Chapter 1

1.1 What Is The Internet?

- Internet can be described in terms of:
    - Nuts-and-bolts (hardware and software)
    - Services it provides to applications
    -
- Estimated 2 billion Internet users

1.1.1 A Nuts-and-Bolts Description

- **Hosts / end systems =** devices hooked up to internet
- These days = many nontraditional hosts (fridges, gaming consoles, etc.) instead of just desktop PCs.
    - Term ‘computer network’ starting to sound outdated
- End systems = connected together via **communication links** and **packet switches**
- Different types of communication links (ie: coaxial cable, copper wire, optical fibre, radio spectrum) can transmit data at different rates → transmission rate (bits/second)
- If host wants to send data to another host → sender host segments data and adds **header bytes** to each segment. Each segment with its unique header bytes is a **packet.** The packets are then sent through network from sender host to receiver host, where the segments of the packet are reassembled into the original data (if they all arrive successfully).
    - **Header =** contains info about a piece/segment of data like its source, destination, etc.
    - **Packet =** Segment of a piece of data (ie: a message) which includes *header bytes.*
- **Packet switch:** Takes a packet arriving on one of its incoming communication links and forwards the packet via one of its outgoing communication links
    - 2 most prominent types = **routers** & **link-layer switches** → both forward packets towards their destinations
    - **Link-layer switches** used in **access networks (**connects subscribers to immediate service providers)
    - **Routers** used in **core networks (**connects local providers to each other)
    - **Route/path** = sequence of communication links & packet switches before a packet reaches its destination
- Cisco estimates global internet traffic in 2012 = 40 exabytes per month
- Packet-switched networks (which transport packets) = similar to transportation networks on IRL roads
    - Suppose a factory needs to move cargo to a faraway warehouse:
        - At the factory (source host), the cargo (data/message) is segmented and loaded into a fleet of trucks (packets). Each truck travels independently through a network of roads, highways and intersections to get to the destination warehouse (the destination host). At the

destination warehouse, the cargo segments from all the trucks are unloaded and grouped back together.
        - **Packets = trucks**
        - **Communication links = highways and roads**
        - **Packet switches = intersections**
        - **hosts/end systems = buildings**

- End systems (including content providers like websites) connect to internet through Internet Service Providers (ISPs)
- ISP = network of packet switches and communication links
- Lower-tier ISPs are interconnected through national and international upper-tier ISPs
- Upper-tier ISPs have high-speed routers interconnected with high-speed fibre-optic communication links
- Every ISP (upper/lower-tier):
    - Is managed independently
    - Runs the IP protocol
    - Conforms to naming and address conventions

- **Protocols:** control the sending and receiving of information within the internet
- Two of the most NB protocols = **Transmission Control Protocol (TCP)** and **Internet Protocol (IP)**
    - **IP:** specifies packet formats
    - **TCP:** originated in original network implementation - complemented IP
    - Internet's principal protocols collectively known as: **TCP/IP**
- **Internet Standards:**
    - Everyone must agree on what each and every protocol does in order for internet to work like it does
    - **Internet Engineering Task Force (IETF)** develops internet standards that define protocols such as TCP, IP, HTTP, etc.
        - Documents containing different standards are called **Requests For Comments** (RFCs)
    - **IEEE 802 LAN/MAN Standards Committee** specifies Ethernet and WiFi standards

1.1.2 A Services Description

- From this perspective, internet = infrastructure that provides services to applications
- Applications = email, web browsing, social networking, IMing, VoIP, p2p, etc.
- We call these applications **distributed applications** since they involve many different hosts exchanging data with each other.
- NB to remember that internet applications run on end systems
    - **not** on packet switches in the network core (which connects [local?] ISPs)
- Packet switches facilitate exchange of data, but are not concerned with the applications where the data comes from (sources of data)
- **How does one program running on one end system instruct the internet to send data to another program on another end system?**

- Answer = end systems use/follow an **Application Programming Interface (API)** which specifies how end systems must ask the internet to deliver data.
- **Human analogy to understand APIs:**

draw upon a simple analogy, one that we will frequently use in this book. Suppose Alice wants to send a letter to Bob using the postal service. Alice, of course, can't just write the letter (the data) and drop the letter out her window. Instead, the postal service requires that Alice put the letter in an envelope; write Bob's full name, address, and zip code in the center of the envelope; seal the envelope; put a stamp in the upper-right-hand corner of the envelope; and finally, drop the envelope into an official postal service mailbox. Thus, the postal service has its own "postal service API," or set of rules, that Alice must follow to have the postal service deliver her letter to Bob. In a similar manner, the Internet has an API that the program sending data must follow to have the Internet deliver the data to the program that will receive the data.

- **Ask yourself:** What are routers? What kinds of communication links are present in the Internet? What is a distributed application?

1.1.3 What Is A Protocol?

- **Protocol:** Set of rules governing the format of data sent over Internet / a network
  - *A protocol defines the format and order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other event.*
- The way a protocol works can be understood via analogy. Consider the following human interaction (and assume decent manners are at play).
  - The 'human manners' protocol dictates that conversation is initiated by party 1 with 'Hi', to which party 2 should respond 'Hi' to indicate that the conversation can proceed.
  - Similarly, network protocols require hosts/devices to 'say hi' / ping each other before going through with data exchange.

It is probably easiest to understand the notion of a computer network protocol by first considering some human analogies, since we humans execute protocols all of the time. Consider what you do when you want to ask someone for the time of day. A typical exchange is shown in Figure 1.2. Human protocol (or good manners, at least) dictates that one first offer a greeting (the first "Hi" in Figure 1.2) to initiate communication with someone else. The typical response to a "Hi" is a returned "Hi" message. Implicitly, one then takes a cordial "Hi" response as an indication that one can proceed and ask for the time of day. A different response to the initial "Hi" (such as "Don't bother me!" or "I don't speak English," or some unprintable reply) might indicate an unwillingness or inability to communicate. In this case, the human protocol would be not to ask for the time of day. Sometimes one gets no response at all to a question, in which case one typically gives up asking that person for the time. Note that in our human protocol, *there are specific messages we send, and specific actions we take in response to the received reply messages or other events* (such as no reply within some given amount of time). Clearly, transmitted and received messages, and actions taken when these messages are sent or received or other events occur, play a central role in a human protocol. If people run different protocols (for example, if one person has manners but the other does not, or if one understands the concept of time and the other does not) the protocols do not interoperate and no useful work can be accomplished. The same is true in networking—it takes two (or more) communicating entities running the same protocol in order to accomplish a task.

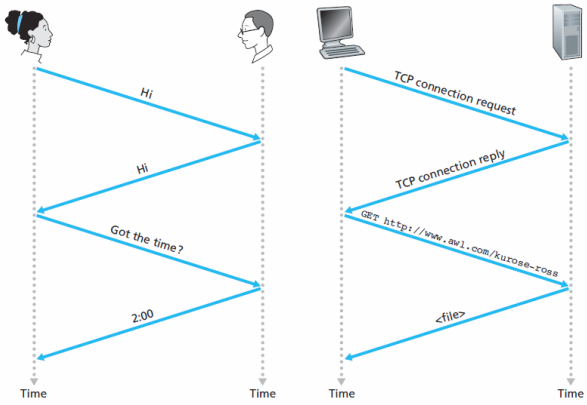


Figure 1.2 ♦ A human protocol and a computer network protocol

- Human analogy to understand network protocols:

- All activity in the Internet that involves multiple communicating remote entities is governed by a protocol
- **Hardware-implemented protocols** in 2 physically connected computers control flow of bits in the connecting wire
- **Congestion-control protocols** in end systems control the rate at which packets are transmitted (transmission rate) from sender to receiver
- **Router protocols** determine a packet's path from source to destination

- Protocol for end systems to to request content from web servers:
    - **End system**: send TCP request
    - **Server**: send TCP reply
    - **End system**: GET http://www.google.com/
    - **Server**: returns the webpage

1.2 The Network Edge

- Computers and other devices connected to Internet called 'end systems' because they sit at the 'edge' of the internet.
- End systems also known as 'hosts' because they *host* (ie: run) application programs such as web browser or email client programs
    - Hosts can be further divided into **clients (users)** and **servers (provide content)**
    - Most content on internet comes from large **data centres**
- **Access network:** network that physically connects an end system to the first router (AKA 'edge router') on a host-to-host data transmission path.

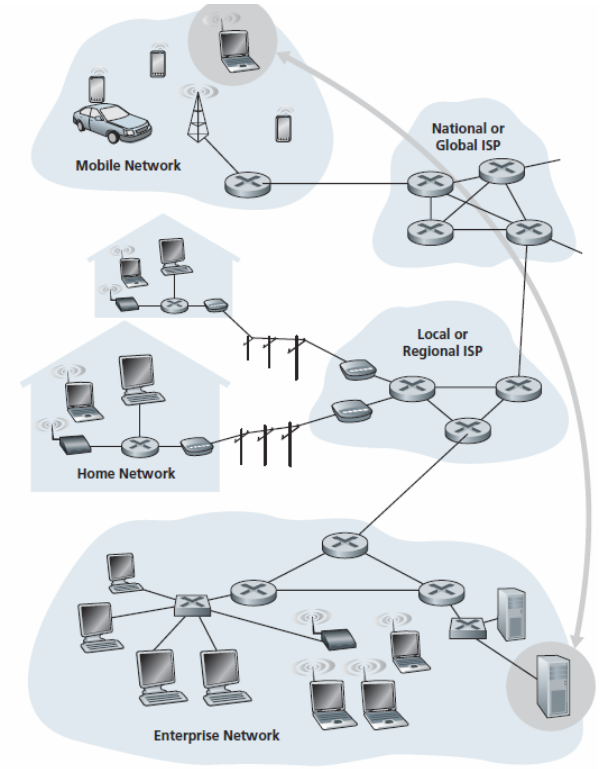


Figure 1.3 ♦ End-system interaction

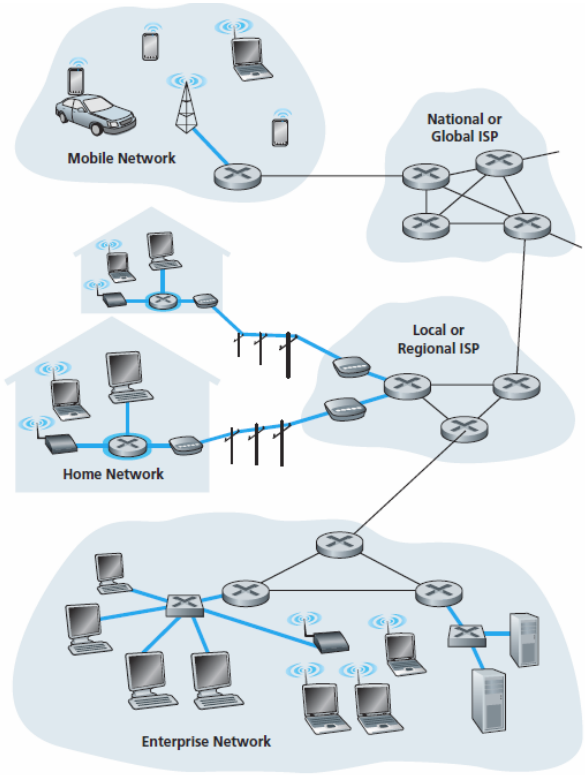


Figure 1.4 ♦ Access networks

- Two most prevalent types of broadband residential access: **DSL** and **Cable (HFC) … (FTTH new and upcoming)**

Digital Subscriber Line (DSL)

- Typically provided by household's local telephone company (telco)
- Thus, if a household has DSL, its ISP is its telco
- A DSL modem uses the existing telephone line (twisted-pair copper wire) to exchange data with a **Digital Subscriber Line Access Multiplexer (DSLAM)** located in the telco's **Local Central Office (CO).**
- DSL modem takes digital data and translates it to high frequency tones (analog signals) for transmission over telephone wires to the CO; these analog signals are translated back into digital format at the DSLAM.
- The residential telephone line carries both data and traditional telephone signals simultaneously, encoded as different frequencies / analog signals.

This approach makes the single DSL link appear as if there were three separate links, so that a telephone call and an Internet connection can share the DSL link at the same time. (We'll describe this technique of frequency-division multiplexing in

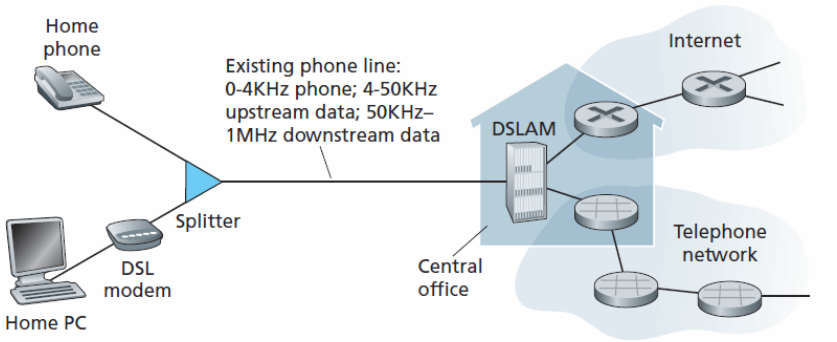


Figure 1.5 ♦ DSL Internet access

- DSL standards define transmission rates of 12 Mbps downstream and 1.8 Mbps upstream, and 24 Mbps downstream and 2.5 Mbps downstream
- Because the up/down streams are different, access = **asymmetric**
- Actual rates can be less than these standards, as DSL providers can purposefully limit households based on their DSL packages, **or** due to the distance between the CO and household **or** due to electrical interference **or** gauge of twisted-pair line.
- Usually can only use DSL if household between 5 and 10km of CO

Cable (aka HFC - Hybrid Fibre Coax)

- Use television companies' cable television infrastructure to provide internet access.
- Household obtains internet access from same company that provides its cable television.
- Fibre nodes connect neighbourhoods to the cable companies, and - within neighbourhoods - households connect to the fibre node via coaxial cable
- Also known as **Hybrid Fibre Coax (HFC)** since both fibre and coaxial cable are employed in this system.

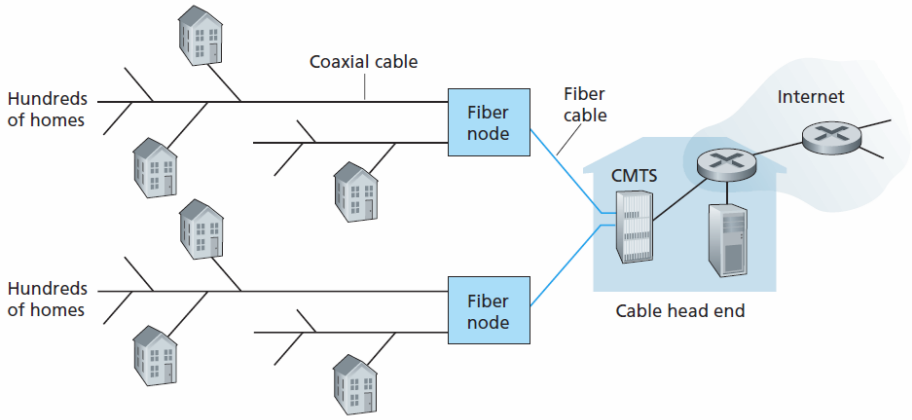


Figure 1.6 ♦ A hybrid fiber-coaxial access network

- Cable internet requires a **cable modem**

- Similar to DSL modem - external device, connects to PC via ethernet port
- **Cable Modem Termination System (CMTS)** converts analog signals from household cable modems back to digital format - similar to what DSL's DSLAM does
- Access is asymmetric (higher downstream than upstream transmission)
    - But cable access = faster than DSL
- **NB:** Cable internet access is 'shared' by everyone in the neighbourhood
    - Every packet sent from/to the head end (the CMTS) passes through every household's link.
    - Thus, if multiple households are downloading videos, their download speed can be much slower than if just one household was downloading.
    - Distributed multiple access protocol used to avoid transmission collisions

Fibre to the home (FTTH)

- Optical fibre path from the CO (telco) to each household
- **Method 1:** Direct Fibre
    - Each home has a personal fibre link to the CO
- **Method 2 (more common):** shared/split fibre
    - Big fibre links from CO are shared by multiple homes. When the fibre gets close to the homes, it branches into personal links for each household.
    - 2 competing technologies/architectures to perform this splitting: **Active Optical Networks (AONs)** and **Passive Optical Networks (PONs)**
        - **AON** is essentially switched ethernet
        - **PON**

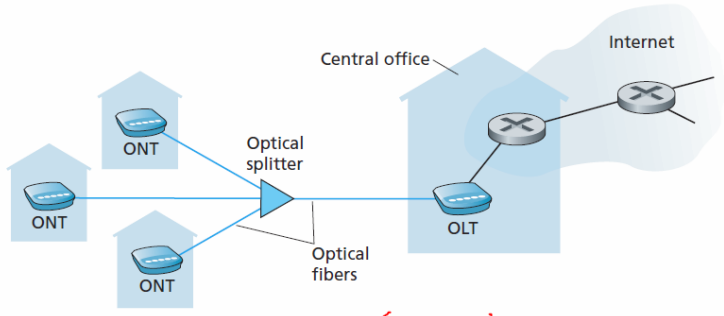


Figure 1.7 ♦ FTTH Internet access (PON)

works as follows:

Each home has an **Optical Network Terminator (ONT)** which connects to a splitter for a group of homes, and this splitter links to an **Optical Line Terminator (OLT)** in the telco's CO. The telco's OLT converts between optical and electrical signals and connects to the internet via a telco router. Each household has a router which is connected to its ONT.

Satellite link can also be used to connect households to internet when DSL, Cable and FTTH are not available.

- More common in rural areas. StarBand and HughesNet are two providers.

Dial-up access is based on same model as DSL

- But is excruciatingly slow
- Home modem connects to ISP modem via telephone line

Internet access in the Enterprise (and home): Ethernet and WiFi

- On corporate + uni campuses (and some homes), end systems connect to edge routers via LAN.
    - Ethernet = most prevalent LAN technology used in this context
    - uses twisted-pair copper wire to connect to switch
    -

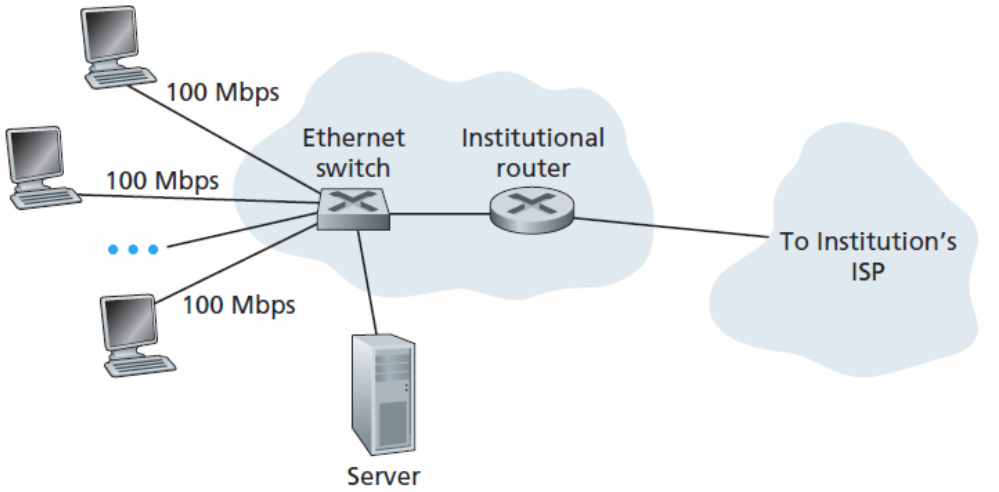


Figure 1.8 ♦ Ethernet Internet access

- Users also connect wirelessly to institutional router via access points linked (wired) to router
    - WiFi = based on IEEE 802.11 technology

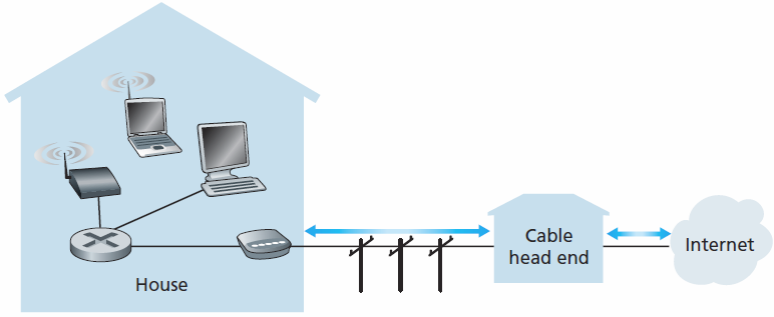


Figure 1.9 ♦ A typical home network

Wire-Area Wireless Access:

- 3G, 4G, LTE provide packet-switched wide-area wireless internet even if user is 10s of kms away from base station
- Radio signals rather than coaxial cable, copper wire etc

Physical Media

- Bits travelling through networks/internet travel via **physical media/mediums**
    - HFC (Cable): fibre cable + coaxial cable
    - DSL, Ethernet: copper wire
    - Mobile networks: radio
    - Two categories of physical media: **guided** and **unguided**
        - **Guided**: waves travel across solid medium (fibre, copper wire, coaxial cable)
        - **Unguided**: waves travel through atmosphere + outer space (wireless LAN, digital satellite channel)

- **Twisted-Pair Copper Wire**
    - Least expensive, most commonly used data transmission medium
    - Two insulated copper wires (1 mm thick) twisted together in spiral pattern to reduce electrical interference from other pairs nearby
    - Multiple pairs can be bundled together in a cable by wrapping each pair in a protective shield
    - 1 wire pair = 1 communication link
    - **Unshielded Twisted Pair (UTP)** typically used for LANs in buildings
    - Data rates depend on wire thickness and length
    - Dominant solution for high-speed LAN networking
    - Data rates up to 10 Gbps

- **Coaxial Cable**
    - Like twisted pair, uses 2 copper conductors
        - Concentric rather than parallel
    - Special braided insulation and shielding
    - Common in cable television systems
    - Can be used as a guided **shared medium**
        - ie: a number of end systems can connect directly to the cable, each receiving what any single connected host sends through the cable
- **Fibre Optics**
    - Conducts light pulses which represent bits
    - Supports insane bit rates
    - Immune to electromagnetic interference, hard to sniff/tap, barely lose speed over long distances
    - Traditionally used for overseas internet cable links / ‘backbone’ of internet
    - **Optical carrier (OC)** standard link speeds range from 51.8 Mbps to 39.8Gbps
        - *OC-n*
        - → $speed = n * 51.8$

- **Terrestrial Radio Channels**
  - Carry signals in electromagnetic spectrum
  - No physical wire required, Can penetrate walls, Mobile users can connect
  - Path loss and shadow fading decrease signal strength (distance, obstructing objects)
  - Multi-path fading: signal reflection off interfering objects
  - Interference (other transmissions, electromagnetic signals)
  - **3 categories:**
    - Very short distance (keyboard, bluetooth headset)
    - Local areas (LAN)
    - Wide area (3G, 4G)

- **Satellite Radio Channels**
  - Communication satellite link ground stations (transmitters/receivers) via microwaves
  - Satellite receives transmissions on one frequency, regenerates signal using repeater, and transmits signal on another frequency
  - **2 types of satellite used:**
    - **Geostationary satellites**
      - Remain in permanent spot above earth
      - ALLOWS INTERNET ACCESS
    - **Low-earth orbiting satellites (LEO)**
      - Closer to earth and rotate around earth
      - Communicate with each other
      - DOESN'T ALLOW INTERNET ACCESS (YET)

The Network Core...

Packet switching:

- In a network application, **messages** are exchanged to perform functions and to send data.
- To send a msg from a source end system to a destination end system, the source breaks long (data) messages into segments called **packets**. Each packet travels through communication links and packet switches (routers and link-layer switches).
- Packets are transmitted over each communication link at the full transmission rate of the link.
  - Thus, if a source end system or packet switch is sending a packet of *L bits* over a link with transmission rate *R* bits/sec, then the time taken to transmit the packet is *L/R* seconds

Store-and-Forward Transmission:

- Used by most packet switches
- Packet switch has to receive the entire packet before it can begin to transmit the first bit of the packet onto the outbound link (not streamlined at all)

- Thus, transmission delay if using *store-and-forward* is *2*L/R* since it takes *L/R* for both source and dest router to gather all the bits of the packet
    - If the switch instead forwarded bits as soon as they arrived, transmission delay would be *L/R*

Now let's calculate the amount of time that elapses from when the source begins to send the first packet until the destination has received all three packets. As before, at time L/R, the router begins to forward the first packet. But also at time L/R the source will begin to send the second packet, since it has just finished sending the entire first packet. Thus, at time $2L/R$, the destination has received the first packet and the router has received the second packet. Similarly, at time $3L/R$, the destination has received the first two packets and the router has received the third packet. Finally, at time $4L/R$ the destination has received all three packets!

Let's now consider the general case of sending one packet from source to destination over a path consisting of N links each of rate R (thus, there are N-1 routers between source and destination). Applying the same logic as above, we see that the end-to-end delay is:

$$d_{\text{end-to-end}} = N\frac{L}{R} \qquad (1.1)$$

- (L = bits, N = links, R = bits/sec transmission rate)
- The transmission delay will be N*(L/R) because it depends on how many packets make up the entire message

Queueing Delays and Packet Loss

- A packet switch has multiple links attached to it
- Each link has an **output buffer/queue** which stores packets that the router is about to send to that link.
- If an arriving packet needs to be transmitted onto a link which is busy with another packet, the arriving packet must wait in the output queue/buffer.
- So in addition to delays from store-and-forward design, packets also suffer **queueing delays** when they need to wait in the output buffer/queue (depends on level of congestion in network)
- If an arriving packet needs to go to a link which is busy, and the buffer is also full, then **packet loss** occurs.
    - Either the arriving packet or one of the queued packets will be dropped

Figure 1.12 illustrates a simple packet-switched network. As in Figure 1.11, packets are represented by three-dimensional slabs. The width of a slab represents the number of bits in the packet. In this figure, all packets have the same width and hence the same length. Suppose Hosts A and B are sending packets to Host E. Hosts A and B first send their packets along 10 Mbps Ethernet links to the first router. The router then directs these packets to the 1.5 Mbps link. If, during a short interval of time, the arrival rate of packets to the router (when converted to bits per second) exceeds 1.5 Mbps, congestion will occur at the router as packets queue in the link's output buffer before being transmitted onto the link. For example, if Host A and B each send a burst of five packets back-to-back at the same time, then most of these packets will spend some time waiting in the queue. The situation is, in fact, entirely analogous to many common-day situations—for example, when we wait in line for a bank teller or wait in front of a tollbooth. We'll examine this queuing delay in more detail in Section 1.4.

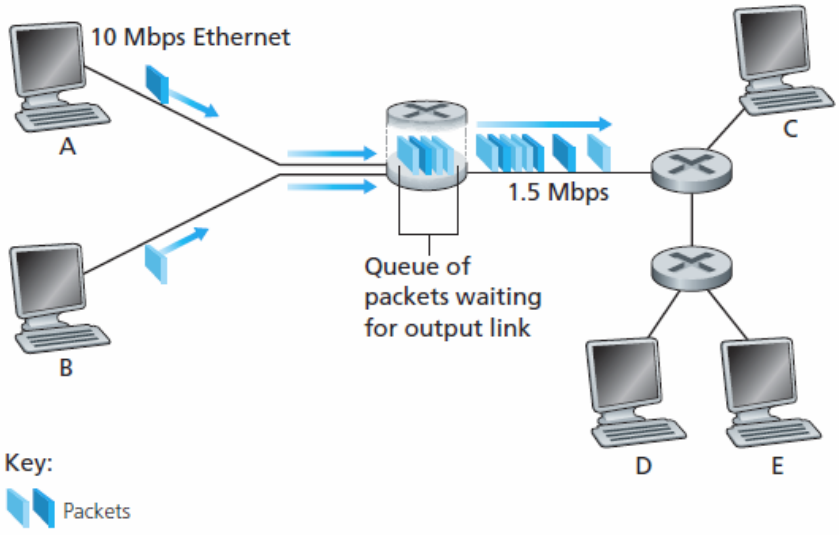


Figure 1.12 ♦ Packet switching

Forwarding Tables and Routing Protocols

- So we know that a router takes a packet arriving on one of its attached communication links and forwards that packet onto another of its attached communication links
  - = **Packet Forwarding**
- But how does the router choose which links to forward arriving packets onto?
  - In the case of the internet, each end system has an IP address
  - When a source end system wants to send data to a destination end system, the source includes the IP address of the destination in the packet headers of the segmented data.

- Each router uses a **forwarding table** that maps (portions of) IP/destination addresses to outbound links
- When a packet arrives at a router, the router examines the packet header corresponding to the destination IP, chooses the most appropriate link (based on which link maps to a portion of the destination address) and forwards the packet to the appropriate adjacent router.
- In summary: A router uses a packet's destination (IP) address to index a forwarding table and determine the appropriate outbound link.
- Forwarding tables are configured using **routing protocols**. A routing protocol might, for example, configure a table based on finding the shortest path to a packet's destination.

Circuit Switching

- In a circuit-switched network, the resources needed along each packet's transmission journey are reserved from the start. On the other hand, as we saw, packet-switched networks can suffer from queueing delays because resources such as links and buffers aren't reserved - they're just accessed on the fly.
- A circuit-switched network is like a restaurant which only takes reservations: There is the hassle of reserving the resources needed for the packet's journey before the packet can leave the source host, but once the reservations have been made, there should be no delays throughout the packet's journey.
- A packet-switched network is like a restaurant which doesn't take reservations: The source host doesn't have to hassle with phoning and making the reservation early, but it does run the risk of hitting a long queue when the packet arrives at the router/restaurant.
- Classic example of circuit-switched network = traditional telephone networks
    - Once the call has been established (resources reserved) the data/conversation is transferred at the *guaranteed* constant rate.

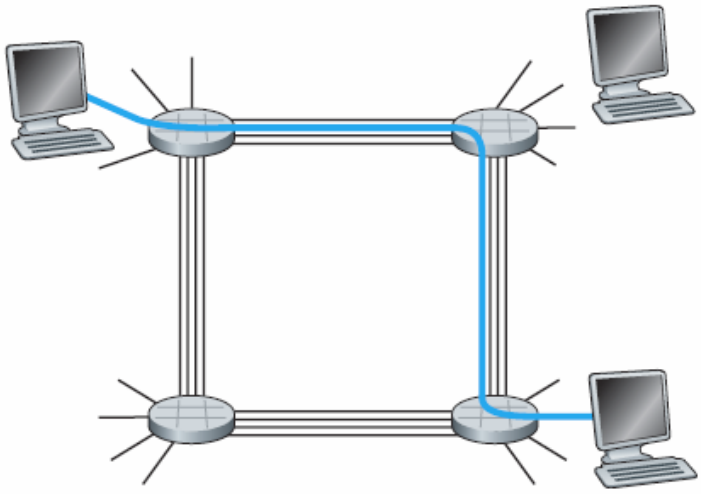


Figure 1.13 ♦ A simple circuit-switched network consisting of four switches and four links

Figure 1.13 illustrates a circuit-switched network. In this network, the four circuit switches are interconnected by four links. Each of these links has four circuits, so that each link can support four simultaneous connections. The hosts (for example, PCs and workstations) are each directly connected to one of the switches. When two hosts want to communicate, the network establishes a dedicated **end-to-end connection** between the two hosts. Thus, in order for Host A to communicate with Host B, the network must first reserve one circuit on each of two links. In this example, the dedicated end-to-end connection uses the second circuit in the first link and the fourth circuit in the second link. Because each link has four circuits, for each link used by the end-to-end connection, the connection gets one fourth of the link's total transmission capacity for the duration of the connection. Thus, for example, if each link between adjacent switches has a transmission rate of 1 Mbps, then each end-to-end circuit-switch connection gets 250 kbps of dedicated transmission rate.

In contrast, consider what happens when one host wants to send a packet to another host over a packet-switched network, such as the Internet. As with circuit switching, the packet is transmitted over a series of communication links. But different from circuit switching, the packet is sent into the network without reserving any link resources whatsoever. If one of the links is congested because other packets need to be transmitted over the link at the same time, then the packet will have to wait in a buffer at the sending side of the transmission link and suffer a delay. The Internet makes its best effort to deliver packets in a timely manner, but it does not make any guarantees.

Multiplexing in circuit-switched networks

- A circuit in a link uses either **frequency-division multiplexing (FDM)** or **time-division multiplexing (TDM).**
  - FDM divides the frequency spectrum of a link based on the established connections to that link. Each connection's allocated frequency band is dedicated to that connection for its lifespan.
    - FM radio stations use FDM to share the frequency spectrum between 88 MHz and 108 MHz, where each station corresponds to a frequency band in that range
    - In telephone networks, the width of the frequency band is typically 4 kHz - hence the name **bandwidth.**
  - TDM divides time into frames of fixed duration, and each frame is further divided into fixed timeslots. When the network establishes a connection across a link, it dedicates one timeslot in every frame to this connection.

Figure 1.14 illustrates FDM and TDM for a specific network link supporting up to four circuits. For FDM, the frequency domain is segmented into four bands, each of bandwidth 4 kHz. For TDM, the time domain is segmented into frames, with four time slots in each frame; each circuit is assigned the same dedicated slot in the revolving TDM frames. For TDM, the transmission rate of a circuit is equal to the frame rate multiplied by the number of bits in a slot. For example, if the link transmits 8,000 frames per second and each slot consists of 8 bits, then the transmission rate of a circuit is 64 kbps.

Proponents of packet switching have always argued that circuit switching is wasteful because the dedicated circuits are idle during **silent periods**. For example,

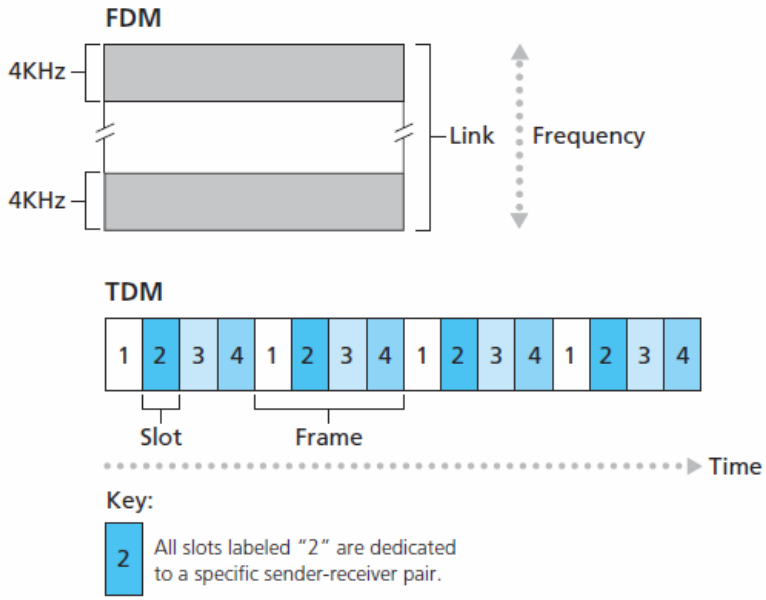


Figure 1.14 ♦ With FDM, each circuit continuously gets a fraction of the bandwidth. With TDM, each circuit gets all of the bandwidth periodically during brief intervals of time (that is, during slots)

- It can be argued that circuit-switching is wasteful because resources are left idle during **silent periods**. Eg:

  when one person in a telephone call stops talking, the idle network resources (frequency bands or time slots in the links along the connection's route) cannot be used by other ongoing connections. As another example of how these resources can be underutilized, consider a radiologist who uses a circuit-switched network to remotely access a series of x-rays. The radiologist sets up a connection, requests an image, contemplates the image, and then requests a new image. Network resources are allocated to the connection but are not used (i.e., are wasted) during the radiologist's contemplation periods. Proponents of packet switching also enjoy pointing out that establishing end-to-end circuits and reserving end-to-end transmission capacity is complicated and requires complex signaling software to coordinate the operation of the switches along the end-to-end path.

- Before we finish our discussion of circuit switching, let's work through a numerical example that should shed further insight on the topic. Let us consider how long it takes to send a file of 640,000 bits from Host A to Host B over a circuit-switched network. Suppose that all links in the network use TDM with 24 slots and have a bit rate of 1.536 Mbps. Also suppose that it takes 500 msec to establish an end-to-end circuit before Host A can begin to transmit the file. How long does it take to send the file? Each circuit has a transmission rate of (1.536 Mbps)/24 = 64 kbps, so it takes (640,000 bits)/(64 kbps) = 10 seconds to transmit the file. To this 10 seconds we add the circuit establishment time, giving 10.5 seconds to send the file. Note that the transmission time is independent of the number of links: The transmission time would be 10 seconds if the end-to-end circuit passed through one link or a hundred links. (The actual end-to-end delay also includes a propagation delay; see Section 1.4.)

Packet switching vs Circuit switching

- **Cons of packet switching:**
  - Not suitable for real-time services (ie: voice/video call) due to variable+unpredictable (queueing) delays
- **Pros of packet switching:**
  - Better sharing of transmission capacity
  - Simpler, more efficient, less costly to implement
  - **Why packet switching is better: example 1 (below)**
- Why is packet switching more efficient? Let's look at a simple example. Suppose users share a 1 Mbps link. Also suppose that each user alternates between periods of activity, when a user generates data at a constant rate of 100 kbps, and periods of inactivity, when a user generates no data. Suppose further that a user is active only 10 percent of the time (and is idly drinking coffee during the remaining 90 percent of the time). With circuit switching, 100 kbps must be *reserved* for *each* user at all times. For example, with circuit-switched TDM, if a one-second frame is divided into 10 time slots of 100 ms each, then each user would be allocated one time slot per frame.

  Thus, the circuit-switched link can support only 10 (= 1 Mbps/100 kbps) simultaneous users. With packet switching, the probability that a specific user is active is 0.1 (that is, 10 percent). If there are 35 users, the probability that there are 11 or more simultaneously active users is approximately 0.0004. (Homework Problem P8 outlines how this probability is obtained.) When there are 10 or fewer simultaneously active users (which happens with probability 0.9996), the aggregate arrival rate of data is less than or equal to 1 Mbps, the output rate of the link. Thus, when there are 10 or fewer active users, users' packets flow through the link essentially without delay, as is the case with circuit switching. When there are more than 10 simultaneously active users, then the aggregate arrival rate of packets exceeds the output capacity of the link, and the output queue will begin to grow. (It continues to grow until the aggregate input rate falls back below 1 Mbps, at which point the queue will begin to diminish in length.) Because the probability of having more than 10 simultaneously active users is minuscule in this example, packet switching provides essentially the same performance as circuit switching, *but does so while allowing for more than three times the number of users.*
  - **Why packet switching is better: example 2 (below)**

- Let's now consider a second simple example. Suppose there are 10 users and that one user suddenly generates one thousand 1,000-bit packets, while other users remain quiescent and do not generate packets. Under TDM circuit switching with 10 slots per frame and each slot consisting of 1,000 bits, the active user can only use its one time slot per frame to transmit data, while the remaining nine time slots in each frame remain idle. It will be 10 seconds before all of the active user's one million bits of data has been transmitted. In the case of packet switching, the active user can continuously send its packets at the full link rate of 1 Mbps, since there are no other users generating packets that need to be multiplexed with the active user's packets. In this case, all of the active user's data will be transmitted within 1 second.

Summary:

- **Circuit switching:** pre-allocates use of the transmission link regardless of demand, with wasted idle link time.
- **Packet switching:** allocates link use *on demand.* Capacities of the transmission links are shared on a packet-by-packet basis, only among users who are actively sending out packets.

- In today's telecommunication networks, trend is starting to lean towards packet switching. ie: telephone networks use packet switching for parts of expensive overseas phone calls.

Network of networks

- We now know how end users are connected to each other through their ISPs, but to complete the 'internet puzzle', how are ISPs connected to each other?
- ISPs connected via a 'network of networks'
- ISPs are not all directly connected to each other (mesh design) - this would be super expensive and impractical.
- No ISP has presence in each and every city in the world
    - 'Access ISPs' in each city/region connect to one of the the big 'regional ISPs' which connect to the **tier-1 ISPs**
    - **Tier-1 ISPs** (AT&T, Sprint, Level 3 Communications, a few more..) have widest presence on earth
- Customer-provider relationship at each level of the hierarchy: Regional ISPs pay Tier-1 ISPs….. Access ISPs pay Regional ISPs….. Households pay Access ISPs
- There can also be different levels of regional ISPs added to the hierarchy
- For a complete picture of the internet today, we must incorporate **Points of Presence (PoPs), multi-homing, peering,** and **Internet exchange points (IXPs)** to this hierarchy.
- **PoP:** router(s) in an ISP's network which allows customer ISPs to connect into the provider ISPs
    - Access ISPs do not have PoPs, since there are no customer ISPs below them on the hierarchy
- **Multi-home:** Any ISP (except tier-1s) can be customer ISPs to multiple provider ISPs (ie: an Access ISP can multi-home with 2 regional ISPs or it can multi-home with 1 or 2 regional ISPs and a tier-1 ISP)
    - A regional ISP can also multi-home with multiple tier-1 ISPs

    - Allows ISP to stay connected to internet even if one of its providers has a failure
    - (Of course not tier-1s since there are no ISPs above them)
- **Peering:** To reduce costs of paying provider ISPs, nearby customer ISPs on the same hierarchy level can connect their networks together so the two of them require only one direct connection to a higher-level ISP rather than each paying a lot for their own connection.
    - Tier-1 ISPs also peer with each other

Internet Exchange Point (IXP):

- Meeting point where multiple ISPs can peer together (usually a building full of switches...around 300 IXPs in the world/internet atm).

Content Provider Networks: Google is the biggest one. Connects to all different levels of the hierarchy, has huge data-centres

- The Google private network attempts to 'bypass' upper tiers of the internet by peering with lower-tier ISPs either directly or at IXPs.
- By having its own huge private network, a content provider can reduce payments to tier-1 ISPs and have firmer control of content delivery to its users.

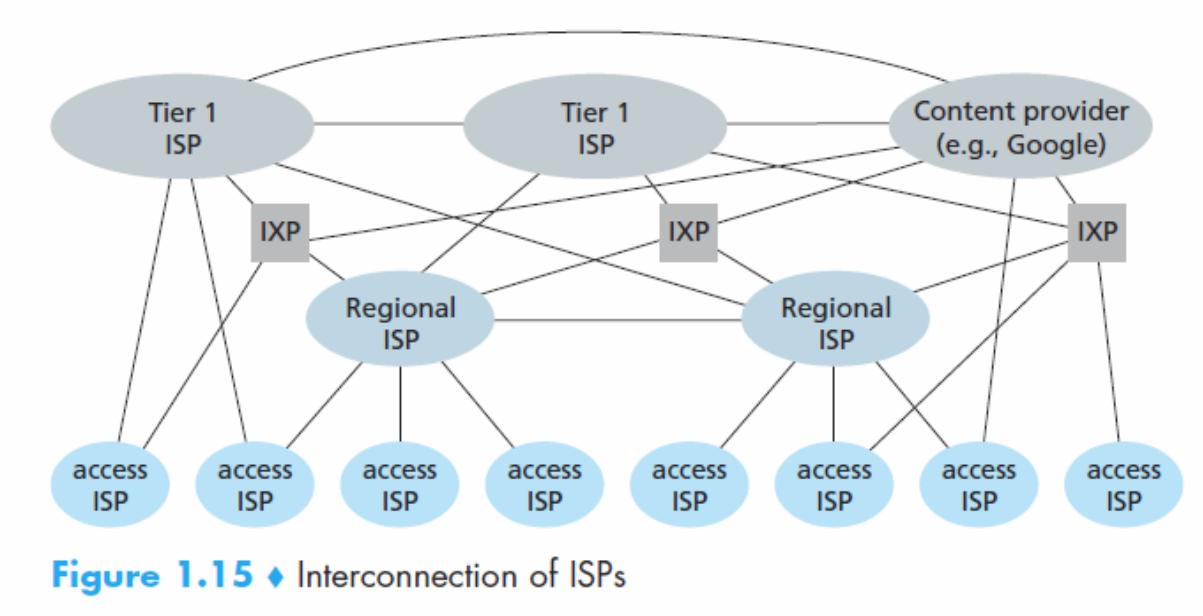


Figure 1.15 ♦ Interconnection of ISPs

Delay, Loss, and Throughput in Packet-Switched Networks

When packet travels from source host to destination host, suffers from delays:

- **Nodal processing delay**
    - Time required to do things like examine packet's header and determine where to direct packet
    - Delays of just microseconds - insignificant
- **Queueing delay**
    - Time taken for packet to finish waiting in the queue and be transmitted to the link
    - Can vary from packet to packet
        - If queue is empty, delay = 0

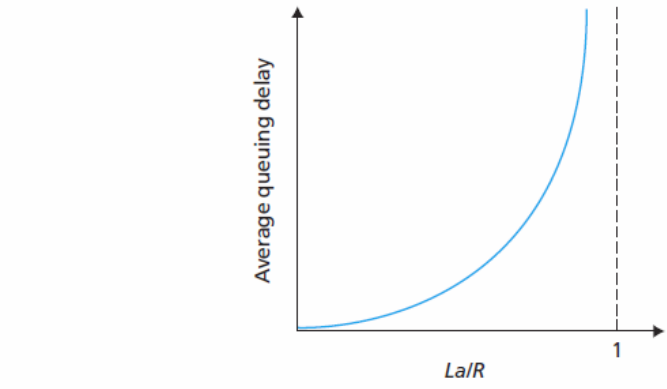
        - If 10 packets arrive at a queue, the first one will suffer no delay, but the last one will suffer a large delay as it waits for the 9 in front of it to be transmitted
        - Have to use statistical measures to estimate queueing delay for different cases - depends on expected behaviour / nature of network traffic - does traffic arrive in bursts or constant stream?
    - When is the queuing delay large and when is it insignificant? The answer to this question depends on the rate at which traffic arrives at the queue, the transmission rate of the link, and the nature of the arriving traffic, that is, whether the traffic arrives periodically or arrives in bursts. To gain some insight here, let a denote the average rate at which packets arrive at the queue (a is in units of packets/sec). Recall that R is the transmission rate; that is, it is the rate (in bits/sec) at which bits are pushed out of the queue. Also suppose, for simplicity, that all packets consist of L bits. Then the average rate at which bits arrive at the queue is La bits/sec. Finally, assume that the queue is very big, so that it can hold essentially an infinite number of bits. The ratio La/R, called the **traffic intensity**, often plays an important role in estimating the extent of the queuing delay. If $La/R > 1$, then the average rate at which bits arrive at the queue exceeds the rate at which the bits can be transmitted from the queue. In this unfortunate situation, the queue will tend to increase without bound and the queuing delay will approach infinity! Therefore, one of the golden rules in traffic engineering is: *Design your system so that the traffic intensity is no greater than 1.*
    - (Rate at which packets arrive at the queue / transmission rate) must be less than 1, otherwise packets are arriving at the queue faster than the link can transmit/forward packets at the front of the queue
    - **NB:** page 40
    - Can cause delay of microseconds or milliseconds, depending on level of traffic arriving at queue
    - 

      Figure 1.18 ♦ Dependence of average queuing delay on traffic intensity
        - As traffic intensity approaches 1, average queueing delay increases rapidly.
        - Small % increase in intensity = much larger % increase in delay
- **Transmission delay**
    - Time required for the router/link to push out all the bits of a packet.
        - AKA Time required to get all bits of a packet *into* link/router A
    - Packet switch has to receive the entire packet (all bits) before it can begin to transmit the first bit of the packet onto the outbound link.
    - Function of packet length and transmission delay
    - *Transmission Delay = L/R* *(L bits / R bits per sec)*
        - Number of bits / transmission rate
- **Propagation delay**

    - Time required to propagate a bit from router/link A to router B
    - Propagation speed depends on physical medium of the link (ie: copper wire, coaxial, fibre)
    - Function of distance between router/link A and B
    - *Propagation Delay* = *d/s* where *d* is the distance between routers A and B, and *s* is the propagation speed of the link
    - Delay of milliseconds
- In a packet-switched network, the first bits in a packet can arrive at router B while many of the packet's remaining bits are still waiting to be transmitted into router A.

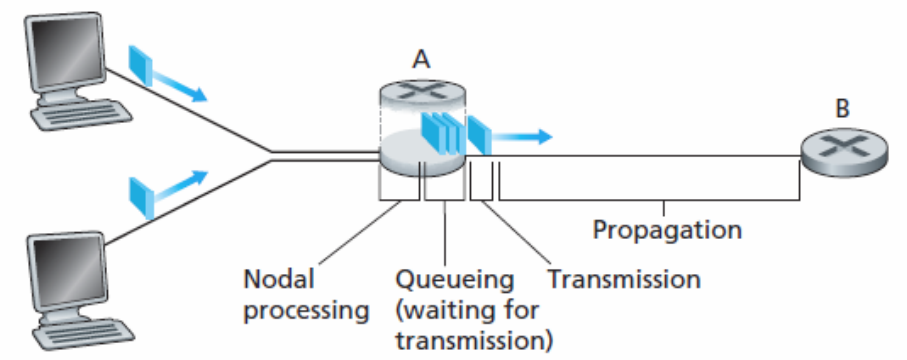


Figure 1.16 ♦ The nodal delay at router A

- Together, these delays make up **total nodal delay**
    - If we let d_{proc}, d_{queue}, d_{trans}, and d_{prop} denote the processing, queuing, transmission, and propagation delays, then the total nodal delay is given by

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

    - The contribution of these delay components can vary significantly. For example, d_{prop} can be negligible (for example, a couple of microseconds) for a link connecting two routers on the same university campus; however, d_{prop} is hundreds of milliseconds for two routers interconnected by a geostationary satellite link, and can be the dominant term in d_{nodal}. Similarly, d_{trans} can range from negligible to significant. Its contribution is typically negligible for transmission rates of 10 Mbps and higher (for example, for LANs); however, it can be hundreds of milliseconds for large Internet packets sent over low-speed dial-up modem links. The processing delay, d_{proc}, is often negligible; however, it strongly influences a router's maximum throughput, which is the maximum rate at which a router can forward packets.

Packet loss:

- Since the queues preceding links/routers have finite capacity, in reality, packet delays do not approach infinity as traffic intensity approaches 1.
- Rather, if a packet tries to enter a full queue, it will be dropped (and lost)
- From a host's POV, packet loss looks like a packet being transmitted to the network core but never emerging at the destination
- As traffic intensity increases, fraction of lost packets increases
- Lost packets can be retransmitted over and over to ensure all data eventually reach the destination

We have an equation for total nodal delay at router A, but what about the total delay from router A to router B? (below) (end-to-end delay)

1.4.3 End-to-End Delay

Our discussion up to this point has focused on the nodal delay, that is, the delay at a single router. Let's now consider the total delay from source to destination. To get a handle on this concept, suppose there are $N - 1$ routers between the source host and the destination host. Let's also suppose for the moment that the network is uncongested (so that queuing delays are negligible), the processing delay at each router and at the source host is d_{proc}, the transmission rate out of each router and out of the source host is R bits/sec, and the propagation on each link is d_{prop}. The nodal delays accumulate and give an end-to-end delay,

$$d_{end\text{-}end} = N\,(d_{proc} + d_{trans} + d_{prop}) \qquad (1.2)$$

where, once again, $d_{trans} = L/R$, where L is the packet size. Note that Equation 1.2 is a generalization of Equation 1.1, which did not take into account processing and propagation delays. We leave it to you to generalize Equation 1.2 to the case of heterogeneous delays at the nodes and to the presence of an average queuing delay at each node.

Traceroute

- Simple program that runs in any internet host → discover network paths and measure delay
- User specifies a hostname → traceroute sends multiple special packets towards that destination. As the special packets pass through routers on the transmission journey, each router sends back to the source host a short message containing its name and address
- When the packet reaches the final router on its transmission journey, the destination router sends back a message to the source host, which records how long it took to receive the message since sending out the special packet.
- It also records the name and address of the destination router/host.
- Thus, the source host can reconstruct routes taken by packets it sends out, and can determine delays at each router along the way.
- Traceroute repeats this 'special packet' experiment to each router along the way 3 times (so it sends 3*N packets to the destination)

-

Here is an example of the output of the Traceroute program, where the route was being traced from the source host gaia.cs.umass.edu (at the University of Massachusetts) to the host cis.poly.edu (at Polytechnic University in Brooklyn). The output has six columns: the first column is the *n* value described above, that is, the number of the router along the route; the second column is the name of the router; the third column is the address of the router (of the form xxx.xxx.xxx.xxx); the last three columns are the round-trip delays for three experiments. If the source receives fewer than three messages from any given router (due to packet loss in the network), Traceroute places an asterisk just after the router number and reports fewer than three round-trip times for that router.

```
1 cs-gw (128.119.240.254) 1.009 ms 0.899 ms 0.993 ms
2 128.119.3.154 (128.119.3.154) 0.931 ms 0.441 ms 0.651 ms
3 border4-rt-gi-1-3.gw.umass.edu (128.119.2.194) 1.032 ms 0.484 ms 0.451 ms
4 acr1-ge-2-1-0.Boston.cw.net (208.172.51.129) 10.006 ms 8.150 ms 8.460 ms
5 agr4-loopback.NewYork.cw.net (206.24.194.104) 12.272 ms 14.344 ms 13.267 ms
6 acr2-loopback.NewYork.cw.net (206.24.194.62) 13.225 ms 12.292 ms 12.148 ms
7 pos10-2.core2.NewYork1.Level3.net (209.244.160.133) 12.218 ms 11.823 ms 11.793 ms
8 gige9-1-52.hsipaccess1.NewYork1.Level3.net (64.159.17.39) 13.081 ms 11.556 ms 13.297 ms
9 p0-0.polyu.bbnplanet.net (4.25.109.122) 12.716 ms 13.052 ms 12.786 ms
10 cis.poly.edu (128.238.32.126) 14.080 ms 13.035 ms 12.802 ms
```

In the trace above there are nine routers between the source and the destination. Most of these routers have a name, and all of them have addresses. For example, the name of Router 3 is `border4-rt-gi-1-3.gw.umass.edu` and its address is `128.119.2.194`. Looking at the data provided for this same router, we see that in the first of the three trials the round-trip delay between the source and the router was 1.03 msec. The round-trip delays for the subsequent two trials were 0.48 and 0.45 msec. These round-trip delays include all of the delays just discussed, including transmission delays, propagation delays, router processing delays, and queuing delays. Because the queuing delay is varying with time, the round-trip delay of packet *n* sent to a router *n* can sometimes be longer than the round-trip delay of packet *n*+1 sent to router *n*+1. Indeed, we observe this phenomenon in the above example: the delays to Router 6 are larger than the delays to Router 7!

End System, application and other delays:

- There are other delays in addition to processing, transmission, propagation delays
- eg: some protocols purposefully delay transmission (ie end system transmitting packet into shared medium like WiFi dongle)
- eg: Media Packetization Delay → present in VoIP apps
    - = time taken to fill a packet with digitized speech before it can be transmitted

Throughput in computer networks:

- **Instantaneous throughput =** rate at which host B is receiving the file
- **Average throughput =** (bits in file) / (seconds taken to receive all bits)
- In apps like VoIP, desirable to have low delay and constant instantaneous throughput above some threshold (because you obviously want to hear what the other person is saying without any breaks in conversation)
- In other apps, delay doesn't really matter but it's desirable to have highest possible throughput → eg: in file transfer, we can deal with a few

microseconds/seconds lost due to delay, but it's still super important that throughput is high as we need to transfer the whole file

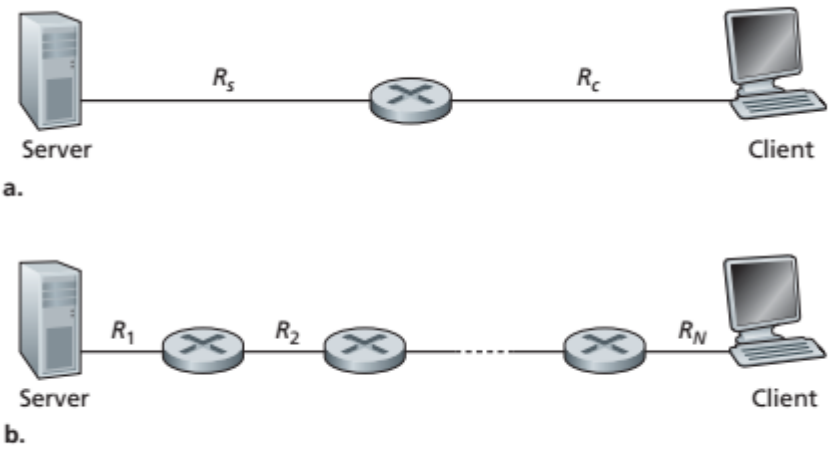


Figure 1.19 ♦ Throughput for a file transfer from server to client

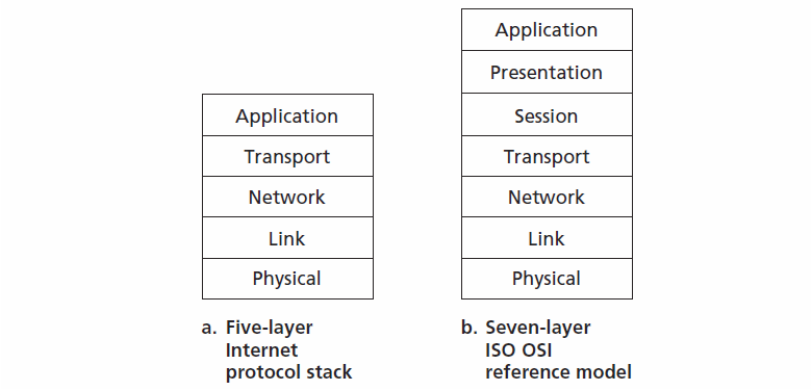
-
- In 1.19(a) above, the Server is sending a file to the Client directly through the router. Rs is the transmission rate from Server to router and Rc is transmission rate from router to Client.
    - If Rc > Rs, there will never be a bottleneck at the router/link since packets will be leaving the router faster than they arrive
    - But if Rc < Rs then there will be a bottleneck at the router as packets are being sent really fast to the router, but the router can't send them out to the client fast enough to have a clear queue.
        - In this case the router is a **bottleneck link**
            - And the throughput of this network is the **transmission rate of the bottleneck link along the server-to-client path = min{Rc,Rs}** (the speed of the link with the slowest speed)
- In 1.19(b), throughput = min{R1, R2,…,Rn)
- Constraining factor for throughput in today's Internet is typically the Access Network
- Throughput largely depends on the transmission rates of server-to-client link
    - If present, traffic also has a strong effect on throughput
- When there is no traffic, throughput can be approximated as the transmission rate of the slowest link (the minimum transmission rate along the path)

Protocol Layers and their Service Models

Layered Architecture

- Protocols implemented in **layers**
    - Remember aeroplane analogy (horizontal to illustrate layers)
- Each protocol belongs to one of the layers
- **Service model:** each layer offers services to the layer above
- Network layer often mixture of hardware & software
- Pros of protocol layering:
    - Structured way to discuss components

    - Modularity = easy to update components
- Cons of protocol layering:
    - Layers may duplicate lower-layer functionality
    - Layers may be dependent on other layers (ie: might need a timestamp) - which defeats the point of layering



- **Figure 1.23** ♦ The Internet protocol stack (a) and OSI reference model (b)
- **Protocol Stack:** the protocols of the various layers
- Internet protocol stack = five layers (see figure above)
    - Physical, link, network, transport, application
        - ATNLP:
            - Application
            - Transport
            - Network
            - Link
            - Physical

Application Layer:

- Where network apps and their app-layer protocols reside
- eg: HTTP, SMTP, FTP
- DNS (Domain Name System) = Application Layer protocol which translates URIs to 32 bit-addresses
- Application-layer protocols = distributed over multiple end systems → application in one end system using the protocol to exchange packets of information with the application in another end system (abiding by that same protocol)

Transport Layer:

- Transports application-layer messages between application endpoints
- Two internet transport layer protocols: **TCP** and **UDP**
    - Both transport application-layer messages
    - **TCP:**
        - connection-oriented service to its applications
        - Guaranteed message delivery
        - flow control (sender/receiver speed matching)
        - Breaks messages into short segments

        - Congestion-control mechanism
    - **UDP:**
        - connectionless service to its applications
        - No reliability, no flow control, no congestion control
- **Message =** packet of information at application layer
- **Segment =** transport-layer packet

Network (IP) Layer:

- = Transport Layer's delivery man
- Moves **datagrams** from host to host via a series of routers
    - **Datagram =** network-layer packet
- Transport Layer passes message+address to Network Layer, and Network Layer delivers the message to the Transport Layer in the host at that address
- Network Layer protocols: **IP Protocol**
    - IP Protocol defines datagram fields + how end-systems/routers should act on the fields
    - All internet components with a network layer must run the IP protocol
- Network Layer also has routing protocols that determine which routes datagrams take to reach destinations
- AKA IP layer since IP Protocol so important

Link Layer

- The Network Layer uses the Link Layer to move datagrams/packets from router to router
    - At each node, network layer passes datagram down to link layer, which delivers it to next node/router, and then passes the datagram ('back') up to the network layer
- Link Layer protocols: **Ethernet, WiFi, DOCSIS, PPP**
- Different link-layer protocols used at different links along the datagram's route
    - eg: datagram handled by Ethernet on one link, WiFi on another
- **Frames =** link-layer packets

Physical Layer

- While Link Layer transports entire frames/packets at a time, the **Physical Layer moved individual bits within the frame/packet** between nodes/routers
- Different protocols used in this layer depending on transmission medium of the link (ie: copper wire, fibre, coaxial, radio, etc.)

OSI Model

- In 1970s, International Organisation for Standardisation (ISO) proposed that computer networks be organised around 7 layers
    - = **Open Systems Interconnection (OSI** model
- **Layers:**
    - Application
    - Presentation
    - Session

    - Transport
    - Network
    - Data Link
    - Physical
- 5 of the layers are similar to those in Internet Protocol Stack
    - 2 aren't:
        - Presentation Layer
            - Allow communicating apps to interpret exchanged data
        - Session Layer
            - Delimiting + synchronisation of data exchange

Encapsulation

- Packet switches (routers + link layer switches) do not implement all layers in the Internet Protocol Stack
    - Usually only the bottom layers
-

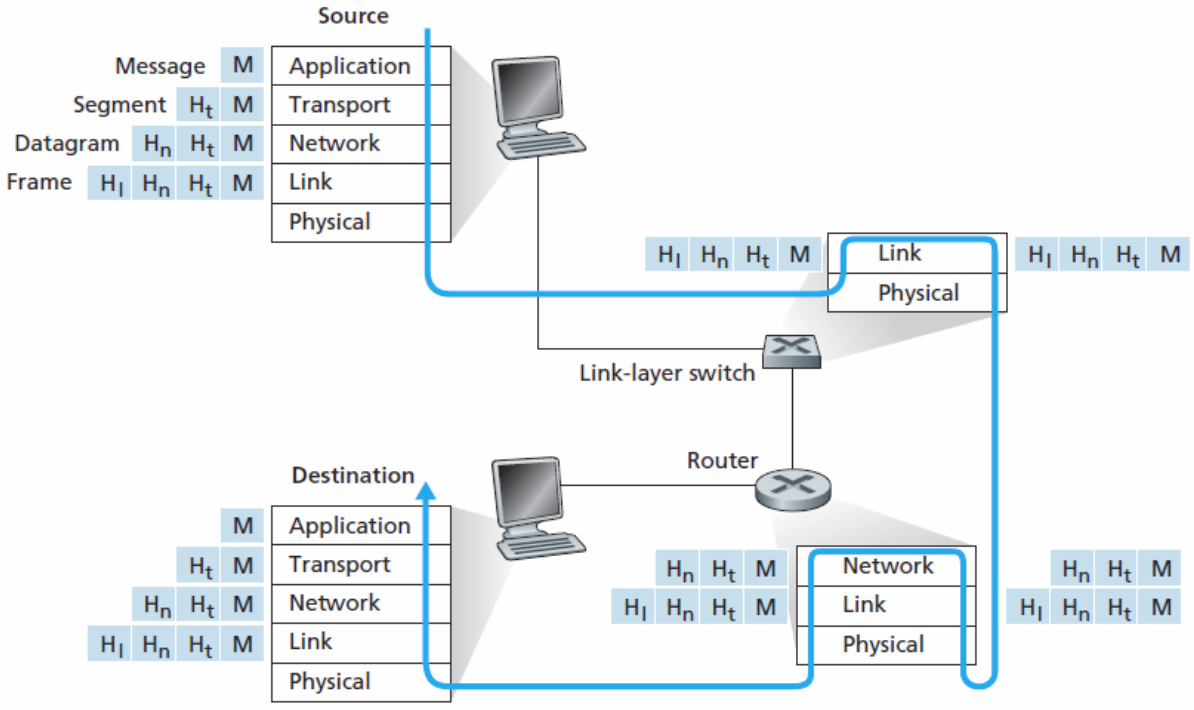


Figure 1.24 ♦ Hosts, routers, and link-layer switches; each contains a different set of layers, reflecting their differences in functionality

- Hosts implement all 5 layers (complexity at edge of internet)
- Each layer appends its own little piece of Header information to packets/datagrams passing through
- **At each layer, a packet has 2 types of fields: Header Fields and Payload Fields**

- Payload = packet from layer above

Networks under Attack

- **Malware:** malicious software/content
    - Often self-replicating → after infecting 1 host, tries to infect more hosts
- **Botnet:** controlled network of compromised devices
- **Virus:** malware that requires user interaction to infect
    - Classic example is dodgy email attachment
- **Worm:** malware that doesn't require user interaction to infect
- **Denial of Services (DoS) attacks:**
    - *Vulnerability attack:*
        - Few special messages/packets that trigger a network crash by exploiting some vulnerability on the host's end
    - *Bandwidth flooding:*
        - Torrent of packets that clog a target's access link (queue) so no legitimate packets can reach it
    - *Connection flooding:*
        - Attacker opens a whole lot of half/fully open (bogus) TCP connections at the target host. The host stops accepting new connections as it attempts to understand all the bogus ones.
- **DDoS**: multiple computers all performing a DoS attack (distributed)
    - Harder to detect than a single DoS attack

Packet sniffing:

- Attacker can intercept packets travelling through network

IP Spoofing:

- Inject packets into internet with a fake/modified source address
- Protect from using *end-to-end authentication* → ensure message originates from where we think it does

History of the internet: textbook pg 60 (section 1.7)

- Three key internet protocols: TCP, IP, UDP
- 4 key components of the web: HTML, HTTP, Web server, browser

Summary and Exercises: Page 66