet

4.3 Datagram Format

Recall that a network-layer packet is referred to as a **datagram**. The IPv4 datagram are the following:

- **Version number**: These 4 bits specify the IP protocol version of the datagram.
- **Header length**: Determine where in the IP datagram the data begins (4 bits)
- **Type of service**: it was included in the IPv4 header to allow different types of IP datagrams to be distinguished from each other. (requiring low delay, high throughput,...)
- **Datagram length**: Total length of the IP datagram.
- **Identifier,flags,fragmentation offset**: These three fields have to do with so-called IP fragmentation.
- **Time-to-live**: Used to ensure that datagrams do not circulate forever in the network.

- **Protocol**: This field is used only when an IP datagram reaches its final destination. It indicates the specific transport-layer protocol.
- **Header checksum**: aids a router in detecting bit errors in a received IP datagram.
- **Source and destination IP addresses**
- **Options**: allow an IP header to b extended.
- **Data**: the data field in the IP datagram contains the transport-layer segment to be delivered to the destination. (It can also carry other types of data, such as ICMP messages)

4.3.1 Fragmentation

The maximum amount of data that a link-layer frame can carry is called the **maximum transmission unit (MTU)**.
Some link-layer protocols can carry big datagrams, whereas others can carry only little packets. To avoid a conflict of datagram length, sometimes IP fragment the data in the IP datagram into two or more smaller IP datagrams, encapsulate each of these smaller IP datagrams in a separate link-layer frame. Fragments need to be reassembled before they reach the transport layer at the destination. (it uses the identification, flag and fragmentation offset field in the IP datagram header) IP fragmentation plays an important role in gluing together the many disparate link-layer technologies.

4.4 IPv4 Addressing

The boundary between the host and the physical link is called an **interface**. A router thus has multiple interfaces, one for each of its links. because every host and router is capable of sending and receiving IP datagrams, IP requires each host and router interface to have its own IP address.

Each IP address is 32 bits long, and there thus a total of 2^{32} possible IP addresses. These addresses are typically written in **dotted-decimal notation**. Thus the address 193.32.216.9 in binary notation is:

11000001 00100000 11011000 00001001

In IP terms, a network interconnecting many host interfaces and many router interfaces forms a **subnet**. IP addressing assigns an address to this subnet i.e: 223.1.1.0/24, where the /24 notation, called a **subnet mask**, indicates that the leftmost 24 bits of the 32-bit quantity define the subnet address.

To determine the subnets, detach each interface from its host or router, creating islands of isolated networks, with interfaces terminating the end points of the isolated networks, Each of these isolated networks is called a ***subnet***

Classless Interdomain Routing (CIDR) is the Internet's address assignment strategy and it generalizes the notion of subnet addressing.
The x most significant bits of an address of the form $a.b.c.d/x$ constitute the network portion of the IP address, and are often referred to as the **prefix** of the address.

There exists another type of IP address, the **IP broadcast address** 255.255.255.255. When a host sends a datagram with this broadcast address, the message is delivered to all hosts on the same subnet.

4.4.1 Obtaining a Host Address: Dynamic Host Configuration Protocol

The **Dynamic Host Configuration Protocol (DHCP)** alloes a host to obtain an IP address automatically. A network administrator can configure DHCP so that a given host receives the same IP address each time it connects to the network, or host may be assigned a **temporary IP address** that will be different each time the sost connects to the network.
For a newly arriving host, the DHCP protocol is a four-step process:

- *DHCP server discovery*: First, the newly host have to find a DHCP server. It is done using a **DHCP discover message**, a UDP packet to port 67 with the destination address of 255.255.255.255 and the source IP address of 0.0.0.0.
- *DHCP server offer*: A DHCP server receive this message and responds to the client with a **DHCP offer message** that is broadcast to all nodes on the subnet and which contains a proposed IP address for the client, the IP address of the DNS server, the IP address of the default gateway router, the network mask and an **IP address lease time** (TTL of the IP address)
- *DHCP request*: The newly arriving client will choose from among one or more server offers and respond to its selected offer with a **DHCP request message**.
- *DHCP ACK*: The server confirms the request of the client.

4.4.2 Network Address Translation (NAT)

If a subnet grew bigger, a larger block of addresses would have to be allocated, But what if the ISP had already allocated all the IP address of the address range? The solution is the **network address translation (NAT)** that allow a subnet to have its own address range which have meaning only within the given subnet. Then, the NAT-router behaves to the outside world as a single device with a single IP address. In essence, the NAT-enabled router is hiding the details of the home network from the outside world. It use a **NAT translation table** to know the internal host to which it should forward a given datagram. When a datagram arrives at the NAT router, the router indexes the NAT translation table using the destination IP address and destination port number to obtain the appropriate IP address and destination port number in the home network.
A problem with the NAT is that it interferes with P2P applications. The essence of the problem is that if Peer B is behind a NAT, it cannot act as a server and accept TCP connections. There exist some tricky solutions to this problems.

4.4.3 Internet Control Message Protocol (ICMP)

ICMP is used by hosts and routers to communicate **network-layer information** to each other. The most typical use is for error reporting and control message. i.e: the well known **ping** program sends an ICMP type 8 code 0 message to the specified host.

4.4.4 IPv6

The most important changes introduced in IPv6 are evident in the datagram format:

- **Expanded addressing capabilities**: increases the size of the IP address from 32 to 128 bits.
- **A streamlined 40-byte headers**: a number of IPv4 fields have been dropped or made optional.
- **Flow labeling and priority**: IPv6 give different treatments to audio-video transmission or to file transfer and e-mail. It can give priority to datagrams from certain applications.

A new version of ICMP has been defiend for IPv6: **ICMPv6**.
Probably the most straightforward way to introduce IPv6-capable nodes is a dual-stack approach, where IPv6 nodes also have a complete IPv4 implementation. These nodes have the ability to send

and receive both IPv4 and IPv6 datagrams. They must have both IPv6 and IPv4 addresses. The problem with this method is that when an IPv6 datagram is converted into a IPv4 datagram, it would lose some of its IPv6-specific fields. A solution of this problem is to use **tunnel** that refer to the intervening set of IPv4 routers between two IPv6 routers. With tunneling, the IPv6 node on the sending side of the tunnel takes the entire IPv6 datagram and puts it in the data field of an IPv4 datagram.

4.5 Routing Algorithms

One of the major topic of this chapter is to determine the path that packets taker from senders to receivers. A graph is used to formulate routing problems. The nodes in the graph represent routers and the edges connecting these nodes represent the physical links between theses routers. Costs are assigned to the carious edges and may reflect the physical length of the corresponding link. We have two type of routing algorithms: global and decentralized:

- A **global routing algorithm** computes the least-cost path between a source and destination using complete, global knowledge about the network. This then requires that the algorithm somehow obtain this information before actually performing the calculation. The calculation itself can be run at one site (a centralized global routing algorithm) or replicated at multiple sites. In practice, algorithms with global state information are often referred to as **link-state (LS) algorithms**. One of these global routing algorithms is **Dijkstra's algorithm**.
- In a **decentralized routing algorithm**, the calculation of the least-cost path is carried out in an iterative,distributed manner. No node has complete information about the costs of all network links. The **Distance-Vector (DV)** routing algorithm is a famous one and it use the **Bellman-Ford algorithm**.

The DV and LS algorithms take complementary approaches towards computing routing. In the end, neither algorithm is an obvious winner over the other.

4.5.1 Hierarchical Routing

As the number of routers becomes large , storing routing information at each of these hosts would clearly require enormous amount of memory. Furthermore, companies want to hide aspects of its network's internal organization from the outside. Both of theses problems van be solved by organizing routers into **autonomous systems (ASs)**, with each AS consisting of a group of routers that are typically under the same administrative control. The routing algorithm running within an AS is called an **intra-AS routing protocol**. It will be necessary to connect ASs to each other and these one or more of the routers in an AS will gave the added task of being responsible for forwarding packets to destinations outside the AS; these routers are called **gateway routers**. These routers runs **inter-AS routing protocol** that is the same in all Internet; **BGP4**.

4.5.2 Intra-AS Routing in the Internet: RIP

The **Routing Information Protocol (RIP)** is a distance-vector protocol that operated in a manner very close to the idealized DV protocol. RIP uses the term **hop**, which is the number of subnets traversed along the shortest path from source router to destination subnet. It is limited to 15, thus limiting the use of RIP to AS that are fewer than 15 hops in diameter.In RIP, routing updates are exchanged between neighbours approximately every 30 seconds using a RIP response message.
If a router does not hear from its neighbour at least once every 180 seconds, that neighbours is considered to be no longer reachable. Then, RIP modifies the local routing table and then propagates this information by sending advertisements to its neighbours.

4.5.3 Intra-AS Routing in the Internet: OSPF

OSPF is typically deployed in upper-tier ISPs whreras TIP is deployed in lower-tier ISPs and enterprise networks. It is a link-state protocol that uses a Dijkstra's least-cost path algorithm. With OSPF, a

router constructs a complete topological map of the entire AS. The router then locally runs Dijkstra's shortest-path algorithm to determine a shortest-path tree to all subnets, with itself at the root node. With OSPF, a router broadcasts routing information to all other routers in the autonomous system, not just to its neighbours. OSPF can provide security : With authentication, only trusted routers can participate in the OSPF protocol within an AS, thus preventing malicious intrudes. An OSPF AS can also be configured hierarchically into areas (like AS in fact).

4.5.4 Inter-AS Routing: BGP4

The **Border Gateway Protocol v4** is the standard inter-AS routing protocol in today's Internet. It provides:

1. Obtain subnet reachability information from neighbouring AS
2. Propagate the reachability information to all routers internal to the AS.
3. Determine "good" routes to subnets based on the reachability information and on AS policy.

A connection between two BGP neighbours must be manually configured between the two routers. They exchange information over semipermanent TCP connections using port 179. This connection is called **BGP seesion**. Furthermore, a BGP seesion that spans two ASs is called an **external BGP (eBGP) seesion**, and a BGP session between routers in the same AS is called an **internal BGP (iBGP) session**. BGP allows each AS to learn which destinations are reachable via its neighbouring ASs. In BGP, destinations are not hosts but instead are CIDRized **prefixe** (subnet of collection of subnets). In BGP, an autonomous system is identified by its globally unique **AS number (ASN)** assigned by ICANN regional registries. When a router advertises a prefix across a BGP session, it includes with the prefix a number of BGP attributes like: AS-PATH and NEXT-HOP.

5 Security in Computer Networks

We can identify the following desirable properties of secure communication:

- **Confidentiality** Only the sender and intended receiver should be able to understand the contents of the transmitted message. This necessarily requires that the message be somehow encrypted.
- **Message integrity** Sender and receiver want to ensure that the content of their communication is not altered, either maliciously or by accident, in transit. Checksumming techniques can be used to provide such message integrity.
- **End-point authentication** Both the sender and receiver should be able to confirm the identity of the other party involved in the communication.
- **Operational security** Almost all organizations today have networks that are attached to the public Internet. Attackers can attempt to deposit worms into the hosts in the network, obtain corporate secrets, map the internal network configurations, and launch DoS attacks. Operational devices such as firewalls and intrusion detection systems are used to counter attacks against an organization's network. A **firewall** sits between the organization's network and the public network controlling packet access to and from the network. An **intrusion detection system** performs "deep packet inspection," alerting the network administrators about suspicious activity.

definition: **Eavesdropping** - sniffing and recording control and data messages on the channel.

5.1 Principles of Cryptography

Cryptographic techniques allow a sender to disguise data so that an intruder can gain no information from the intercepted data. The receiver must be able to recover the original data from the disguised data.

5.1.1 Basic Symmetric Key Cryptography

In **symmetric key systems**, Alice's and Bob's keys are identical and are secret.
There are many encryption and decryption algorithm that use symmetric key:

- **Caesar cipher**: taking each letter in the plaintext message and substituting the letter that is k letters later in the alphabet. In this case, k is the key. Not strong because there only exists 25 possible key.
- **Monoalphabetic cipher**: substitutes ones letter of the alphabet with another letter of the alphabet (unique substitution needed). By statistical analysis of the plaintext language, looking for the letter's frequency, it is relatively easy to break this code.
- **Polyalphabetic encryption**: use multiple monoalphabetic ciphers to encode a letter in a specific position in the plaintext message. (You can use a word)

5.1.2 Block Ciphers

Block ciphers are used in many secure Internet protocols, including PGP (for secure e-mail), SSL (for secure TCP connection), and IPsec (for securing the network-layer transport).
In a block cipher, the message to be encrypted is processed in blocks of k bits. To encode a block, the cipher uses a one-to-one mapping to map the k-bit block of cleartext to a k-bit black of ciphertext. The number of possible mapping for a general k-block cipher is $2^k!$, which is astronomical. But the size of these tables is also gigantic so block ciphers typically use functions that simulate randomly permuted tables (smaller). i.e: The function first breaks a 64-bits block into 8 chunks. So we have 8 different 8-bit square tables. We apply n times these tables on the different chunks, permuting the chunk between each iteration (scrambling). The key for this block cipher algorithm would be the eight permutation tables (assuming the scramble function is publicly known).

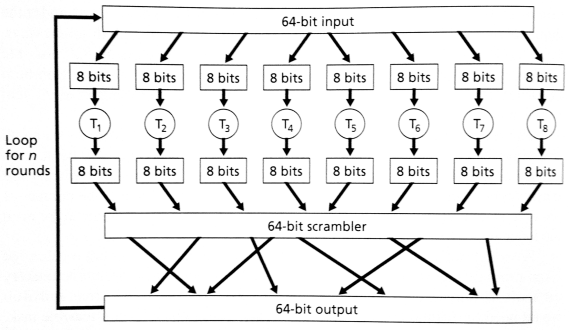

Figure 1: Block cipher algorithm

Now observe that two or more of the cleartext blocks can be identical, and thus an attacker could potentially guess the cleartext. To adress this problem, we can mix some randomness into the cipher. Block cipher typically use a technique called **Cipher Block Chaining (CBC)** which operates as follows:

1. Before encrypting the message, the sender generates a random k-bit string, called the **Initialization Vector (IV)** and sends it to the receiver in cleartext.
2. For the first block, the sender calculates $c(1) = K_s(m(1) \oplus c(0))$ and sends it to the receiver. (K_s is the block cypher, $m(i)$ is the ith block, and $c(0)$ is the IV)
3. For the ith block, the sender generates the ith ciphertext block from $c(i))K_s(m(i) \oplus c(i-1))$.

5.1.3 Public Key Encryption

In **public key systems**, a pair of keys is used. On of the key is known to both and the other key is known only by either Bob or Alice. It resolve the symmetric key encryption's problem that was to first share the securely the key. In the public key encryption, the sender use the public key to encrypt the message and only the receiver can decrypt it because only you must use the private key to decrypt the message. (In some case, it is the sender who has the private key).
RSA has become almost synonymous with public key cryptography. RSA makes extensive use of arithmetic operations using modulo-n arithmetic. We know the identity $(a \bmod n)^d = a^d \bmod n$ To generate public and private RSA keys:

1. Choose two large prime numbers , p and q
2. Compute $n = pq$ and $z = (p-1)(q-1)$
3. Choose a number e, less that n, that has no common factors with z
4. Find a number d, such that $ed \bmod z = 1$.
5. The public key available to the world is the pair of number (n, e) and the private key is the pair of numbers (n, d)

Then the encryption and the decryption are done as follows:

- Suppose Alice want to send the integer number m. Then Alice computes the ciphertext c :

$$c = m^e \bmod n$$

- To decrypt the ciphertext message, Bob computes

$$m = c^d \bmod n$$

These computation are expensive, so in the case someone want to send a large file, he can use the RSA to send securely a symmetric key, and then use it to the message's encryption (i.e: cipher block).

5.2 Message Integrity and Digital Signatures

The authentication of a message needs to verify:

- The message indeed originated from Alice
- The message was not tampered with on its ways to Bob

5.2.1 Cryptographic Hash Functions

A hash function takes an input,m, and computes a fixedsize string $H(m)$ known as a hash. A **cryptographic hash function** is required to have the following additional property:
It is computationally infeasible to find any two different messages x and y such that $H(x) = H(y)$.
MD5 and SHA-1 are two major hash algorithm in use today.

To provide integrity on a message, Alice and Bob shared a secret, the **authentication key** (s). Alice computes $H(m + s)$, called the **message authentication code (MAC)**, and appends it to the message m. Then Bob computes $H(m + s)$ and compare the two hash message.
The problem here is to distribute these shared authentication keys.

5.2.2 Digital Signature

With the **digital signature**, it must be possible to prove that a document signed by an individual was indeed signed by that individual and that only that individual could have signed the document. Public key cryptography will be use for providing digital signature. Bob will use his private key to encrypt the message and send this ciphertext in addition to the original message to Alice. Alice use the public key to decrypt the ciphertext and compare with the original. Since the encryption and decryption are expensive, Bob can just encrypt $H(m)$ that is much smaller than m.

An important application of digital signatures is **public key authentication**, that is, certifying that a public key belongs to a specific entity. So binging a public key to a particular entity is typically done by a **Certification Authority (CA)**, whose job is to validate identities an issue certificates. Once the CA verifies the identity of the entity, the CA creates a certificate that binds public key of the entity to the identity.

5.3 End-Point Authentication

End-point authentication is the process of one entity proving its identity to another entity over a computer network. Typically, an authentication protocol would run before the two communicating parties run some other protocol. What can Bob do to identify Alice:

- verifying that the source address on the IP datagram carrying the authentication message matches Alice's well-known address.
- A secret password can be use to authentication (Facebook, Gmail,...). The password can be encrypt but the intruder can play back (by sniffing) the encrypted password to pretend that he is Alice.
- A **nonce** is a number that a protocol will use only once in a lifetime. Our protocol can use a nonce as follows:
  1. Alice sends the message "I'm Alice" to Bob.
  2. Bob chooses a nonce, R, and sends it to Alice.
  3. Alice encrypts the nonce using Alice and Bob's symmetric secret key and sends it back.
  4. Bob decrypt the message and compare the nonce with the one he sent to Alice.

5.4 Securing E-Mail

A secure e-mail system needs to provide:

- **Confidentiality**: Alice selects a random symmetric session key, encrypt her message with this symmetric key, encrypts the symmetric key with Bob's public key, concatenates the encrypted message and the encrypted symmetric key to form a "package", and sends the package to Bob's e-mail address.
- **Sender authentication and message integrity**: Alice applies a hash function to her message to obtain a message digest, signs the result of the hash function with her private key to create a digital signature, concatenates the original message with the signature to create a package, and sends the package to Bob's e-mail.

To be sure that Alice have the Bob's public key, she can certify the public key using a CA. Now to provide both of the above points, this can be simply done by combining the procedures.

Pretty Good Privacy (PGP) is an e-mail encryption scheme that has become a standard. It provide everything, even the public key certification.

5.5 Securing TCP Connections: SSL

Secure Sockets Layer (SSL) is supported by all popular Web browsers and Web servers, and it is used by essentially all Internet commerce sites. It has three phases:

1. **Handshake**: Bob needs to establish a TCP connection with Alice, verify that Alice is really Alice, and send Alice a master secret key, which will be used by both Alice and Bob to generate all the symmetric keys they need for the SSL session.
2. **Key derivation**: From the master secret, they both generate four keys: encryption and MAC key for the two directions.
3. **Data transfer**: SSL breaks the data stream into records, appends a MAC to each record for integrity checking, and then encrypts the record+MAC. To avoid a reordering or replaying of the records, SSL maintains a sequence number counter and increment it for each SSL record he sends. It includes the sequence number in the MAC calculation.

5.6 Network-Layer Security: IPsec and Virtual Private Networks

With a **Virtual Private Network (VPN)**, the institution's inter-office traffic is sent over the public Internet rather than over a physically independent network. But to provide confidentiality, the inter-office traffic is encrypted before it enters the public Internet.

In the IPsec protocol suite there are two principal protocols: the **Authentication Header (AH)** protocol and the **Encapsulation Security Payload (ESP)** protocol. The AH protocol provides source authentication and data integrity. The ESP protocol provides source authentication, data integrity, and confidentiality. The ESP protocol is much more widely used than the AH protocol.
Before sending IPsec datagrams from source to destination. the source and destination entities create a network-layer logical connection, called a **security association (SA)**. Routers will maintain state information abour SAs, which will include a identifier for the SA called **Security Parameter Index (SPI)**, the origin interface of the SA, the destination interface, keys,...

In summary, IPsec provides between any pair of devices that process packets through the network layer confidentiality, source authentication, data integrity, and replay-attack prevention.

5.7 Operational Security: Firewalls and Intrusion Detection Systems

In computer network, when traffic entering/leaving a network is security-checked, logged, dropped, or forwarded, it is done by operational devices known as firewalls, intrusion detection systems (IDSs), and intrusion prevention systems (IPSs).

5.7.1 Firewalls

A **firewall** is a combination of hardware and software that isolates an organization's internal network from the Internet at large, allowing some packets to pass and blocking others. It has three goals:

- All traffic from outside to inside, and vice versa, passes through the firewall.
- Only authorized traffic, as defined by the local security policy, will be allowed to pass.
- The firewall itself is immune to penetration.

Moreover, firewalls can be classified in three categories:

- **Traditional packet filters**: examine each datagram in isolation determining whether the datagram should be allowed to pass or should be dropped based on administrator-specific rules.
- **Stateful packet filters**: track TCP connection, and use this knowledge to make filtering decisions.
- **Application gateways**: it is an application-specific server through which all application data must pass. Application gateways look beyond the IP/TCP/UDP headers and make policy decisions based on application data. Notice that a different application gateway is needed for each application.

5.7.2 Intrusion Detection Systems

It is a device that examines the headers of all packets passing through it, but also performs **deep packet inspection**, that is, look into the actual application data that the packets carry. When such a devices observers a suspicious packet, it could prevent those packets from entering the organizational network (IPSs)or sends alerts to a network administrator (IDSs).
An IDS can be used to detect a wide range of attackes, including network mapping, port scans, TCP stack scans, DoS, worms and viruses,...
A **signature-based IDS** maintains an extensive database of attack signatures. The signature are normally created by skilled network security engineers who research known attacks.

6 The Link Layer

The basic service of any link layer is to move a datagram from one node to an adjacent node over a single communication link. Possible services that can be offered by a link-layer protocol include:

- **Framing**: All link-layer protocols encapsulate each network-layer datagram within a link-layer frame before transmission over the link.
- **Link access**: A medium access control (MAC) protocol specifies the rules by which a frame is transmitted onto the link. (interesting when multiple nodes share a single broadcast link)
- **Reliable delivery**: it guarantees to move each network-layer datagram across the link without error. It is often used for links that are prone to high error rates, such as wireless link. Many wired link-layer protocols do not provide a reliable delivery service.
- **Error detection and correction**: Because there is no need to forward a datagram that has an error, many link-layer protocols provide a mechanism to detect such bit errors. It is usually more sophisticated than a simple checksum and is implemented in hardware.

The link layer is implemented in a **network adapter**, also sometimes known as a **network interface card (NIC)**. The software components of the link layer implement higher-level link-layer functionality such as assembling link-layer addressing information and activating the controller hardware.

6.0.3 Error-Detection and -Correction Techniques

At the sending node, data, to be protected against bit errors is augmented with error-detection and -correction bits (EDC). Even with the use of error.detection bits there still may be undetected bit errors. Error detection at the link layer is implemented in dedicated hardware in adapters, which can rapidly perform more complex techniques. There exist different techniques for detecting errors:

- **Parity Checks**: In a even parity scheme, the sender simply includes one additional bit an chooses its value such that the total number of 1s in the $d+1$ bits is even.
- **Two-dimensional parity**: The d bits in D are divided into rows and columns, A parity value is computed for each row and for each column. The receiver can thus not only detect the fact that a single bit error has occurred, but can use the column and row indices to identify the bit that was corrupted and correct that error.
- **Checksumming methods**: the d bits of data are treated as a sequence of k-bit integers and use the resulting sum as the error-detection bits.
- **Cyclic redundancy**: An error-detection technique used widely in today's computer networks is based on **cyclic recundancy check (CRC) codes** (also known as polynomial codes). The sender and receiver must first agree on an r+1 bit pattern, known as a **generator (G)**. The most significant bit of G must be a 1. The sender add some bits, R, to D such as it is divisible by G. The receiver divides the d+r received bits by G, and check if the remainder is nonzero.

6.1 Multiple Access Links and Protocols

There are two types of network links:

- A **point-to-point link** consists of a single sender at one end of the link and a single receiver at the other end of the link.
- a **broadcast link**, can have multiple sending and receiving nodes all connected to the same, single, shared broadcast channel.

Computer networks similarly have protocols, called **multiple access protocols**, by which nodes regulate their transmission int o the shared broadcast channel. We can classify any multiple access protocol as belonging to on of three categories:

6.1.1 Channel partitioning protocols

- **Time-division multiplexing (TDM)** divides time into time frames and further divides each time frame into N time slots. Each time slot is then assigned to one of the N nodes. Once everyone had had a chance to talk, the pattern would repeat.
- **Frequency-division multiplexing (FDM)** divides the R bps channel into different frequencies (each with a bandwidth of R/N) and assigns each frequency to one of the N nodes. In the two techniques, a node is limited, even when it is the only node with packets to send.
- **Code division multiple access (CDMA)** assigns a different code to each node. Each node then uses its unique code to encode the data bits it send. If the codes are chosen carefully, nodes can transmit simultaneously and yet have their respective receivers correctly receive a sender's encoded data bits in spit of interfering transmissions by other nodes.

6.1.2 Random access protocols

A transmitting node always transmits at the full rate of the channel. When there is a collision, each node involved in the collision repeatedly retransmits its frame until its frame gets through without a collision. When a node experiences a collision, it waits a random delay before retransmitting the frame.

- **Slotted ALOHA**: When the node has a fresh frame to send, it waits until the beginning of the next slot and transmits the entire frame in the slot. If there is a collision, the node detects the collision before the end of the slot. The node retransmits its frame in each subsequent slot with probability p until the frame is transmitted without a collision.
- **Aloha**: when a frame first arrives, the node immediately transmits the frame in its entirety into the broadcast channel. If a transmitted frame experiences a collision, the node will then immediately retransmit the frame with probability p. Otherwise, the node waits for a frame transmission time. After this wait, it then transmits the frame with probability p, or waits for another frame time with probability 1-p.
- **Carrier Sense Multiple Access (CSMA)**: Two important rules are embodied in the CSMA:
  - *Listen before speaking*: If a frame from another node is currently being transmitted into the channel, a node then waits until it detects no transmissions for a short amount of time and then begins transmission. It is called **carrier sensing**.
  - *If someone else begins talking at the same time, stop talking*: A transmitting node listens to the channel while it is transmitting. If it detects that another node is transmitting an interfering frame , it stops transmitting and waits a random amount of time before repeating the sense-and-transmit-when-idle cycle. It is called **collision detection**.

6.1.3 Taking-turns protocols

Taking-turns protocols have the properties that when only one node is active, the active node has a throughput of R bps, and when M nodes are active, then each active node has a throughput of R/M bps.

- The **polling protocol** requires on of the nodes to be designated as a master node. The master node polls each of the nodes in a round-robin fashion.
- In the **token-passing protocol**, a small, special-purpose frame known as a token is exchanged among the nodes in some fixed order. When a node receives a token, it holds onto the token only if it has some frames to transmit. otherwise, it immediately forwards the token to the next node. If a node does have frames to transmit when it receives the token, it send up to a maximum number of frames and then forwards the token to the next node.

6.2 Switched local Area Networks

Host's and router's network interfaces have link-layer addresses (a router can have multiple link-layer addresses). The job of the link-layer switch is to carry datagrams between hosts and routers; a switch does this job transparently without the host or router having to explicitly address the frame to the intervening switch. A link-layer address is called LAN address or **MAC address**,is 6 bytes long, and is usually fixed for the interface. A laptop with an Ethernet interface always has the same MAC address, no matter where the computer goes.

The job of the **Address Resolution Protocol (ARP)** is to translate between IP addresses and MAC addresses. An ARP module in the sending host takes any IP address on the same LAN (subnet) as input, and returns the corresponding MAC address. Each host and router has an ARP table in its memory, which contains mappings of IP addresses to MAC addresses. If the ARP doesn't know a "translation", it broadcast the network (subnet) with a special packet called an ARP packet. The target system will send back a response ARP packet with the desired matching.

Sending a Datagram off the subnet is more complicated because the sender can't use ARP to determine the MAC address of the destination host. So it first sends the datagram to the first router (using ARP) and putting the IP address of the final destination. Then the router receive it and use the IP address to send the datagram in its correct interface. If one of them is in the final destination's subnet, then it can use the ARP of this subnet to send the datagram.

6.2.1 Ethernet

a **hub** is a physical-layer device that acts on individual bits rather than frames. When a bit, representing a zero or a one, the hub simply re-creates the bit, boosts its energy strength, and transmits the bit onto all the other interfaces. Thus, Ethernet with a hub-based star topology is also a broadcast LAN, whenever a hub receives a bit from one of its interfaces, it send a copy out on all of its other interfaces.

The Ethernet frame structure contains six fields:

- **Data field**: This field carries the IP datagram.
- **Destination address**: This field contains the MAC address of the destination adapter.
- **Source address**: This field contains the MAC address of the adapter that transmits the frame onto the LAN.
- **Type field**: The type field permits Ethernet to multiplex network-layer protocols.
- **CRC**: allow the receiving adapter to detect bit errors in the frame.
- **Preamble**: a 8-byte field use partly to synchronise the sender and receiver clock.

Ethernet technologies provide connectionless and unreliable service to the network-layer and us the CSMA/CD protocol.

6.2.2 Link-layer Switches

The role of the **switch** is to receive incoming link-layer frames and forward them onto outgoing links. **Filtering** is the switch function that determines whether a frame should be forwarded to some interface or should just be dropped. **Forwarding** is the switch function that determines the interfaces to which a frame should be directed, and then moves the frame to those interfaces. Filtering and forwarding are done with a **switch table** that contains (MAC address, interface number, time info).

When a frame arrive at a switch, it looks in its switch table. If the destination MAC address isn't in its switch table, it broadcast the frame. Otherwise, if the interface number is the same that the one from

which it receives the frame, then it dropped the frame. Finally, the switch performs its forwarding function by putting the frame in an output buffer that precede the target's interface.

The switch table is initially empty. For each incoming frame received on an interface, the switch stores in its table the MAC address and the interface number's (and the current time). The switch deletes an address in the table if no frames are received with that address as the source address after some period of time.

A switch that supports **VLANs (virtual local area networks)** allows multiple virtual local area networks to be defined over a single physical local area network infrastructure. Hosts within a VLAN communicate with each other as if they were connected to the switch. In a port-based VLAN, the switch's port (interfaces) are divided into groups by the network manager. Each group constitutes a VLAN, with the ports in each VLAN forming a broadcast domain. It solves the problem of "lack of traffic isolation", "inefficient use of switches" and "managing users".