UE19EC301 – Computer Communication Networks – Lecture Notes – Unit 1

1 Introduction to internet

- Today's Internet is arguably the largest engineered system ever created by mankind,with hundreds of mil-lions of connected computers,communication links,and switches;with billions of users who connect via lap-tops,tablets,and smartphones; and with an array of new Internet-connected devices such as sensors,Web cams, game consoles,picture frames,and even washing machines.
- Given that the Internet is so large and has so many diverse components and uses,is there any hope of under-standing how it works?
- Are there guiding principles and structure that can provide a foundation for understanding such an amazingly large and complex system?
- And if so,is it possible that it actually could be both interesting and fun to learn about computer networks?
- Fortunately, the answers to all of these questions is a resounding YES!
- Indeed,it's our aim in this course to provide you with a modern introduction to the dynamic field of com- puter networking,giving you the principles and practical insights you'll need to understand not only today's networks,but tomorrow's as well.
- This first unit presents a broad overview of computer networking and the Internet.
- Our goal here is to paint a broad picture and set the context for the rest of this course,to see the forest throughthe trees.
- We'll cover a lot of ground in this introductory unit and discuss a lot of the pieces of a computer network,withoutlosing sight of the big picture.
- We'll structure our overview of computer networks in this unit as follows.
- After introducing some basic terminology and concepts,we'll first examine the basic hardware and software components that make up a network.
- We'll begin at the network's edge and look at the end systems and network applications running in the network.
- We'll then explore the core of a computer network,examining the links and the switches that transport data,as well as the access networks and physical media that connect end systems to the network core.
- We'll learn that the Internet is a network of networks,and we'll learn how these networks connect with each other.
- After having completed this overview of the edge and core of a computer network,we'll take the broader and more abstract view in the second half of this unit.
- We'll examine delay,loss,and throughput of data in a computer network and provide simple quantitative models for end-to-end throughput and delay: models that take into account transmission,propagation,and queuing delays.
- We'll then introduce some of the key architectural principles in computer networking,namely,protocol layering and service models.
- We'll also learn that computer networks are vulnerable to many different types of attacks;we'll survey some of these attacks and consider how computer networks can be made more secure.

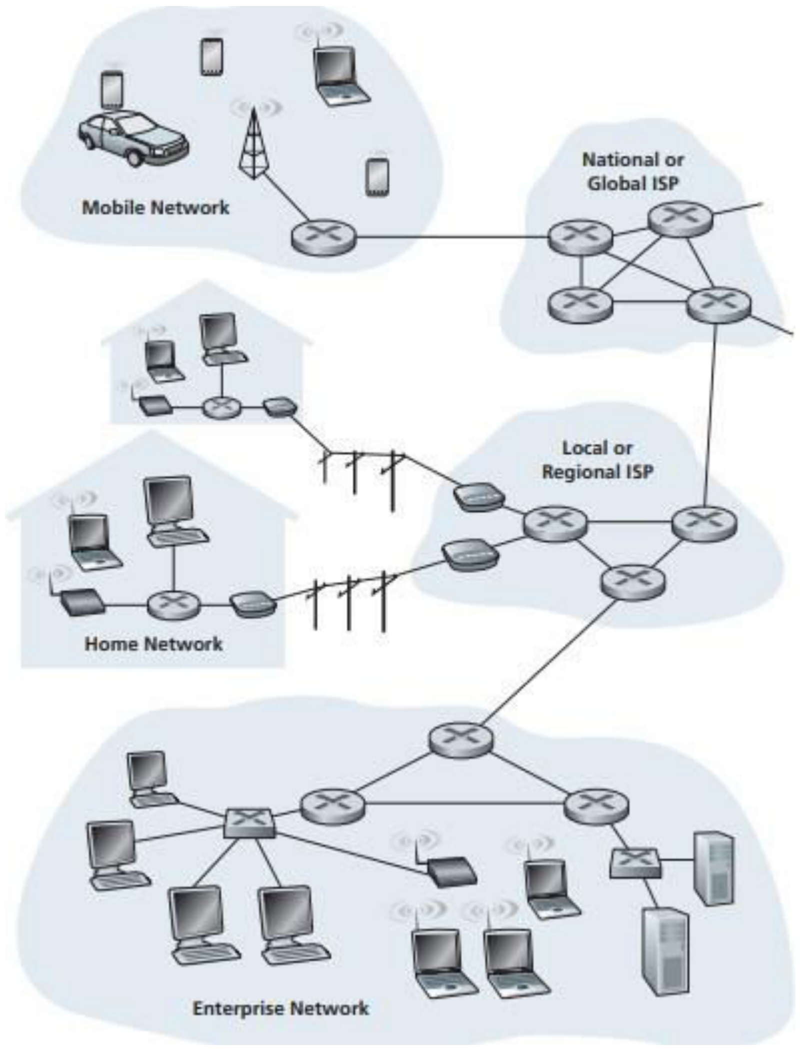


Figure 1: Some pieces of the Internet

1.1 What is the Internet?

- In this course,we'll use the public Internet,a specific computer network,as our principal vehicle for discussing computer networks and their protocols.
- But what is the Internet? There are a couple of ways to answer this question.
- First,we can describe the nuts and bolts of the Internet,that is,the basic hardware and software componentsthat make up the Internet.
- Second,we can describe the Internet in terms of a networking infrastructure that provides services to distributedapplications.
- Let's begin with the nuts-and-bolts description,using Figures 1 & 2 to illustrate our discussion.

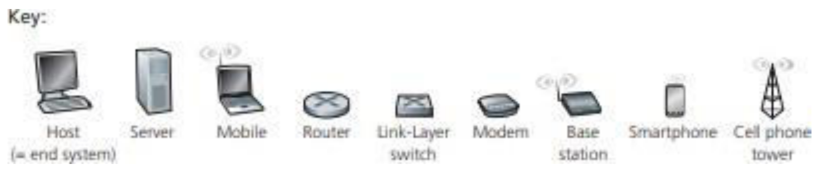


Figure 2: Some pieces of the Internet

1.1.1 A Nuts-and-Bolts Description

- The Internet is a computer network that interconnects hundreds of millions of computing devices throughout the world.
- Not too long ago,these computing devices were primarily traditional desktop PCs,Linux workstations,and so-called servers that store and transmit information such as Web pages and e-mail messages.
- Increasingly,however,nontraditional Internet end systems such as laptops, smartphones,tablets,Tvs,gaming consoles,Webcams,automobiles,environmental sensing devices,picture frames,and home electrical and security systems are being connected to the Internet.
- Indeed,the term computer network is beginning to sound a bit dated,given the many nontraditional devices thatare being hooked up to the Internet.
- In Internet jargon,all of these devices are called hosts or end systems.
- As of July 2011,there were nearly 850 million end systems attached to the Internet,not counting smart phones,laptops,and other devices that are only intermittently connected to the Internet.
- Overall,more there are an estimated 2 billion Internet users.
- End systems are connected together by a network of communication links and packet switches.
- There are many types of communication links,which are made up of different types of physical media,including coaxial cable,copper wire,optical fiber,and radio spectrum.
- Different links can transmit data at different rates,with the transmission rate of a link measured in bits/second.
- When one end system has data to send to another end system,the sending end system segments the data andadds header bytes to each segment.
- The resulting packages of information,known as packets in the jargon of computer networks,are then sent throughthe network to the destination end system, where they are reassembled into the original data.
- A packet switch takes a packet arriving on one of its incoming communication links and forwards that packeton one of its outgoing communication links.
- Packet switches come in many shapes and flavors,but the two most prominent types in today's Internet are routers and link-layer switches.
- Both types of switches forward packets toward their ultimate destinations.
- Link-layer switches are typically used in access networks,while routers are typically used in the network core.
- The sequence of communication links and packet switches traversed by a packet from the sending end systemto the receiving end system is known as a route or path through the network.
- The exact amount of traffic being carried in the Internet is difficult to estimate but Cisco estimates global Internet traffic will be nearly 40 exabytes per month in 2012.
- Packet-switched networks (which transport packets) are in many ways similar to transportation networks of highways,roads,and intersections (which transport vehicles).
- Consider,for example,a factory that needs to move a large amount of cargo to some destination warehouse located thousands of kilometers away.
- At the factory,the cargo is segmented and loaded into a fleet of trucks.
- Each of the trucks then independently travels through the network of highways, roads,and intersections to the destination warehouse.

- At the destination warehouse,the cargo is unloaded and grouped with the rest of the cargo arriving from thesame shipment.
- Thus,in many ways,packets are analogous to trucks,communication links are analogous to highways and roads,packetswitches are analogous to intersections,and end systems are analogous to buildings.
- Just as a truck takes a path through the transportation network,a packet takes a path through a computernetwork.
- End systems access the Internet through Internet Service Providers (ISPs), including residential ISPs such as local cable or telephone companies;corporate ISPs;university ISPs;and ISPs that provide WiFi access in airports,hotels,coffee shops,and other public places.
- 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,including residential broadband access suchas cable modem or DSL,high-speed local area network access,wireless access,and 56 kbps dial-up modem access.
- ISPs also provide Internet access to content providers,connecting Web sites directly to the Internet.
- The Internet is all about connecting end systems to each other,so the ISPs that provide access to end systems must also be interconnected.
- These lower-tier ISPs are interconnected through national and international upper-tier ISPs such as Level 3 Communications,AT&T,Sprint,and NTT.
- An upper-tier ISP consists of high-speed routers interconnected with high-speed fiber-optic links.
- Each ISP network,whether upper-tier or lower-tier,is managed independently, runs the IP protocol,and conformsto certain naming and address conventions.
- End systems,packet switches,and other pieces of the Internet run protocols that control the sending and receivingof information within the Internet.
- The Transmission Control Protocol (TCP) and the Internet Protocol (IP) are two of the most important protocols in the Internet.
- The IP protocol specifies the format of the packets that are sent and received among routers and end systems.
- The Internet’s principal protocols are collectively known as TCP/IP.
- Given the importance of protocols to the Internet,it’s important that everyone agree on what each and every protocol does,so that people can create systems and products that interoperate.
- This is where standards come into play.
- Internet standards are developed by the Internet Engineering Task Force (IETF).
- The IETF standards documents are called requests for comments (RFCs).
- RFCs started out as general requests for comments (hence the name) to resolve network and protocol design problems that faced the precursor to the Internet.
- RFCs tend to be quite technical and detailed.
- They define protocols such as TCP,IP,HTTP(for the Web),and SMTP (for e-mail).
- There are currently more than 6,000 RFCs.
- Other bodies also specify standards for network components,most notably for network links.
- The IEEE 802 LAN/MAN Standards Committee,for example,specifies the Ethernet and wireless WiFi standards.

1.1.2 A Services Description

- Our discussion above has identified many of the pieces that make up the Internet.
- But we can also describe the Internet from an entirely different angle—namely, as an infrastructure that providesservices to applications.
- These applications include electronic mail,Web surfing,social networks, instant messaging,Voiceover-IP (VoIP),videostreaming,distributed games,peer-to-peer (P2P) file sharing,television over the Internet,remote login,and much,much more.
- The applications are said to be distributed applications,since they involve multiple end systems that exchange data with each other.
- Importantly,Internet applications run on end systems—they do not run in the packet switches in the network core.
- Although packet switches facilitate the exchange of data among end systems, they are not concerned with the application that is the source or sink of data.
- Let's explore a little more what we mean by an infrastructure that provides services to applications.
- To this end,suppose you have an exciting new idea for a distributed Internet application,one that may greatly benefit humanity or one that may simply make you rich and famous.
- How might you go about transforming this idea into an actual Internet application?
- Because applications run on end systems,you are going to need to write programs that run on the end systems.
- You might,for example,write your programs in Java,C,or Python.
- Now,because you are developing a distributed Internet application,the programs running on the different end systems will need to send data to each other.
- And here we get to a central issue — one that leads to the alternative way of describing the Internet as a platform for applications.
- How does one program running on one end system instruct the Internet to deliver data to another program running on another end system?
- End systems attached to the Internet provide an Application Programming Interface (API) that specifies how a program running on one end system asks the Internet infrastructure to deliver data to a specific destination program running on another end system.
- This Internet API is a set of rules that the sending program must follow so that the Internet can deliver the data to the destination program.
- For now,let's draw upon a simple analogy,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 thepostal 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.
- The postal service,of course,provides more than one service to its customers.

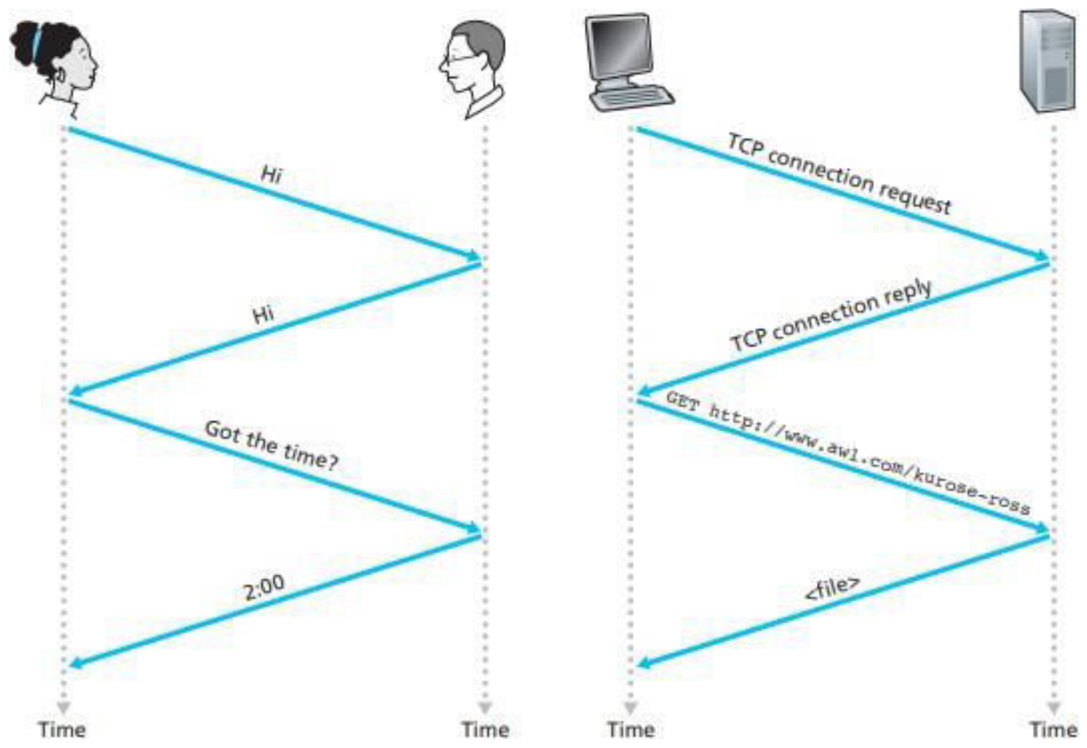


Figure 3: A human protocol and a computer network protocol

- ➢ It provides express delivery,reception confirmation,ordinary use,and many more services.
- ➢ In a similar manner,the Internet provides multiple services to its applications.
- ➢ We have just given two descriptions of the Internet;one in terms of its hardware and software components,theother in terms of an infrastructure for providing services to distributed applications.

But perhaps you are still confused as to what the Internet is.

What are packet switching and TCP/IP?

What are routers?

What kinds of communication links are present in the Internet?

What is a distributed application?

How can a toaster or a weather sensor be attached to the Internet?

1.1.3 What is a Protocol?

Now that we've got a bit of a feel for what the Internet is,let's consider another important buzzword in computer networking:protocol.

What is a protocol? What does a protocol do?

A Human Analogy

- ➢ 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 3.

- Human protocol (or good manners,at least) dictates that one first offer a greeting (the first "Hi" in Figure 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 otherevents 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 under-stands the concept of time and the other does not) the protocols do not interoperate and no useful work can beaccomplished.
- The same is true in networking—it takes two (or more) communicating entities running the same protocol in order to accomplish a task.
- Let's consider a **second human analogy**.
- Suppose you're in a college class (a computer networking class,for example!).
- The teacher is droning on about protocols and you're confused.
- The teacher stops to ask,"Are there any questions?" (a message that is transmitted to,and received by,all students who are not sleeping).
- You raise your hand (transmitting an implicit message to the teacher).
- Your teacher acknowledges you with a smile,saying "Yes . . ." (a transmitted message encouraging you to ask your question—teachers love to be asked questions), and you then ask your question (that is,transmit your message to your teacher).
- Your teacher hears your question (receives your question message) and answers (transmits a reply to you).
- Once again,we see that the transmission and receipt of messages,and a set of conventional actions taken whenthese messages are sent and received,are at the heart of this question-and-answer protocol.

Network Protocols

- A network protocol is similar to a human protocol,except that the entities exchanging messages and taking actions are hardware or software components of some device (for example,computer,smartphone,tablet,router,or other network-capable device).
- All activity in the Internet that involves two or more communicating remote entities is governed by a protocol.
- For example,hardware-implemented protocols in two physically connected computers control the flow of bits on the "wire" between the two network interface cards;congestion-control protocols in end systems control the rate at which packets are transmitted between sender and receiver;protocols in routers determine a packet's path from source to destination.

- Protocols are running everywhere in the Internet,and consequently much of this course is about computernetwork protocols.
- As an example of a computer network protocol with which you are probably familiar,consider what happenswhen you make a request to a Web server,that is, when you type the URL of a Web page into your Web browser.
- The scenario is illustrated in the right half of Figure 3.
- First,your computer will send a connection request message to the Web server and wait for a reply.
- The Web server will eventually receive your connection request message and return a connection reply message.
- Knowing that it is now OK to request the Web document,your computer then sends the name of the Web pageit wants to fetch from that Web server in a GET message.
- Finally,the Web server returns the Web page (file) to your computer.
- Given the human and networking examples above,the exchange of messages and the actions taken when these messages are sent and received are the key defining elements of a protocol:
- A **protocol** defines the format and the order of messages exchanged between two or more communicatingentities,as well as the actions taken on the transmission and/or receipt of a message or other event.
- The Internet,and computer networks in general,make extensive use of protocols.
- Different protocols are used to accomplish different communication tasks.
- As we go through this course,you will learn that some protocols are simple and straightforward,while others arecomplex and intellectually deep.
- Mastering the field of computer networking is equivalent to understanding the what,why,and how of networkingprotocols.

1.2 The Network Edge

- In the previous section we presented a high-level overview of the Internet and networking protocols.
- We are now going to delve a bit more deeply into the components of a computer network (and the Internet,in particular).
- We begin in this section at the edge of a network and look at the components with which we are most famil-iar—namely,the computers,smartphones and other devices that we use on a daily basis.
- In the next section we'll move from the network edge to the network core and examine switching and routingin computer networks.
- Recall from the previous section that in computer networking jargon,the computers and other devices connectedto the Internet are often referred to as end systems.
- They are referred to as end systems because they sit at the edge of the Internet,as shown in Figure 4.
- The Internet's end systems include desktop computers (e.g.,desktop PCs,Macs, and Linux boxes),servers (e.g.,Weband e-mail servers),and mobile computers (e.g., laptops,smartphones,and tablets).
- Furthermore,an increasing number of non-traditional devices are being attached to the Internet as end systems.
- End systems are also referred to as hosts because they host (that is,run) application programs such as a

Webbrowser program,a Web server program,an e-mail client program,or an e-mail server program.

- Throughout this course we will use the terms hosts and end systems interchangeably;that is,host = end system.
- Hosts are sometimes further divided into two categories: clients and servers.

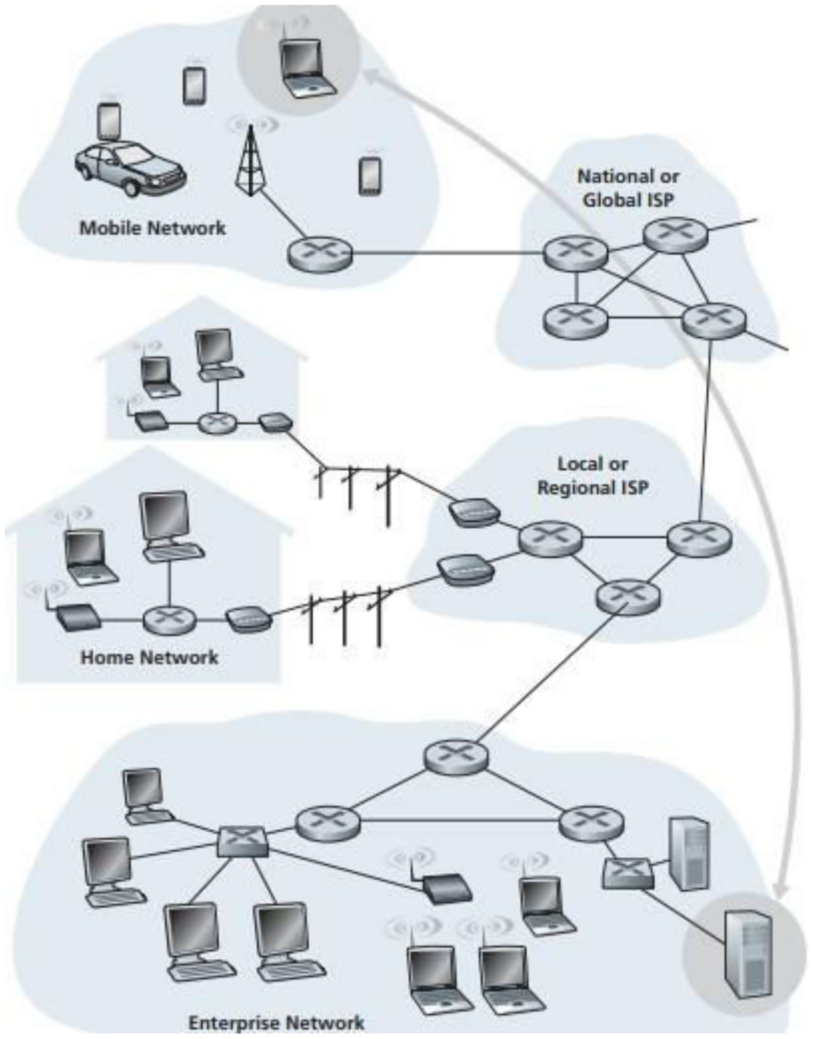


Figure 4: End-system interaction

- Informally,clients tend to be desktop and mobile PCs,smartphones,and so on, whereas servers tend to be more powerful machines that store and distribute Web pages,stream video,relay e-mail,and so on.
- Today,most of the servers from which we receive search results,e-mail,Web pages,and videos reside in large datacenters.
- For example,Google has 30–50 data centers,with many having more than one hundred thousand servers.

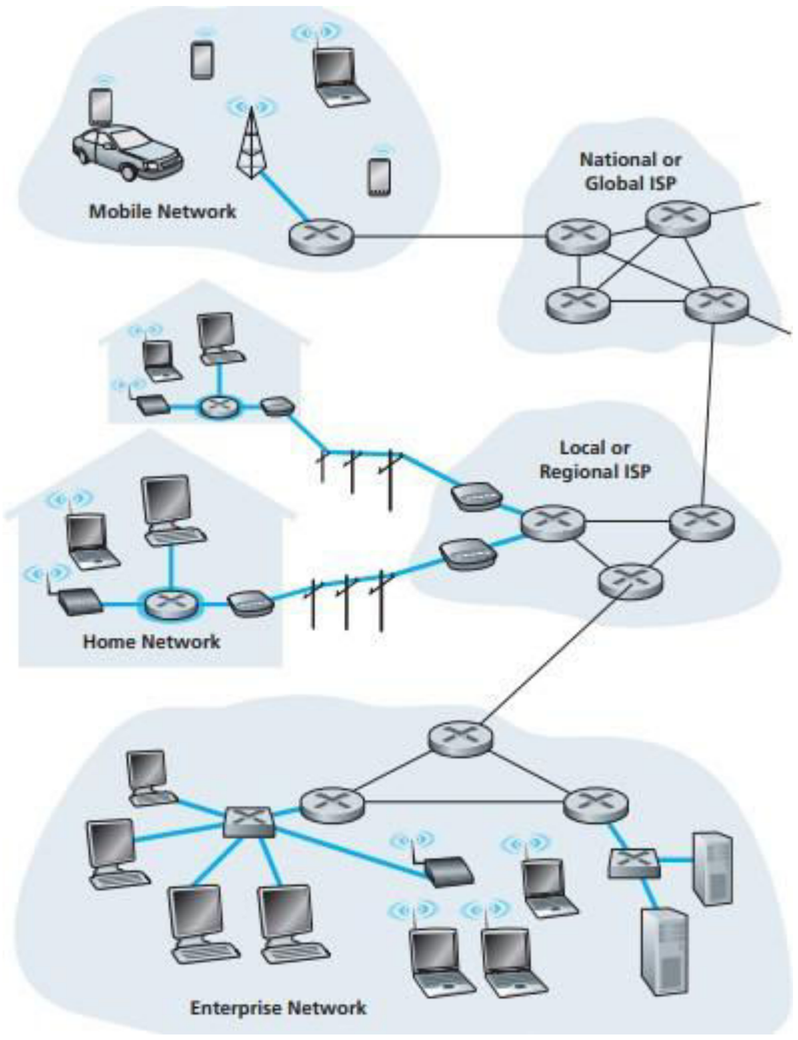


Figure 5: Access networks

1.2.1 Access Networks

- Having considered the applications and end systems at the ”edge of the network”,let’s next consider the access network — the network that physically connects an end system to the first router (also known as the ”edge router”) on a path from the end system to any other distant end system.
- Figure 5 shows several types of access networks with thick,shaded lines,and the settings (home,enterprise,and wide-area mobile wireless) in which they are used.

Home Access: DSL,Cable,FTTH,Dial-Up,and Satellite

- In developed countries today,more than 65 percent of the households have Internet access,with Korea,Netherlands,Finland,and Sweden leading the way with more than 80 percent of households having Internet access,almost all via a high-speed broadband connection.
- Finland and Spain have recently declared high-speed Internet access to be a ”legal right”

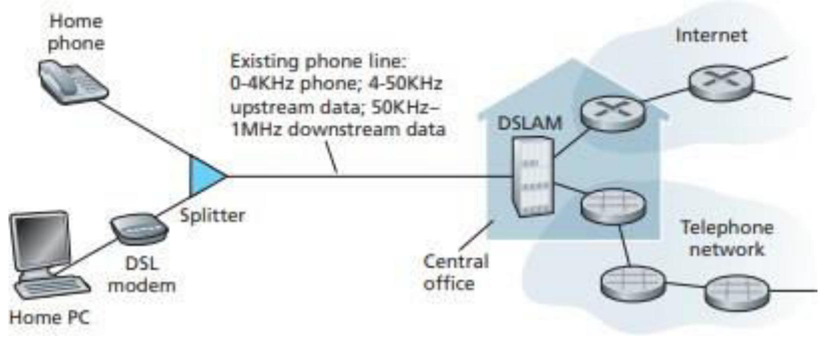

Home phone
Existing phone line: 0-4KHz phone; 4-50KHz upstream data; 50KHz–1MHz downstream data
Internet
DSLAM
Splitter
DSL modem
Home PC
Central office
Telephone network

Figure 6: DSL Internet access

- Given this intense interest in home access,let's begin our overview of access networks by considering how homesconnect to the Internet.
- Today,the two most prevalent types of broadband residential access are digital subscriber line (DSL) and cable.
- A residence typically obtains DSL Internet access from the same local telephone company (telco) that providesits wired local phone access.
- Thus,when DSL is used,a customer's telco is also its ISP.
- As shown in Figure 6,each customer's 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).
- The home's DSL modem takes digital data and translates it to high frequency tones for transmission over telephone wires to the CO;the analog signals from many such houses are translated back into digital format at the DSLAM.

The residential telephone line carries both data and traditional telephone signals simultaneously,which are encoded at different frequencies:

1. A high-speed downstream channel,in the 50 kHz to 1 MHz band
2. A medium-speed upstream channel,in the 4 kHz to 50 kHz band
3. An ordinary two-way telephone channel,in the 0 to 4 kHz band

- This approach makes the single DSL link appear as if there were three separate links,so that a telephone calland an Internet connection can share the DSL link at the same time.
- On the customer side,a splitter separates the data and telephone signals arriving to the home and forwards the data signal to the DSL modem.
- On the telco side,in the CO,the DSLAM separates the data and phone signals and sends the data into the Internet.
- Hundreds or even thousands of households connect to a single DSLAM.
- The DSL standards define transmission rates of 12 Mbps downstream and 1.8 Mbps upstream as per ITU 1999,and 24 Mbps downstream and 2.5 Mbps upstream as per ITU 2003.
- Because the downstream and upstream rates are different,the access is said to be asymmetric.
- The actual downstream and upstream transmission rates achieved may be less than the rates noted above,as the DSL provider may purposefully limit a residential rate when tiered service (different rates,available at different prices) are offered,or because the maximum rate can be limited by the distance between the home and the CO,the gauge of the twisted-pair line and the degree of electrical interference.

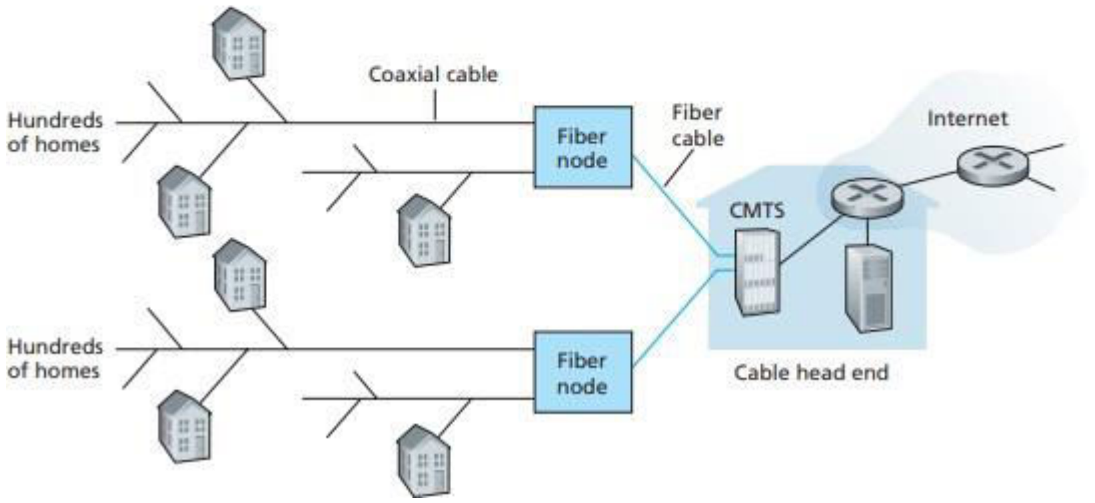


Figure 7: A hybrid fiber-coaxial access network

- Engineers have expressly designed DSL for short distances between the home and the CO;generally,if the resi- dence is not located within 5 to 10 miles of the CO,the residence must resort to an alternative form of Internet access.
- While DSL makes use of the telco's existing local telephone infrastructure, cable Internet access makes use of the cable television company's existing cable television infrastructure.
- A residence obtains cable Internet access from the same company that provides its cable television.
- As illustrated in Figure 7,fiber optics connect the cable head end to neighborhood-level junctions,from which traditional coaxial cable is then used to reach individual houses and apartments.
- Each neighborhood junction typically supports 500 to 5,000 homes.
- Because both fiber and coaxial cable are employed in this system,it is often referred to as hybrid fiber coax (HFC).
- Cable internet access requires special modems,called cable modems.
- As with a DSL modem,the cable modem is typically an external device and connects to the home PC throughan Ethernet port.
- At the cable head end,the cable modem termination system (CMTS) serves a similar function as the DSL network's DSLAM — turning the analog signal sent from the cable modems in many downstream homes back into digital format.
- Cable modems divide the HFC network into two channels,a downstream and an upstream channel.
- As with DSL,access is typically asymmetric,with the downstream channel typically allocated a higher transmis- sion rate than the upstream channel.
- The DOCSIS 2.0 standard defines downstream rates up to 42.8 Mbps and upstream rates of up to 30.7 Mbps.
- As in the case of DSL networks,the maximum achievable rate may not be realized due to lower contracted data rates or media impairments.
- One important characteristic of cable Internet access is that it is a shared broadcast medium.
- In particular,every packet sent by the head end travels downstream on every link to every home and every packet sent by a home travels on the upstream channel to the head end.

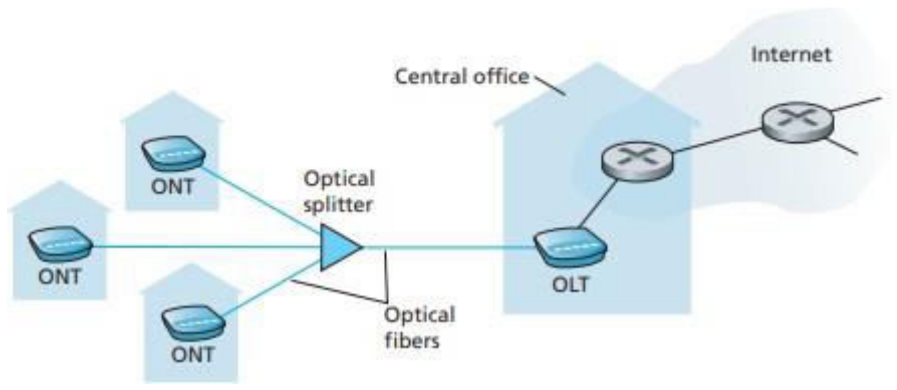


Figure 8: FTTH Internet access

- For this reason,if several users are simultaneously downloading a video file on the downstream channel,the actual rate at which each user receives its video file will be significantly lower than the aggregate cable downstream rate.
- On the other hand,if there are only a few active users and they are all Web surfing,then each of the users may actually receive Web pages at the full cable downstream rate,because the users will rarely request a Web page at exactly the same time.
- Because the upstream channel is also shared,a distributed multiple access protocol is needed to coordinate transmissions and avoid collisions.
- Although DSL and cable networks currently represent more than 90 percent of residential broadband access in the United States,an up-and-coming technology that promises even higher speeds is the deployment of fiber to the home (FTTH).
- As the name suggests,the FTTH concept is simple — provide an optical fiber path from the CO directly to thehome.
- In the United States,Verizon has been particularly aggressive with FTTH with its FIOS service.
- There are several competing technologies for optical distribution from the CO to the homes.
- The simplest optical distribution network is called direct fiber,with one fiber leaving the CO for each home.
- More commonly,each fiber leaving the central office is actually shared by many homes;it is not until the fiber gets relatively close to the homes that it is split into individual customer-specific fibers.
- There are two competing optical-distribution network architectures that perform this splitting: active optical networks (AONs) and passive optical networks (PONs).
- AON is essentially switched Ethernet.
- Here,we briefly discuss PON,which is used in Verizon's FIOS service.
- Figure 8 shows FTTH using the PON distribution architecture.
- Each home has an optical network terminator(ONT),which is connected by dedicated optical fiber to a neigh-borhood splitter.
- The splitter combines a number of homes (typically less than 100) onto a single,shared optical fiber,which connects to an optical line terminator (OLT) in the telco's CO.
- The OLT,providing conversion between optical and electrical signals,connects to the Internet via a telco router.

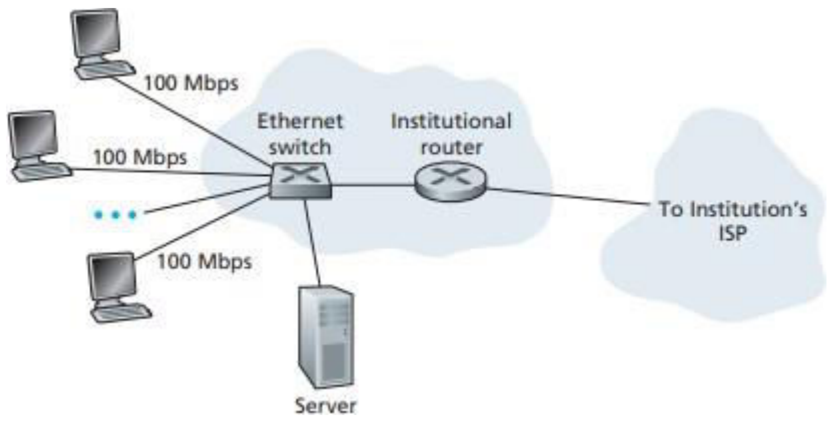


Figure 9: Ethernet Internet access

- In the home,users connect a home router (typically a wireless router) to the ONT and access the Internet via this home router.
- In the PON architecture,all packets sent from OLT to the splitter are replicated at the splitter (similar to a cable head end).
- FTTH can potentially provide Internet access rates in the gigabits per second range.
- However,most FTTH ISPs provide different rate offerings,with the higher rates naturally costing more money.
- The average downstream speed of US FTTH customers was approximately 20 Mbps in 2011 (compared with 13 Mbps for cable access networks and less than 5 Mbps for DSL).
- Two other access network technologies are also used to provide Internet access to the home.
- In locations where DSL,cable,and FTTH are not available (e.g.,in some rural settings),a satellite link can be used to connect a residence to the Internet at speeds of more than 1 Mbps;StarBand and HughesNet are two such satellite access providers.
- Dial-up access over traditional phone lines is based on the same model as DSL — a home modem connects over a phone line to a modem in the ISP.
- Compared with DSL and other broadband access networks,dial-up access is excruciatingly slow at 56 kbps.

Access in the Enterprise (and the Home): Ethernet and WiFi:

- On corporate and university campuses,and increasingly in home settings, a local area network (LAN) is used to connect an end system to the edge router.
- Although there are many types of LAN technologies,Ethernet is by far the most prevalent access technology in corporate,university,and home networks.
- As shown in Figure 9,Ethernet users use twisted-pair copper wire to connect to an Ethernet switch.
- The Ethernet switch,or a network of such interconnected switches,is then in turn connected into the larger Internet.
- With Ethernet access,users typically have 100 Mbps access to the Ethernet switch,whereas servers may have 1 Gbps or even 10 Gbps access.
- Increasingly,however,people are accessing the Internet wirelessly from laptops,smartphones,tablets,and other devices.

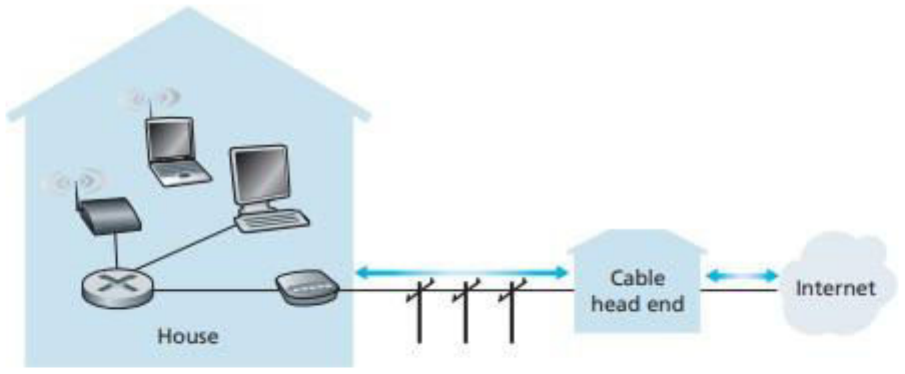


Figure 10: A typical home network

- In a wireless LAN setting,wireless users transmit/receive packets to/from an access point that is connected into the enterprise's network (most likely including wired Ethernet),which in turn is connected to the wired Internet.
- A wireless LAN user must typically be within a few tens of meters of the access point.
- Wireless LAN access based on IEEE 802.11 technology,more colloquially known as WiFi,is now just about everywhere — universities,business offices,cafes,airports, homes,and even in airplanes.
- In many cities,one can stand on a street corner and be within range of ten or twenty base stations (for a browseable global map of 802.11 base stations that have been discovered and logged on a Web site by people who take great enjoyment in doing such things.
- 802.11 today provides a shared transmission rate of up to 54 Mbps.
- Even though Ethernet and WiFi access networks were initially deployed in enterprise (corporate,university) settings,they have recently become relatively common components of home networks.
- Many homes combine broadband residential access (that is,cable modems or DSL) with these inexpensive wireless LAN technologies to create powerful home networks.
- Figure 10 shows a typical home network.
- This home network consists of a roaming laptop as well as a wired PC;a base station (the wireless access point),which communicates with the wireless PC;a cable modem,providing broadband access to the Internet;and a router,which interconnects the base station and the stationary PC with the cable modem.
- This network allows household members to have broadband access to the Internet with one member roaming from the kitchen to the backyard to the bedrooms.

Wide-Area Wireless Access:3G and LTE:

- Increasingly,devices such as iPhones,BlackBerrys,and Android devices are being used to send email,surf the Web,Tweet,and download music while on the run.
- These devices employ the same wireless infrastructure used for cellular telephony to send/receive packets through a base station that is operated by the cellular network provider.
- Unlike WiFi,a user need only be within a few tens of kilometers (as opposed to a few tens of meters) of the base station.
- Telecommunications companies have made enormous investments in so-called third-generation (3G) wireless,which provides packet-switched wide-area wireless Internet access at speeds in excess of 1 Mbps.
- But even higher-speed wide-area access technologies — a fourth-generation (4G) of wide-area wireless networks — are already being deployed.

- LTE (for ”Long-Term Evolution” — a candidate for Bad Acronym of the Year Award) has its roots in 3G technology,and can potentially achieve rates in excess of 10 Mbps.
- LTE downstream rates of many tens of Mbps have been reported in commercial deployments.

1.2.2 Physical Media

- In the previous subsection,we gave an overview of some of the most important network access technologies in the Internet.
- As we described these technologies,we also indicated the physical media used.
- For example,we said that HFC uses a combination of fiber cable and coaxial cable.
- We said that DSL and Ethernet use copper wire.
- And we said that mobile access networks use the radio spectrum.
- In this subsection we provide a brief overview of these and other transmission media that are commonly used in the Internet.
- In order to define what is meant by a physical medium,let us reflect on the brief life of a bit.
- Consider a bit traveling from one end system,through a series of links and routers,to another end system.
- This poor bit gets kicked around and transmitted many,many times.
- The source end system first transmits the bit,and shortly thereafter the first router in the series receives the bit;the first router then transmits the bit,and shortly thereafter the second router receives the bit;and so on.
- Thus our bit,when traveling from source to destination,passes through a series of transmitter-receiver pairs.
- For each transmitter-receiver pair,the bit is sent by propagating electromagnetic waves or optical pulses across a physical medium.
- The physical medium can take many shapes and forms and does not have to be of the same type for each transmitter-receiver pair along the path.
- Examples of physical media include twisted-pair copper wire,coaxial cable, multimode fiber-optic cable,terrestrial radio spectrum,and satellite radio spectrum.
- Physical media fall into two categories: guided media and unguided media.
- With guided media,the waves are guided along a solid medium,such as a fiber-optic cable,a twisted-pair copper wire,or a coaxial cable.
- With unguided media,the waves propagate in the atmosphere and in outer space, such as in a wireless LAN or a digital satellite channel.
- But before we get into the characteristics of the various media types,let us say a few words about their costs.
- The actual cost of the physical link (copper wire,fiber-optic cable,and so on) is often relatively minor compared with other networking costs.
- In particular,the labor cost associated with the installation of the physical link can be orders of magnitude higher than the cost of the material.
- For this reason,many builders install twisted pair,optical fiber,and coaxial cable in every room in a building.

- Even if only one medium is initially used,there is a good chance that another medium could be used in the near future, and so money is saved by not having to lay additional wires in the future.

Twisted-Pair Copper Wire

- The least expensive and most commonly used guided transmission medium is twisted-pair copper wire.
- For over a hundred years it has been used by telephone networks.
- In fact,more than 99 percent of the wired connections from the telephone handset to the local telephone switch use twisted-pair copper wire.
- Most of us have seen twisted pair in our homes and work environments.
- Twisted pair consists of two insulated copper wires,each about 1 mm thick, arranged in a regular spiral pattern.
- The wires are twisted together to reduce the electrical interference from similar pairs close by.
- Typically,a number of pairs are bundled together in a cable by wrapping the pairs in a protective shield.
- A wire pair constitutes a single communication link.
- Unshielded twisted pair (UTP) is commonly used for computer networks within a building,that is,for LANs.
- Data rates for LANs using twisted pair today range from 10 Mbps to 10 Gbps.
- The data rates that can be achieved depend on the thickness of the wire and the distance between transmitter and receiver.
- When fiber-optic technology emerged in the 1980s,many people disparaged twisted pair because of its relatively low bit rates.
- Some people even felt that fiberoptic technology would completely replace twisted pair.
- But twisted pair did not give up so easily.
- Modern twisted-pair technology,such as category 6a cable,can achieve data rates of 10 Gbps for distances up to a hundred meters.
- In the end,twisted pair has emerged as the dominant solution for high-speed LAN networking.
- As discussed earlier,twisted pair is also commonly used for residential Internet access.
- We saw that dial-up modem technology enables access at rates of up to 56 kbps over twisted pair.
- We also saw that DSL (digital subscriber line) technology has enabled residential users to access the Internet at tens of Mbps over twisted pair (when users live close to the ISP’s modem).

Coaxial Cable

- Like twisted pair,coaxial cable consists of two copper conductors,but the two conductors are concentric rather than parallel.
- With this construction and special insulation and shielding,coaxial cable can achieve high data transmission rates.
- Coaxial cable is quite common in cable television systems.
- As we saw earlier,cable television systems have recently been coupled with cable modems to provide residential users with Internet access at rates of tens of Mbps.

- In cable television and cable Internet access,the transmitter shifts the digital signal to a specific frequency band,and the resulting analog signal is sent from the transmitter to one or more receivers.
- Coaxial cable can be used as a guided shared medium.
- Specifically,a number of end systems can be connected directly to the cable, with each of the end systems receiving whatever is sent by the other end systems.

Fiber Optics

- An optical fiber is a thin,flexible medium that conducts pulses of light,with each pulse representing a bit.
- A single optical fiber can support tremendous bit rates,up to tens or even hundreds of gigabits per second.
- They are immune to electromagnetic interference,have very low signal attenuation up to 100 kilometers,and arevery hard to tap.
- These characteristics have made fiber optics the preferred longhaul guided transmission media,particularly for overseas links.
- Many of the long distance telephone networks in the United States and elsewhere now use fiber optics exclusively.
- Fiber optics is also prevalent in the backbone of the Internet.
- However,the high cost of optical devices — such as transmitters,receivers,and switches — has hindered their deployment for short-haul transport,such as in a LAN or into the home in a residential access network.
- The Optical Carrier (OC) standard link speeds range from 51.8 Mbps to 39.8 Gbps;these specifications are often referred to as OCn,where the link speed equals $n * 51.8$ Mbps.
- Standards in use today include $OC-1$,$OC-3$,$OC-12$,$OC-24$,$OC-48$,$OC-96$, $OC-192$,$OC-768$ providecoverage of various aspects of optical networking.

Terrestrial Radio Channels

- Radio channels carry signals in the electromagnetic spectrum.
- They are an attractive medium because they require no physical wire to be installed,can penetrate walls,provide connectivity to a mobile user,and can potentially carry a signal for long distances.
- The characteristics of a radio channel depend significantly on the propagation environment and the distance over which a signal is to be carried.
- Environmental considerations determine path loss and shadow fading (which decrease the signal strength as the signal travels over a distance and around/through obstructing objects),multipath fading (due to signal reflection off

of interfering objects),and interference (due to other transmissions and electromagnetic signals).

- ➢ Terrestrial radio channels can be broadly classified into three groups: those that operate over very short distance(e.g.,with one or two meters); those that operate in local areas,typically spanning from ten to a few hundred meters;and those that operate in the wide area,spanning tens of kilometers.
- ➢ Personal devices such as wireless headsets,keyboards,and medical devices operate over short distances;the wire- less LAN technologies use local-area radio channels;the cellular access technologies use wide-area radio channels.

Multiplexing in Circuit-Switched Networks

- A circuit in a link is implemented with either **frequency-division multiplexing (FDM)** or **time-division multiplexing (TDM).** With FDM, the frequency spectrum of a link is divided up among the connections established across the link.
- The link dedicates a **frequency band** to each connection for the duration of the connection.
- For a **TDM link, time is divided into frames** of fixed duration, and each **frame is divided into a fixed number of time slots**.
- When the network establishes a connection across a link, the network dedicates **one time slot in every frame** to this connection. These slots are dedicated for the sole use of that connection, with one time slot available for use (in every frame) to transmit the connection's data.

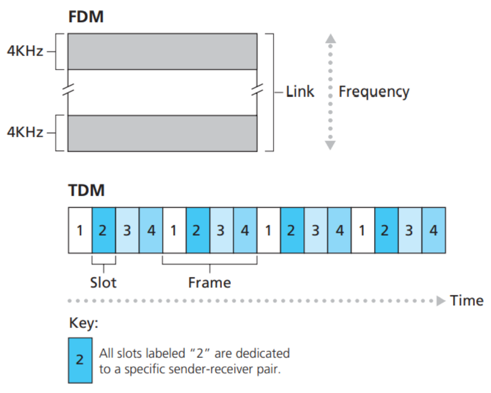


Figure 1.14 ♦ With FDM, each circuit continuously gets a fraction of the bandwidth. With TDM, each circuit gets all of the bandwidth periodically during brief intervals of time (that is, during slots)

Numerical example

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.

Packet Switching Versus Circuit Switching

packet switching:

- It offers better sharing of **transmission capacity** than circuit switching and
- It is **simpler, more efficient, and less costly** to implement than circuit switching.
- Packet switching on the other hand allocates link use **on demand**. Link transmission capacity will be shared on a packet-by-packet basis only among those users who have packets that need to be transmitted over the link.
- Packet switching provides essentially the same performance as circuit switching, but does so while allowing for **more than three times the number of users**.

Circuit switching:

- When there are **fewer active users**, users' packets flow through the link essentially without delay.
- Circuit switching **pre-allocates use of the transmission link** regardless of demand, with allocated but unneeded link time going unused.

A Network of Networks

- The **access ISP** can provide either **wired or wireless connectivity**, using an array of access technologies including **DSL, cable, FTTH, Wi-Fi, and cellular**
- The access ISPs themselves must be interconnected; this is done by creating a **network of networks**
- The overarching goal is to interconnect the **access ISPs so that all end systems** can send packets to each other.
- One naive approach would be to have each access ISP directly connect with every other access ISP. Such a **mesh design** is, of course, much **too costly** for the access ISPs, as it would require each access ISP to have a separate communication link to each of the hundreds of thousands of other access ISPs all over the world
- <u>**Network Structure 1**</u>, interconnects all of the access ISPs with a **single global transit ISP**. Of course, it would be **very costly** for the global ISP to build such an extensive network. To be profitable, it would naturally **charge each of the access ISPs for connectivity**, with the pricing reflecting (but not necessarily directly proportional to) the amount of traffic an access ISP exchanges with the global ISP
- <u>**Network Structure 2**</u>, which consists of the hundreds of thousands of access ISPs and **multiple global transit ISPs**. The global transit **ISPs themselves must interconnect**. Otherwise access ISPs connected to one of the global transit providers would **not be able to communicate** with access ISPs connected to the other global transit providers
- <u>**Network Structure 3**</u>**:** There are approximately a **dozen tier-1 ISPs**, including Level 3 Communications, **AT&T, Sprint, and NTT.** Note that the tier-1 ISPs **do not pay anyone** as they are at the top of the hierarchy.

- In some regions, there may be a **larger regional ISP** (possibly spanning an **entire country**) to which the **smaller regional ISPs** in that region connect; the **larger regional ISP then connects to a tier-1 ISP.** We refer to this **multi-tier hierarchy**
- **Network Structure 4:** We must add **points of presence (PoPs)**, multi-homing, peering, and **Internet exchange points (IXPs)** to the hierarchical Network Structure 3
- A **PoP is simply a group of one or more routers** (at the same location) in the provider's network where customer ISPs can connect into the provider ISP.
- A third-party company can create an **Internet Exchange Point (IXP)** (typically in a stand-alone building with its own switches), which is a **meeting point** where multiple ISPs can peer together. There are roughly 300 IXPs in the Internet today. We refer to this **ecosystem**—consisting of access ISPs, regional ISPs, tier-1 ISPs, PoPs, multi-homing, peering, and IXPs
- **Network Structure 5:** It builds on top of Network Structure 4 by adding content provider networks. **Google is currently one of the leading examples** of such a content provider network.

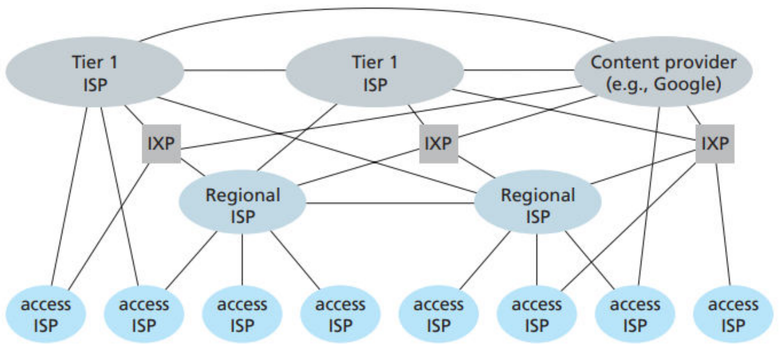


Figure 1.15 ♦ Interconnection of ISPs

- Google has **30 to 50 data centers** distributed across North America, Europe, Asia, South America, and Australia.
- Some of these data centers house over one hundred **thousand servers**, while other data centers are smaller, housing only hundreds of servers.
- The Google data centers are all **interconnected via Google's private TCP/IP network**
- The Google private network attempts to **"bypass" the upper tiers** of the Internet by peering (settlement free) with lower-tier ISPs, either by directly connecting with them or by connecting with them at IXPs
- Google network also **connects to tier-1 ISPs,** and **pays** those ISPs for the traffic it exchanges with them

In summary,

today's Internet—a network of networks—is complex, consisting of a **dozen or so tier-1 ISPs and hundreds of thousands of lower-tier ISPs**.

The ISPs are diverse in their coverage, with some spanning **multiple continents** and oceans, and others limited to narrow geographic regions.

The lower-tier ISPs connect to the higher-tier ISPs, and the higher-tier ISPs interconnect with one another.

Users and **content providers** are **customers of lower-tier ISPs**, and **lower-tier ISPs are customers of higher-tier ISPs.** In recent years, major content providers have also created their own networks and connect directly into lower-tier ISPs where possible.

Unit 1 – Class 3

1.4 Delay, Loss, and Throughput in Packet-Switched Networks

1.4.1 Overview of Delay in Packet-Switched Networks

- As a packet travels from one node (host or router) to the subsequent node (host or router) along this path, the packet suffers from several types of **delays** at each node along the path.
- The most important of these delays are the nodal **processing delay, queuing delay, transmission delay, and propagation delay**; together, these delays accumulate to give a total **nodal delay**

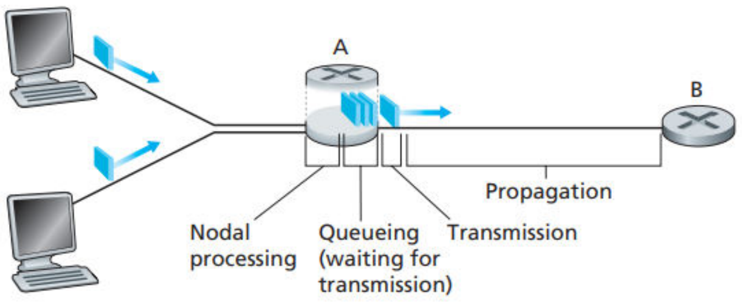


Figure 1.16 ♦ The nodal delay at router A

Processing Delay

- The time required to **examine the packet's header** and determine where to direct the packet is part of the processing delay
- the time needed to **check for bit-level errors** in the packet

Queuing Delay

- The packet experiences a queuing delay as it **waits** to be transmitted onto the link
- Length of the queuing delay of a specific packet will depend on the **number of earlier-arriving packets that are queued** and waiting for transmission onto the link.
- Queuing delays can be on the order of **microseconds to milliseconds** in practice.

Transmission Delay

- Assuming Packets are transmitted in a **first-come-first-served manner**
- Transmission delays are typically on the order of **microseconds to milliseconds** in practice
- Denote the length of the packet by **L bits**, and denote the transmission rate of the link from router A to router B by R bits/sec.
- For example, **for a 10 Mbps Ethernet link, the rate is R = 10 Mbps; for a 100 Mbps Ethernet link, the rate is R = 100 Mbps**.
- The transmission delay is **L/R**

Propagation Delay

- Once a bit is pushed into the link, it needs to propagate to router B. The **time required to propagate** from the beginning of the link to router B is the propagation delay
- The propagation speed depends on the **physical medium** of the link (that is**, fiber optics, twisted-pair copper wire, and so on**) and
- The range is 2 x10^8 meters/sec to 3x10^8 meters/sec which is equal to, or a little less than, the speed of light.
- The propagation delay is the **distance between two routers divided by the propagation speed**. That is, the **propagation delay is d/s**, where d is the distance between router A and router B and s is the propagation speed of the link
- In wide-area networks, propagation delays are on the order of **milliseconds**.

1.4.2 Queuing Delay and Packet Loss T

- For example, if 10 packets arrive at an empty queue at the same time, the **first packet transmitted will suffer no queuing delay**, while the last packet transmitted will suffer a relatively large queuing delay
- If packets arrive in bursts but periodically, there can be a significant average queuing delay
- let a denote the average rate at which packets arrive at the queue (a is in units of packets/sec).
- R is the transmission rate; that is, it is the rate (in bits/sec)
- All packets consist of L bits
- The ratio La/R, called the **traffic intensity**

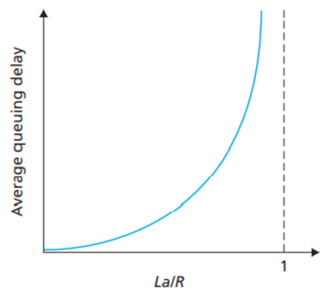
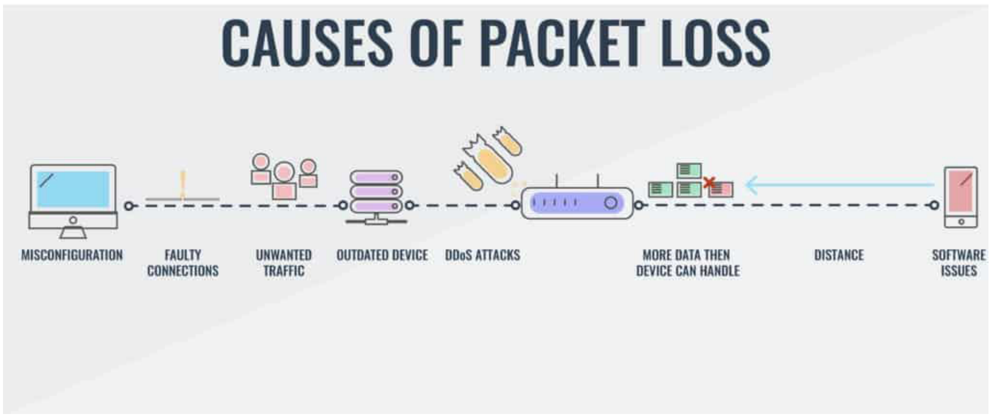


Figure 1.18 ♦ Dependence of average queuing delay on traffic intensity

Packet Loss

- packet having been transmitted into the network core but **never emerging** from the network at the **destination**.



1.4.3 End-to-End Delay

- Total delay from source to destination. Suppose there are **N - 1 routers** between the source host and the destination host

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

- The processing delay at each router and at the source host is d_{proc},
- d_{trans} = L/R, where L is the packet size.
- The propagation on each link is d_{prop}.

1.4.4 Throughput in Computer Networks

- **Throughput** or **network throughput** is the rate of successful message delivery over a communication channel.
- If the file consists of **F bits** and the transfer takes **T seconds** for Host B to receive all F bits, then the **average throughput** of the file transfer is **F/T bits/sec**

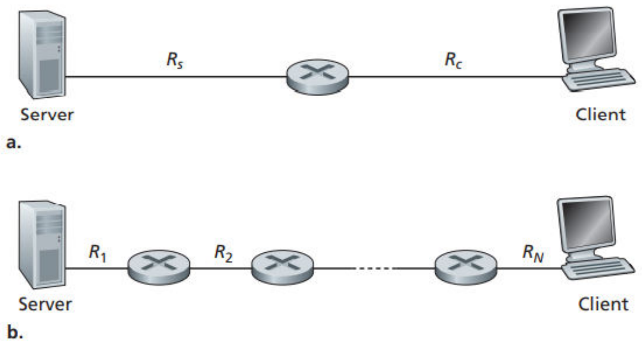


Figure 1.19 ♦ Throughput for a file transfer from server to client

- If **Rs < Rc**, then the bits pumped by the server will “flow” right through the router and arrive at the client at a rate of Rs bps, giving a throughput of **Rs bps**.
- If, on the other hand, **Rc < Rs** , then the router will not be able to forward bits as quickly as it receives them, giving an end-to-end throughput of **Rc**
- Thus, for this simple two-link network, the throughput is **min{Rc , Rs }**, that is, it is the transmission rate of the **bottleneck link.**
- For example, suppose you are downloading an MP3 file of **F = 32 million bits**, the server has a **transmission rate of Rs = 2 Mbps**, and you have an access link of **Rc = 1 Mbps**. The time needed to transfer the file is then **32 seconds**.

- Of course, these expressions for throughput and transfer time are only approximations, as they do not account for store-and-forward and processing delays as well as protocol issues.
- With N links between the server and the client, the throughput for a file transfer from server to client is **min{R1, R2,..., RN},** which is once again the transmission rate of the bottleneck link along the path between server and client

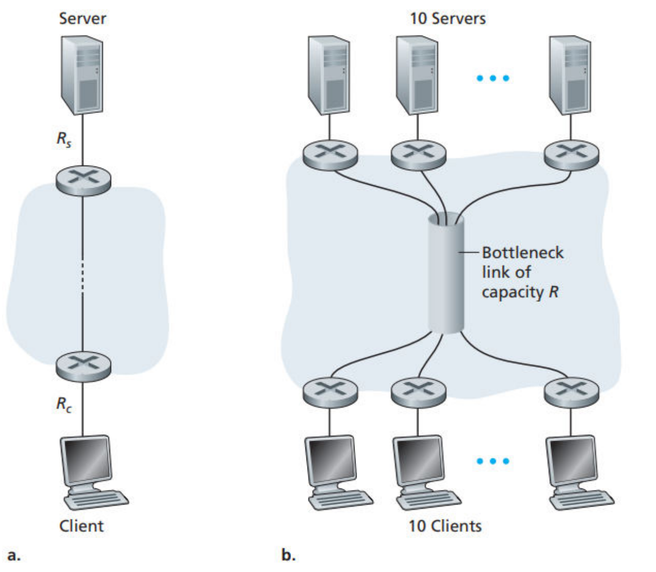


Figure 1.20 ♦ End-to-end throughput: (a) Client downloads a file from server; (b) 10 clients downloading with 10 servers

- R for the transmission rate of this link R.
- Let's suppose that all server access links have the same rate Rs ,
- all client access links have the same rate Rc

- Throughput depends on the transmission rates of the links over which the data flow

Unit 1 – Class 4

1.5 Protocol Layers and Their Service Models

1.5.1 Layered Architecture

Each layer provides its service by

- performing certain actions within that layer (for example, at the gate layer, loading and unloading people from an airplane) and by
- using the services of the layer directly below it (for example, in the gate layer, using the runway-to runway passenger transfer service of the takeoff/landing layer)

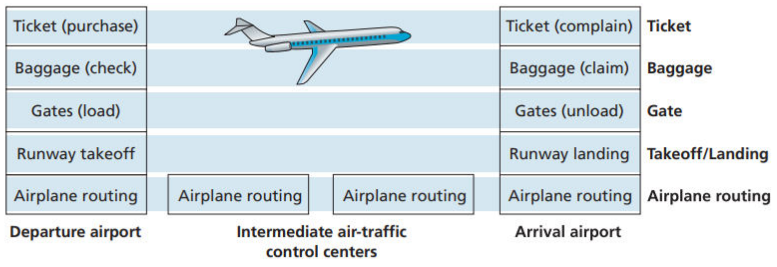


Figure 1.22 ♦ Horizontal layering of airline functionality

Protocol Layering

Just as in the case of our airline example, each layer provides its service by

- performing certain actions within that layer and by
- using the services of the layer directly below it.

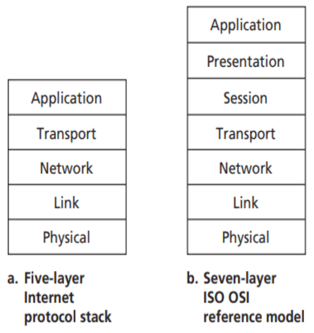
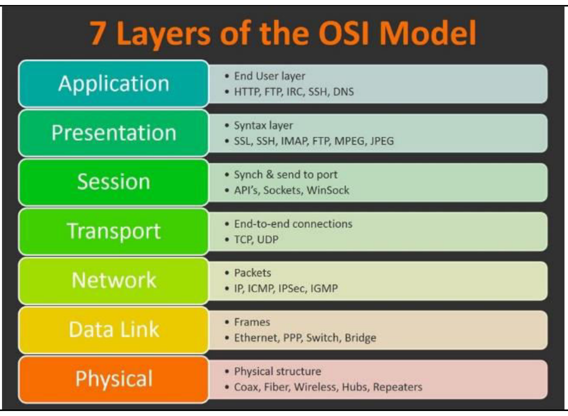
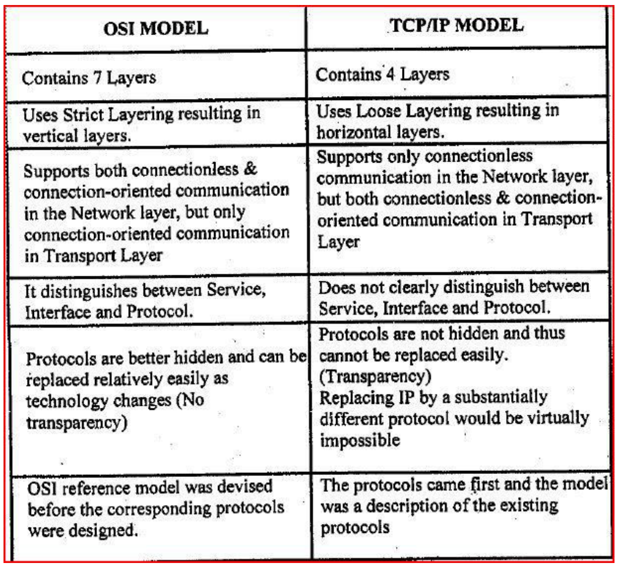


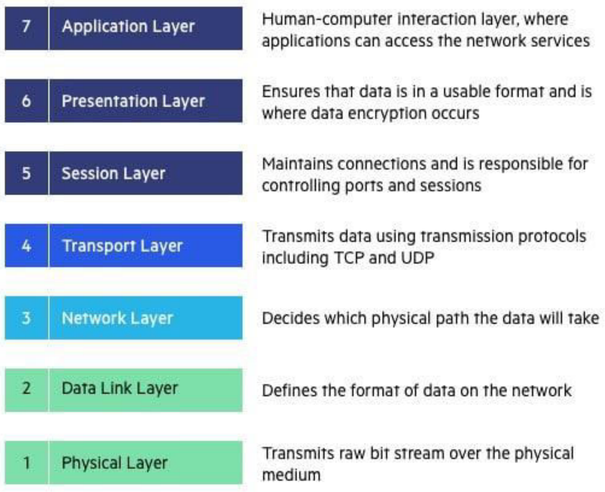
Figure 1.23 ♦ The Internet protocol stack (a) and OSI reference model (b)



- OSI refers to Open Systems Interconnection whereas
- TCP/IP refers to Transmission Control Protocol.



OSI MODEL	TCP/IP MODEL
Contains 7 Layers	Contains 4 Layers
Uses Strict Layering resulting in vertical layers.	Uses Loose Layering resulting in horizontal layers.
Supports both connectionless & connection-oriented communication in the Network layer, but only connection-oriented communication in Transport Layer	Supports only connectionless communication in the Network layer, but both connectionless & connection-oriented communication in Transport Layer
It distinguishes between Service, Interface and Protocol.	Does not clearly distinguish between Service, Interface and Protocol.
Protocols are better hidden and can be replaced relatively easily as technology changes (No transparency)	Protocols are not hidden and thus cannot be replaced easily. (Transparency) Replacing IP by a substantially different protocol would be virtually impossible
OSI reference model was devised before the corresponding protocols were designed.	The protocols came first and the model was a description of the existing protocols



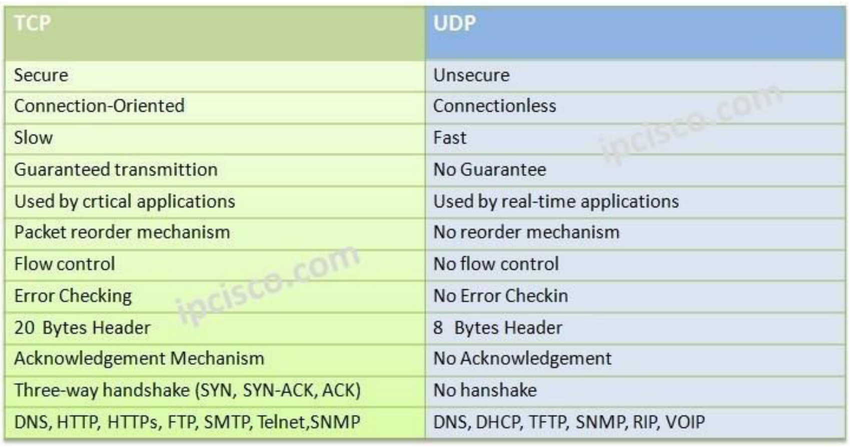
7	Application Layer	Human-computer interaction layer, where applications can access the network services
6	Presentation Layer	Ensures that data is in a usable format and is where data encryption occurs
5	Session Layer	Maintains connections and is responsible for controlling ports and sessions
4	Transport Layer	Transmits data using transmission protocols including TCP and UDP
3	Network Layer	Decides which physical path the data will take
2	Data Link Layer	Defines the format of data on the network
1	Physical Layer	Transmits raw bit stream over the physical medium

Application Layer

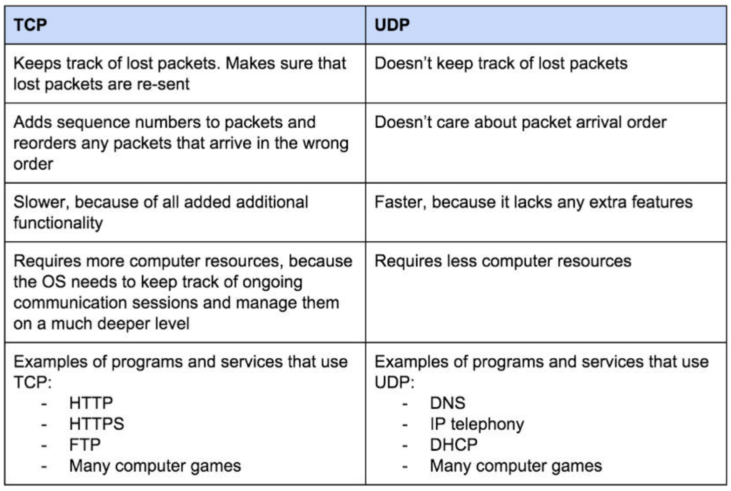
- The Internet's application layer includes many protocols, such as the
- HTTP protocol (which provides for Web document request and transfer),
- SMTP (which provides for the transfer of e-mail messages), and
- FTP (which provides for the transfer of files between two end systems).

Transport Layer

- In the Internet there are two transport protocols,
- **TCP and UDP**, either of which can transport application-layer messages
- **TCP** provides a connection-oriented service to its applications
- TCP also breaks **long messages into shorter segments** and provides a **congestion-control mechanism**
- The **UDP protocol** provides a **connectionless service** to its applications**, no reliability, no flow control, and no congestion control**



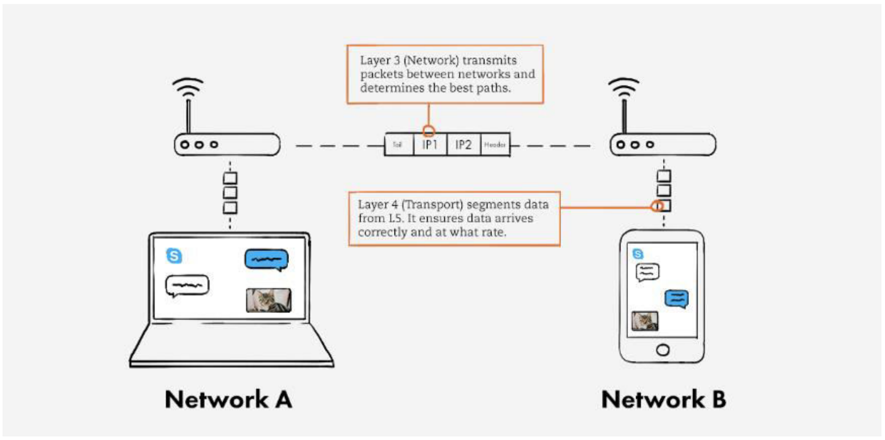
TCP	UDP
Secure	Unsecure
Connection-Oriented	Connectionless
Slow	Fast
Guaranteed transmittion	No Guarantee
Used by crtical applications	Used by real-time applications
Packet reorder mechanism	No reorder mechanism
Flow control	No flow control
Error Checking	No Error Checkin
20 Bytes Header	8 Bytes Header
Acknowledgement Mechanism	No Acknowledgement
Three-way handshake (SYN, SYN-ACK, ACK)	No hanshake
DNS, HTTP, HTTPs, FTP, SMTP, Telnet,SNMP	DNS, DHCP, TFTP, SNMP, RIP, VOIP



TCP	UDP
Keeps track of lost packets. Makes sure that lost packets are re-sent	Doesn't keep track of lost packets
Adds sequence numbers to packets and reorders any packets that arrive in the wrong order	Doesn't care about packet arrival order
Slower, because of all added additional functionality	Faster, because it lacks any extra features
Requires more computer resources, because the OS needs to keep track of ongoing communication sessions and manage them on a much deeper level	Requires less computer resources
Examples of programs and services that use TCP: - HTTP - HTTPS - FTP - Many computer games	Examples of programs and services that use UDP: - DNS - IP telephony - DHCP - Many computer games

Network Layer

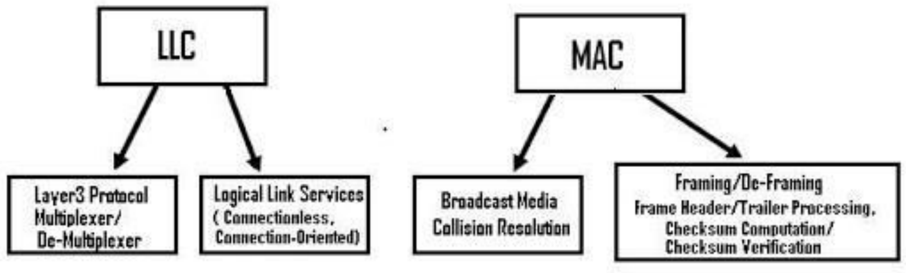
- The Internet's network layer is responsible for **moving network-layer packets** known as datagrams from one host to another.
- It handles the service requests from the transport **layer** and further forwards the service request to the data link **layer**.
- There is **only one IP protocol,** and all Internet components that have a network layer must run the IP protocol


Layer 3 (Network) transmits packets between networks and determines the best paths.
Tail
IP1
IP2
Header
Layer 4 (Transport) segments data from L5. It ensures data arrives correctly and at what rate.
Network A
Network B

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

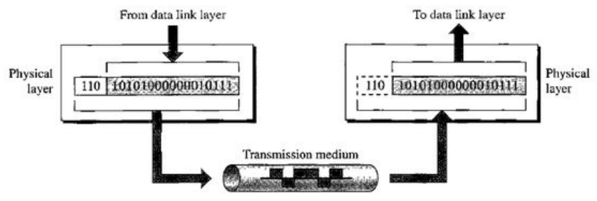
We refer to the linklayer packets as ***frames***.


LLC
MAC
Layer3 Protocol Multiplexer/ De-Multiplexer
Logical Link Services (Connectionless, Connection-Oriented)
Broadcast Media Collision Resolution
Framing/De-Framing Frame Header/Trailer Processing, Checksum Computation/ Checksum Verification

Physical Layer

The job of the physical layer is to **move the individual bits** within the frame from one node to the next.

The protocols in this layer are again link dependent and further depend on the **actual transmission medium** of the link (for example, twisted-pair copper wire, single-mode fiber optics)



The physical layer is responsible for movements of individual bits from one hop (node) to the next.

The OSI Model

- The **International Organization for Standardization (ISO)** proposed that computer networks be organized around seven layers, called the **Open Systems Interconnection (OSI)** model
- The OSI model took shape when the **protocols** that were to become the **Internet protocols**
- The functionality of five of these layers is roughly the same as their similarly named Internet counterparts.
- The two additional layers present in the OSI reference model—the **presentation layer and the session layer.**
- The role of the presentation layer is to provide services that allow communicating applications to interpret the meaning of data exchanged. These services include **data compression** and **data encryption** as well as **data decryption**

1.5.2 Encapsulation

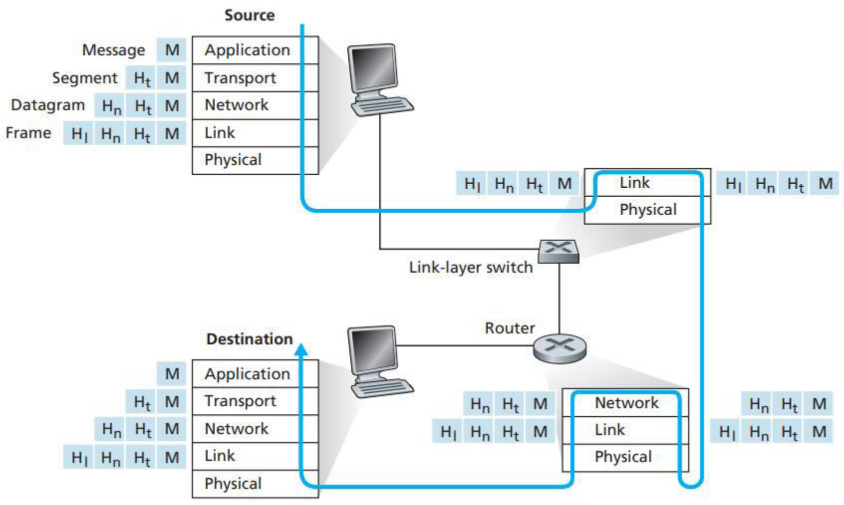


Figure 1.24 ♦ Hosts, routers, and link-layer switches; each contains a different set of layers, reflecting their differences in functionality

- Figure 1.24 shows the physical path that data takes down a sending end system's protocol stack
- Routers and link-layer switches are both packet switches
- Routers and link-layer switches do not implement all of the layers in the protocol stack; they typically implement only the bottom layers
- Figure 1.24 also illustrates the important concept of encapsulation
- The transport layer takes the message and appends additional information (so-called transport-layer header information, Ht in Figure 1.24). The transport-layer segment thus encapsulates the application-layer message
- The application-layer message and the transport-layer header information together constitute the transport-layer segment
- The transport-layer segment thus encapsulates the application-layer message.

2.1 Principles of Network Applications

- In a client-server architecture, there is an **always-on host**, called the server, which services requests from many other hosts, called clients. A classic example is the **Web application**
- **Clients do not directly communicate with each other,** two browsers do not directly communicate
- Because the server has a fixed, well-known address, and because the server is always on, a client can always contact the server by **sending a packet to the server's IP address**.
- Applications with a client-server architecture include the Web, FTP, Telnet, and e-mail
- Google has 30 to 50 **data centers distributed around the world**, which collectively handle search, YouTube, Gmail, and other services. A **data center can have hundreds of thousands of servers**, which must be powered and maintained.
- In a **P2P architecture, there is minimal (or no) reliance** on dedicated servers in data centers. 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 desktops and laptops controlled by users, with most of the peers residing in **homes, universities, and offices**.
- Because the peers communicate without passing through a dedicated server, the architecture is called **peer-to-peer**

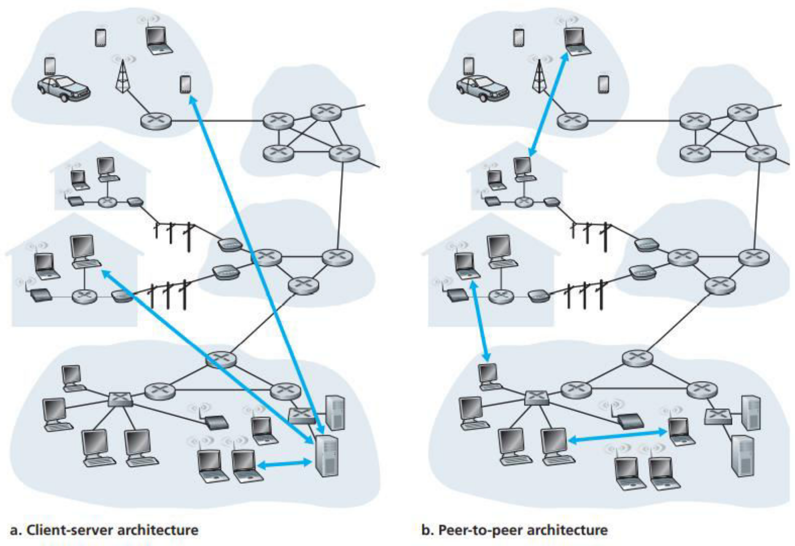


Figure 2.2 ♦ (a) Client-server architecture; (b) P2P architecture

2.1.2 Processes Communicating

- Programs, running in multiple end systems, communicate with each other.
- A process can be thought of as a **program that is running within an end system**
- When processes are running on the same end system, they can communicate with each other with inter-process communication, using rules that are governed by the end system's operating system
- Processes on two different end systems communicate with each other by **exchanging messages** across the computer network

Client and Server Processes

- With P2P file sharing, the peer that is **downloading** the file is labeled as the **client**, and the peer that is **uploading** the file is labeled as the **server**
- a process in a P2P file-sharing system can both upload and download files.
- In the context of a communication session between a pair of processes, the **process that initiates the communication** (that is, initially contacts the other process at the beginning of the session) is labeled as the **client**. The **process that waits to be contacted** to begin the session is the **server**.

The Interface Between the Process and the Computer Network

- A process sends messages into, and receives messages from, the network through a software interface called a **socket**.
- **A process is analogous to a house and its socket is analogous to its door**
- A **socket is the interface between the application layer and the transport layer** within a host. . It is also referred to as the **Application Programming Interface (API)** between the application and the network

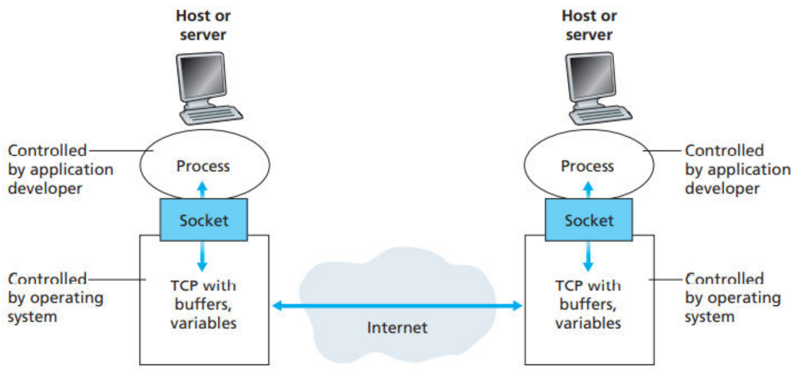

Host or server
Host or server
Controlled by application developer
Process
Process
Controlled by application developer
Socket
Socket
Controlled by operating system
TCP with buffers, variables
Internet
TCP with buffers, variables
Controlled by operating system

Figure 2.3 ♦ Application processes, sockets, and underlying transport protocol

Addressing Processes

- To identify the receiving process, two pieces of information need to be specified:
- the **address of the host** and (2) an identifier that specifies the receiving process in the **destination host**.
- In the Internet, the **host is identified by its IP address**

2.1.3 Transport Services Available to Applications

- Many networks, including the Internet, provide **more than one transport-layer protocol**.
- The situation is **similar to choosing either train or airplane** transport for travel between two cities

Unit 1 - Class 5

2.2 The Web and HTTP

- The **HyperText Transfer Protocol (HTTP)**, the Web's application-layer protocol, is at the heart of the Web
- HTTP is implemented in two programs: **a client program and a server program**.
- The client program and server program, executing on different end systems, talk to each other by **exchanging HTTP** messages.
- A Web page (also called a document) consists of **objects**. An object is simply a file—such as an **HTML file, a JPEG image, a Java applet, or a video clip**—that is addressable by a single URL
- **For example**, if a Web page contains **HTML text and five JPEG images**, then the Web page has **six objects**
- **Each URL has two components**: the **hostname** of the server that houses the object and the object's **path name**.
- **For example**, the URL http://www.someSchool.edu/someDepartment/picture.gif has **www.someSchool.edu for a hostname** and **/someDepartment/ picture.gif for a path name**
- **Figure 2.6. When a user requests a Web page** (for example, clicks on a **hyperlink**), the browser sends HTTP request messages for the objects in the page to the server. The server receives the requests and responds with HTTP response messages that contain the objects

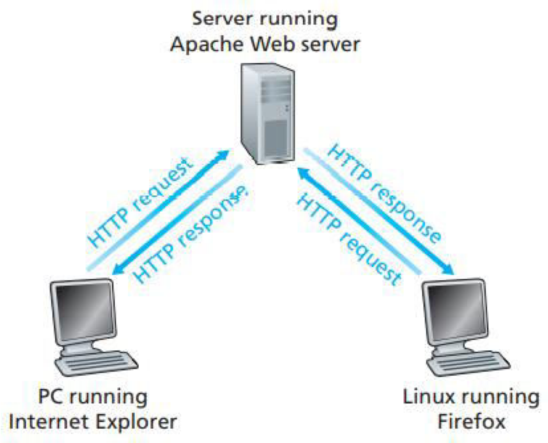
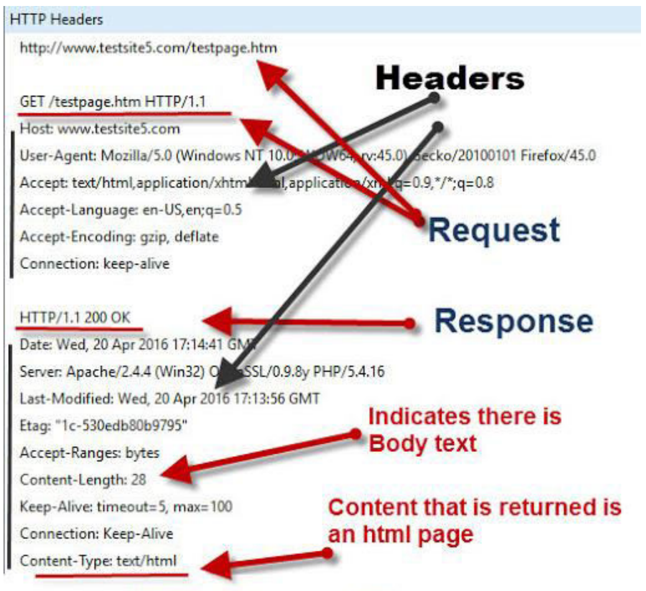


Figure 2.6 ♦ HTTP request-response behavior

- On the client side the **socket interface** is the door between the **client process and the TCP connection**; on the server side it is the door between the **server process and the TCP connection**



```
HTTP Headers
http://www.testsite5.com/testpage.htm

GET /testpage.htm HTTP/1.1
Host: www.testsite5.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive

HTTP/1.1 200 OK
Date: Wed, 20 Apr 2016 17:14:41 GMT
Server: Apache/2.4.4 (Win32) OpenSSL/0.9.8y PHP/5.4.16
Last-Modified: Wed, 20 Apr 2016 17:13:56 GMT
Etag: "1c-530edb80b9795"
Accept-Ranges: bytes
Content-Length: 28
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

Headers
Request
Response
Indicates there is Body text
Content that is returned is an html page

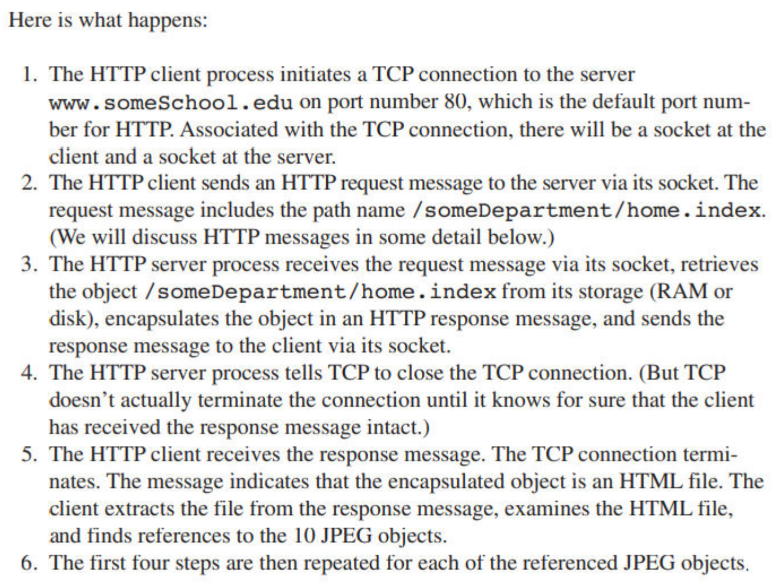
HTTP Request-Response

2.2.2 Non-Persistent and Persistent Connections

- **Non-persistent connections** - Each request/response pair be sent over a **separate TCP connection**
- **Persistent connections**- all of the requests and their corresponding responses be sent over the **same TCP connection**
- HTTP uses **persistent connections in its default mode**, HTTP clients and servers **can be configured to use non-persistent connections** instead.

HTTP with Non-Persistent Connections

- Let's suppose the page consists of a base file and 10 JPEG images, and that all 11 of these objects reside on the same server. Further suppose the URL for the base HTML file is
- http://www.someSchool.edu/someDepartment/home.index



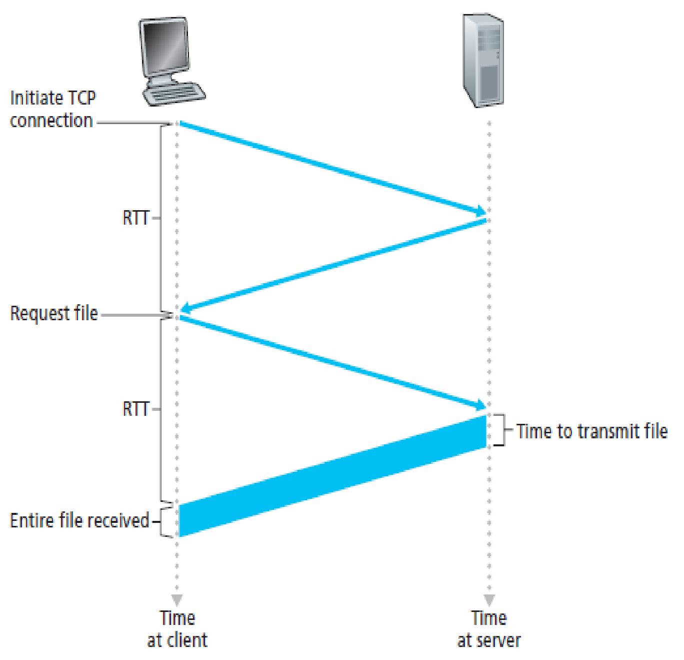
Here is what happens:

1. The HTTP client process initiates a TCP connection to the server `www.someSchool.edu` on port number 80, which is the default port number for HTTP. Associated with the TCP connection, there will be a socket at the client and a socket at the server.
2. The HTTP client sends an HTTP request message to the server via its socket. The request message includes the path name `/someDepartment/home.index`. (We will discuss HTTP messages in some detail below.)
3. The HTTP server process receives the request message via its socket, retrieves the object `/someDepartment/home.index` from its storage (RAM or disk), encapsulates the object in an HTTP response message, and sends the response message to the client via its socket.
4. The HTTP server process tells TCP to close the TCP connection. (But TCP doesn't actually terminate the connection until it knows for sure that the client has received the response message intact.)
5. The HTTP client receives the response message. The TCP connection terminates. The message indicates that the encapsulated object is an HTML file. The client extracts the file from the response message, examines the HTML file, and finds references to the 10 JPEG objects.
6. The first four steps are then repeated for each of the referenced JPEG objects.

- ➢ The steps above illustrate the use of **non-persistent connections**, where each **TCP connection is closed** after the server sends the object

Round-trip time (RTT), which is the time it takes for a small packet to travel from client to server and then back to the client.

The **RTT includes** packet-**propagation delays**, packet **queuing delays** in intermediate routers and switches, and packet-**processing delays**.



- ➢ The **total response** time is **two RTTs plus the transmission time** at the server of the HTML file

TCP connection between the browser and the Web server; this involves a **“three-way handshake”**—

- ➢ The client sends a small **TCP segment** to the server, the server **acknowledges** and responds with a small TCP segment
- ➢ The **client acknowledges back** to the server. The first two parts of the three-way handshake take **one RTT**.
- ➢ The client sends the **HTTP request message** combined with the third part of the three-way handshake (the acknowledgment) into the TCP connection.
- ➢ Once the request message arrives at the server, the **server sends the HTML file** into the TCP connection.

Non-persistent connections have some **shortcomings**.

- ➢ First, a brand-new **connection must be established and maintained** for each requested object.
- ➢ For each of these connections, **TCP buffers must be allocated** and TCP variables must be kept in both the client and server.
- ➢ This can place a **significant burden** on the Web server, which may be serving requests from hundreds of different clients simultaneously.
- ➢ Second, each object suffers a delivery **delay of two RTTs**— one RTT to establish the TCP connection and one RTT to request and receive an object.

HTTP with Persistent Connections

- ➢ The server leaves the **TCP connection open** after sending a response.
- ➢ Subsequent requests and **responses** between the same client and server can be sent over the **same connection.**
- ➢ An entire Web page (in the example above, the base HTML file and the **10 images**) can be sent over a **single persistent TCP connection**.

- Requests for objects can be made **back-to-back, without waiting** for replies to pending requests (pipelining).
- HTTP server closes a connection when it isn't used for a certain time (a configurable **timeout interval**).
- The **default** mode of HTTP uses persistent connections with **pipelining**.

2.2.3 HTTP Message Format

Below we provide a typical **HTTP request message**:

GET /somedir/page.html HTTP/1.1

Host: www.someschool.edu

Connection: close

User-agent: Mozilla/5.0

Accept-language: fr

- The message is written in ordinary ASCII text
- The message consists of five lines, each followed by a carriage return and a line feed
- The last line is followed by an additional carriage return and line feed
- The first line of an HTTP request message is called the **request line**; the subsequent lines are called the **header lines**
- The request line has three fields: the **method field, the URL field, and the HTTP version field**.
- The **method field** can take on several different values, including **GET, POST, HEAD, PUT, and DELETE**
- The **GET method** is used when the browser **requests** an object

GET /somedir/page.html HTTP/1.1	In this example, the browser is requesting the object **/somedir/page.html**. The version is self-explanatory; in this example, the browser implements version **HTTP/1.1**.
Connection: close	By including the **Connection: close** header line, the browser is telling the server that it **doesn't want to bother with persistent connections**; it wants the server to close the connection after sending the requested object
User-agent: Mozilla/5.0	The **User-agent**: header line specifies the user agent, that is, the **browser type that is making the request to the server**. Here the user agent is **Mozilla/5.0**, a Firefox browser.
Accept-language: fr	The Accept-language: header indicates that the user prefers to receive a French version of the object, if such an object exists on the server; otherwise, the server should send its default version.

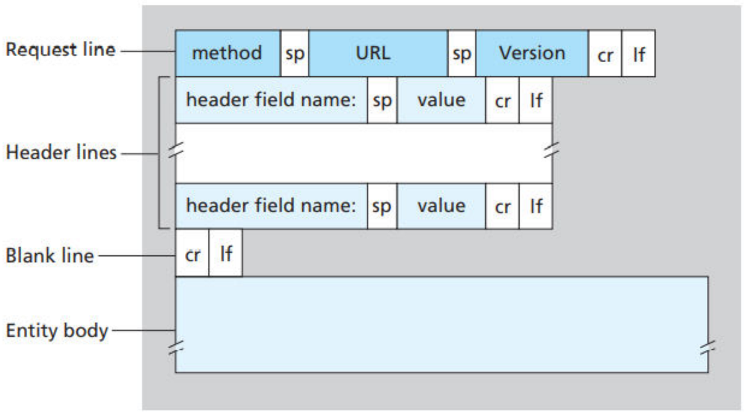


Figure 2.8 ♦ General format of an HTTP request message

CR (carriage return), single space (**SP**), line feed **(lf)**

The **HEAD method** is similar to the GET method. When a server receives a request with the HEAD method, it responds with an HTTP message but it leaves out the requested object. Application developers often use the **HEAD method** for debugging.

The **PUT method** is often used in conjunction with Web publishing tools. It allows a user to upload an object to a specific path (directory) on a specific Web server.

The **PUT method** is also used by applications that need to upload objects to Web servers.

The **DELETE method** allows a user, or an application, to delete an object on a Web server.

HTTP Response Message

Below we provide a typical HTTP response message. This response message could be the response to the example request message just discussed.

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 data ...)

It has **three sections**: an **initial status line, six header lines, and then the entity body**

HTTP/1.1 200 OK	The **status line** has **three fields**: the protocol **version field, a status code, and a corresponding status message**. In this example, the status line indicates that the server is using **HTTP/1.1** and that everything is OK. (that is, the server has found, and is sending, the requested object). **200 OK**: Request succeeded and the information is returned in the response **301 Moved Permanently**: Requested object has been permanently moved; **400 Bad Request**: This is a generic error code indicating that the request could not be understood by the server. **404 Not Found**: The requested document does not exist on this server. **505 HTTP Version Not Supported**: The requested HTTP protocol version is not supported by the server.
Connection: close	Connection: close header line to tell the **client that it is going to close the TCP connection** after sending the message
Date: Tue, 09 Aug 2011 15:44:04 GMT	The Date: header line indicates the **time and date when the HTTP response was created** and sent by the server
Server: Apache/2.2.3 (CentOS)	The Server: header line indicates that the **message was generated by an Apache Web server**; it is analogous to the User-agent: header line in the HTTP request message.
Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT	The Last-Modified: header line indicates the **time and date when the object was created or last modified.**
Content-Length: 6821	Content-Length: header line indicates the **number of bytes in the object being sent**.
Content-Type: text/html	The Content-Type: header line indicates that the **object in the entity body is HTML text**.
(data data data data data ...)	The entity body is the meat of the message—it **contains the requested object itself (represented by data data data data data ...)**

Unit 1_Class 6:

2.3 Electronic Mail in the Internet

- Electronic mail has been around since the **beginning of the Internet.** It remains one of the Internet's **most important and utilized applications**
- e-mail is an asynchronous communication medium—electronic mail is fast, easy to distribute, and inexpensive.
- Modern e-mail has many powerful features, including messages with attachments, hyperlinks, HTML-formatted text, and embedded photos

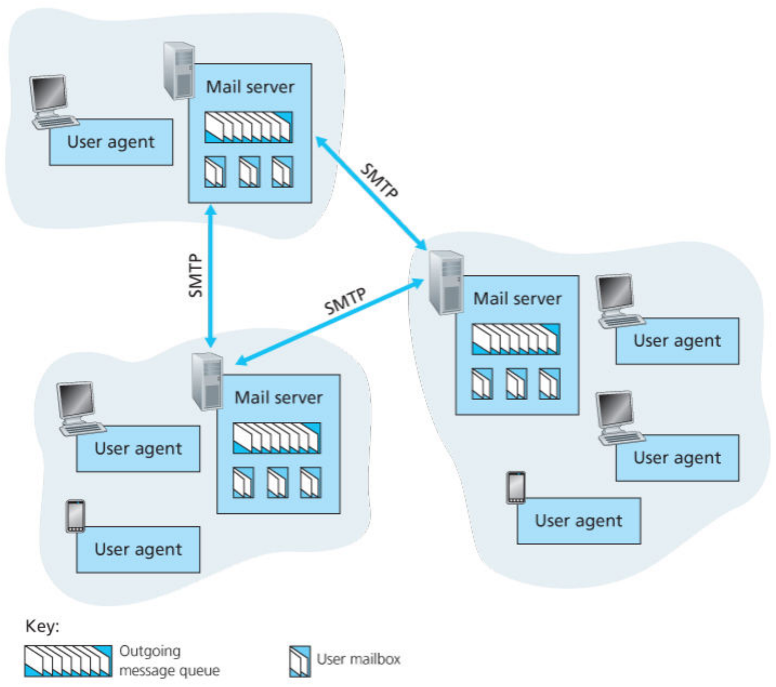


Figure 2.14 A high-level view of the Internet e-mail system

- We see from this diagram that it has three major components: **user agents, mail servers, and the Simple Mail Transfer Protocol (SMTP).**
- User agents allow users to **read, reply to, forward, save, and compose messages.**
- 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.
- Server holds the message in a message **queue** and attempts to transfer the message later.
- If there is **no success** after several days, the server removes the message and **notifies the sender with an e-mail message.**
- **SMTP** is the principal **application-layer** protocol for **Internet electronic mail**. It uses the reliable data **transfer service of TCP** to transfer mail from the sender's mail server to the recipient's mail server
- **SMTP has two sides**: a **client side,** which executes on the sender's mail server, and a **server side**, which executes on the recipient's mail server
- **When a mail server sends mail to other mail servers, it acts as an SMTP client. When a mail server receives mail from other mail servers, it acts as an SMTP server**.

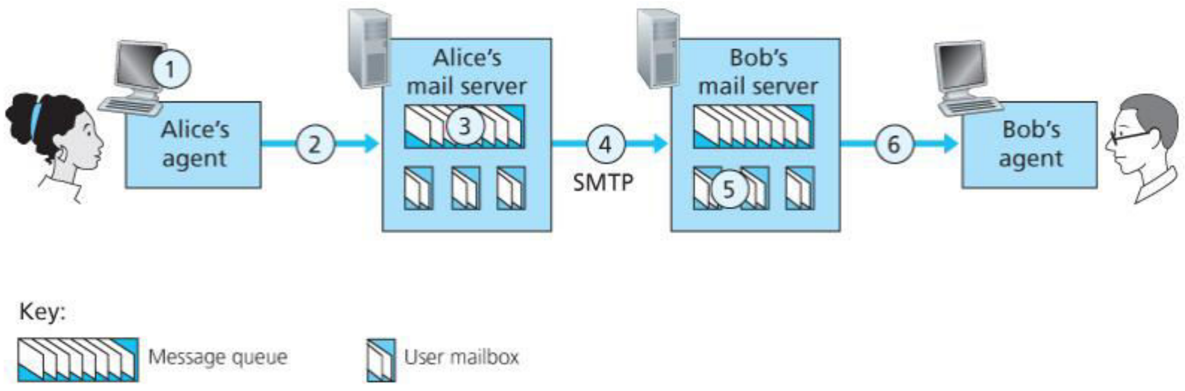


Figure 2.15 Alice sends a message to Bob

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

Example transcript of messages exchanged between an SMTP client (C) and an SMTP server (S).

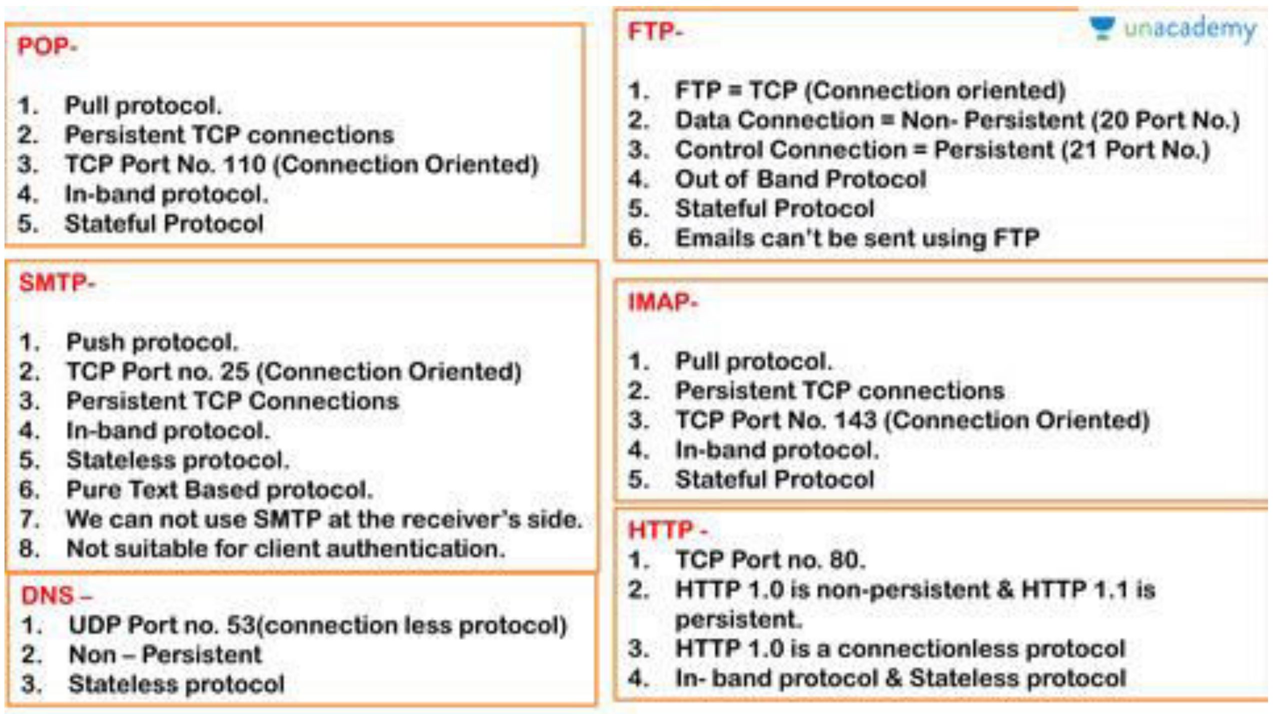
The following transcript begins as soon as the **TCP connection** is established.

S: 220 hamburger.edu
C: **HELO** crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: **MAIL FROM**:
S: 250 alice@crepes.fr ... Sender ok
C: **RCPT** TO:
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

- The hostname of the **client is crepes.fr** and the hostname of the **server is hamburger.edu**
- The client sends a message (“ **Do you like ketchup? How about pickles?** ”) from mail server crepes.fr to mail server hamburger.edu
- **CR and LF** stand for **carriage return** and **line feed**, respectively.
- The client begins the process with a new **MAIL FROM: crepes.fr** , designates the end of message with an isolated period, and issues **QUIT** only after all messages have been sent.
- The ASCII text lines prefaced with **C: are exactly the lines the client sends into its TCP socket**, and the ASCII text lines prefaced with **S: are exactly the lines the server sends into its TCP socket.**

2.3.2 SMTP Comparison with HTTP

- Both protocols are used to transfer files from one host to another
- SMTP transfers files (that is, e-mail messages) from one mail server to another mail server.
- **HTTP is mainly a pull protocol**—someone loads information on a Web server and users use HTTP to **pull the information from the server** at their convenience.
- In particular, the TCP connection is initiated by the machine that wants to receive the file.
- On the other hand, **SMTP is primarily a push protocol**—the sending mail server pushes the file to the receiving mail server.
- In particular, the TCP connection is initiated by the machine that **wants to send the file**.
- SMTP requires each message, including the body of each message, to be in 7-bit ASCII format.
- If the message contains characters that are not 7-bit ASCII (for example, French characters with accents) or contains binary data (such as an image file), then the message has to be encoded into 7-bit ASCII.
- HTTP data does not impose this restriction.


POP-

1. Pull protocol.
2. Persistent TCP connections
3. TCP Port No. 110 (Connection Oriented)
4. In-band protocol.
5. Stateful Protocol

FTP-

1. FTP = TCP (Connection oriented)
2. Data Connection = Non- Persistent (20 Port No.)
3. Control Connection = Persistent (21 Port No.)
4. Out of Band Protocol
5. Stateful Protocol
6. Emails can't be sent using FTP

SMTP-

1. Push protocol.
2. TCP Port no. 25 (Connection Oriented)
3. Persistent TCP Connections
4. In-band protocol.
5. Stateless protocol.
6. Pure Text Based protocol.
7. We can not use SMTP at the receiver's side.
8. Not suitable for client authentication.

IMAP-

1. Pull protocol.
2. Persistent TCP connections
3. TCP Port No. 143 (Connection Oriented)
4. In-band protocol.
5. Stateful Protocol

DNS –
1. UDP Port no. 53(connection less protocol)
2. Non – Persistent
3. Stateless protocol

HTTP -
1. TCP Port no. 80.
2. HTTP 1.0 is non-persistent & HTTP 1.1 is persistent.
3. HTTP 1.0 is a connectionless protocol
4. In- band protocol & Stateless protocol

POP3

- POP3, is used to transfer mail from the **recipient's mail server to the recipient's user agent**.
- POP3 is an extremely simple **mail access protocol**
- POP3 begins when the user agent (the client) opens a **TCP connection** to the mail server (the server) on **port 110**.
- POP3 progresses through three phases: **authorization, transaction, and update**
- First phase, authorization, the user agent sends a **username and a password**
- Second phase, transaction, the **user agent retrieves messages**; also during this phase, the user agent can **mark messages for deletion**, remove deletion marks, and obtain **mail statistics**.
- The third phase, update, occurs after the **client has issued the quit command**, ending the POP3 session; at this time, the mail **server deletes the messages** that were marked for deletion.
- Two possible responses: **+OK**, used by the server to indicate that the **previous command was fine**; and **-ERR**, used by the server to indicate that **something was wrong with the previous command**.

Suppose that *mailServer* is the name of your mail server. You will see something like:

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

- If you **misspell** a command, the POP3 server will reply with an -**ERR message**.

Transaction phase: A user agent using POP3 can often be configured (by the user) to **"download and delete"** or to **"download and keep."**

In the **download-and-delete mode**, the user agent will issue the **list , retr** , and **dele** commands.

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: (blah blah ... S: ................ S: ..........blah)
S: .
C: dele 1
C: retr 2
S: (blah blah ...
S: ................
S: ..........blah)
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

- The user agent first asks the mail server to list the **size of each of the stored messages**.
- The user agent then **retrieves and deletes each message from the server**.
- Note that after the authorization phase, the user agent employed only four commands: **list , retr , dele , and quit** .
- After processing the **quit command**, the POP3 server enters the update phase and **removes messages 1 and 2 from the mailbox**

Class 7
2.4 DNS—The Internet's Directory Service

- Just as humans can be identified in **many ways** (For example, we can be identified by the **names that appear on our birth certificates**), so too can Internet hosts.
- One identifier for a host is its hostname. **Hostnames**—such as www.facebook.com, www.google.com , gaia.cs.umass.edu
- A hostname such as **www.eurecom.fr** , which ends with the country code .fr , tells us that the host is probably in **France**
- hosts are also identified by **IP addresses**.

- An **IP address consists of four bytes** and has a rigid hierarchical structure.
- An IP address **looks like 121.7.106.83** , where each period separates one of the bytes expressed in decimal notation **from 0 to 255**.
- An IP address is **hierarchical** because as we scan the address from left to right, we obtain more and more specific information about **where the host is located in the Internet** (that is, within which network, in the network of networks)

2.4.1 Services Provided by DNS

- There are two ways to identify a host—by a **hostname and by an IP address**. People prefer the more mnemonic hostname identifier,
- We need a directory service that **translates hostnames to IP addresses**. This is the main task of the Internet's **domain name system (DNS).**
- The DNS protocol runs over UDP and uses **port 53**.
- DNS is commonly employed by other application-layer protocols—including **HTTP and SMTP** to translate user-supplied hostnames to IP addresses.

In order for the **user's host** to be able to send an **HTTP request message** to the Web server **www.someschool.edu** , the user's host must first **obtain the IP address of www.someschool.edu**.

This is done as follows.

1. The same user machine runs the **client side of the DNS** application.
2. The **browser extracts the hostname, www.someschool.edu , from the URL** and passes the hostname to the client side of the DNS application.
3. The **DNS client sends a query** containing the hostname to a DNS server.
4. The DNS client eventually **receives a reply**, which includes the **IP address for the hostname**.
5. Once the browser receives the IP address from DNS, it can **initiate a TCP connection** to the HTTP server process located at **port 80 at that IP address**.

2.4.2 Overview of How DNS Works

- A simple design for DNS would have one DNS server that contains all the mappings.
- In this **centralized design**, clients simply direct all queries to the **single DNS server**, and the DNS server responds directly to the **querying clients**

The problems with a centralized design include:

- **A single point of failure**: If the DNS server crashes, so does the entire Internet!
- **Traffic volume**: A single DNS server would have to **handle all DNS queries** (for all the HTTP requests and e-mail messages generated from hundreds of millions of hosts).
- **Distant centralized database**: A single DNS server cannot be **"close to" all the querying clients**. If we put the single DNS server in New York City, then all queries from Australia must travel to the other side of the globe, perhaps over **slow and congested links.** This can lead to **significant delays**.
- **Maintenance:** The single DNS server would have to keep records for all Internet hosts. Not only would this **centralized database be huge**, but it would have to be **updated frequently** to account for every new host.

  DNS is distributed by design. In fact, the DNS is a wonderful example of how a distributed database can be implemented in the Internet

A Distributed, Hierarchical Database

- The DNS uses a large number of servers, organized in a **hierarchical fashion** and distributed around the world.
- No single DNS server has all of the mappings for all of the hosts in the Internet.
- There are three classes of DNS servers—**root DNS servers**, **top-level domain (TLD) DNS servers**, and **authoritative DNS servers**

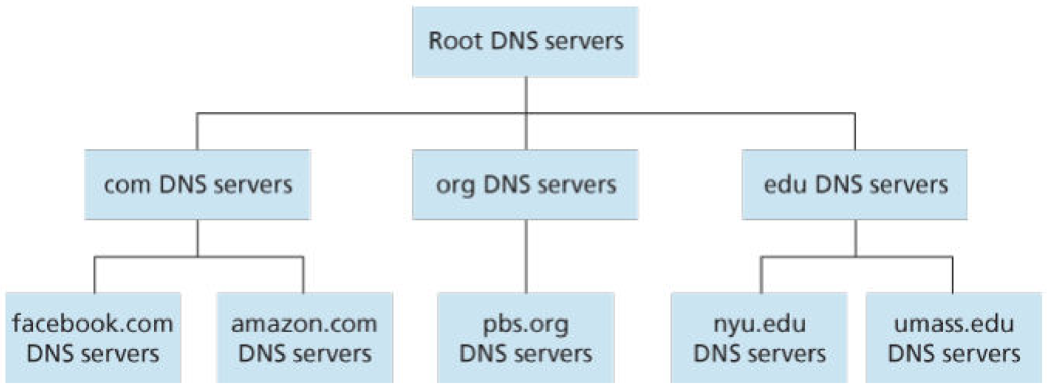


Figure 2.17 Portion of the hierarchy of DNS servers

- The client first contacts one of the **root servers**, which **returns IP addresses for TLD servers** for the top-level domain **com** .
- The client then contacts one of these TLD servers, which **returns the IP address of an authoritative server for amazon.com .**
- Finally, the client contacts one of the authoritative servers for amazon.com , which **returns the IP address for the hostname www.amazon.com .**

- **Root DNS servers:** There are over **400 root name servers** scattered all over the world. Root name servers provide the IP **addresses of the TLD servers**.
- These root name servers are managed by **13 different organizations**.
- **Top-level domain (TLD) servers**: For each of the top-level domains — top-level domains such as **com, org, net, edu, and gov**, and all of the country **top-level domains such as uk, fr, ca, and jp — there is TLD server** (or server cluster).
- **Authoritative DNS servers:** Every organization with publicly accessible hosts (such as **Web servers and mail servers**) on the Internet must provide **publicly accessible DNS records** that map the names of those hosts to IP addresses

There is another important type of DNS server called the **local DNS server**. A local DNS server **does not strictly belong to the hierarchy of servers** but is nevertheless central to the DNS architecture.

Each ISP—such as a **residential ISP or an institutional ISP**—has a local DNS server (also called a default name server). When a host connects to an ISP, the **ISP provides the host with the IP addresses** of one or more of its local DNS servers

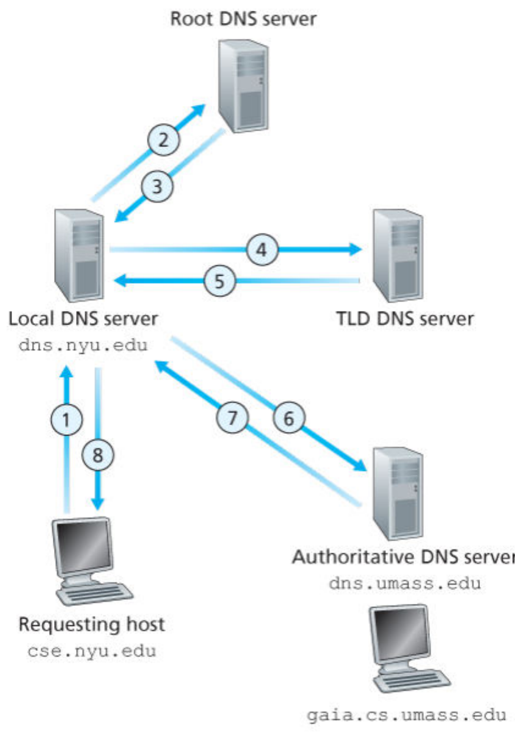
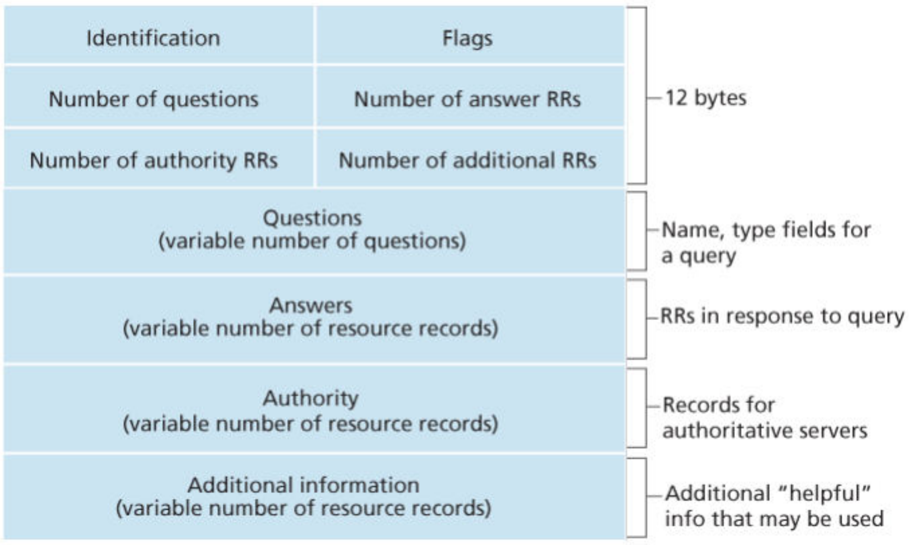


Figure 2.19 Interaction of the various DNS servers

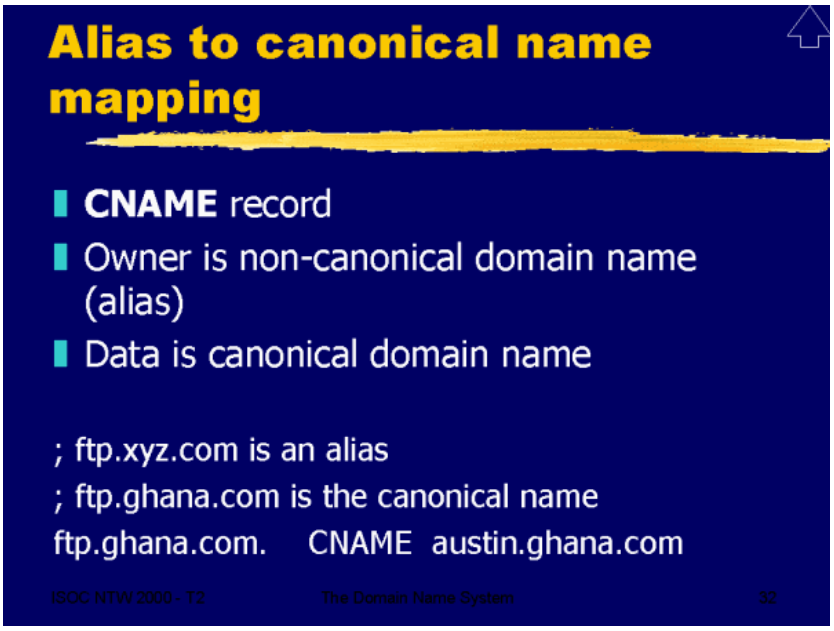
- Suppose the host **cse.nyu.edu desires the IP address of gaia.cs.umass.edu** .
- Also suppose that NYU's **local DNS server for cse.nyu.edu is called dns.nyu.edu** and that an **authoritative DNS server for gaia.cs.umass.edu is called dns.umass.edu**
- As shown in Figure 2.19, the host **cse.nyu.edu** first sends a D**NS query message to its local DNS server, dns.nyu.edu** .
- The query message contains the hostname to be translated, namely, gaia.cs.umass.edu . The **local DNS server forwards the query message to a root DNS server**.
- The root DNS server takes note of the **edu suffix** and returns to the local DNS server **a list of IP addresses for TLD servers responsible for edu** .
- The local DNS server then resends the **query message** to one of these TLD servers.
- The TLD server takes note of the umass.edu suffix and responds with the **IP address of the authoritative DNS server** for the University of Massachusetts, namely, dns.umass.edu .
- Finally, the **local DNS server resends the query message directly to dns.umass.edu** , which **responds with the IP address of gaia.cs.umass.edu** .
- Note that in this example, in order to obtain the mapping for **one hostname, eight DNS messages were sent: four query messages and four reply messages**!

DNS Messages

- **DNS query and reply messages:** These are the only two kinds of DNS messages. Both query and reply messages have the **same format**


Identification
Flags
Number of questions
Number of answer RRs
Number of authority RRs
Number of additional RRs
12 bytes
Questions (variable number of questions)
Name, type fields for a query
Answers (variable number of resource records)
RRs in response to query
Authority (variable number of resource records)
Records for authoritative servers
Additional information (variable number of resource records)
Additional "helpful" info that may be used

- The first **12 bytes is the header** section, which has a number of fields.
- The first field is a **16-bit number** that identifies the query.
- There are a number of **flags in the flag field.** A 1-bit query/reply flag indicates whether the message is a **query (0) or a reply (1).**
- A **1-bit authoritative flag** is set in a reply message
- A **1-bit recursion-desired flag** is set when a client (host or DNS server) desires that the DNS server perform **recursion** when it doesn't have the record.
- The **question section** contains information about the **query** that is being made.
- The **answer section** contains the **resource records** for the name that was originally queried.
- The **authority section** contains records of other authoritative servers.
- The **additional section** contains other helpful records

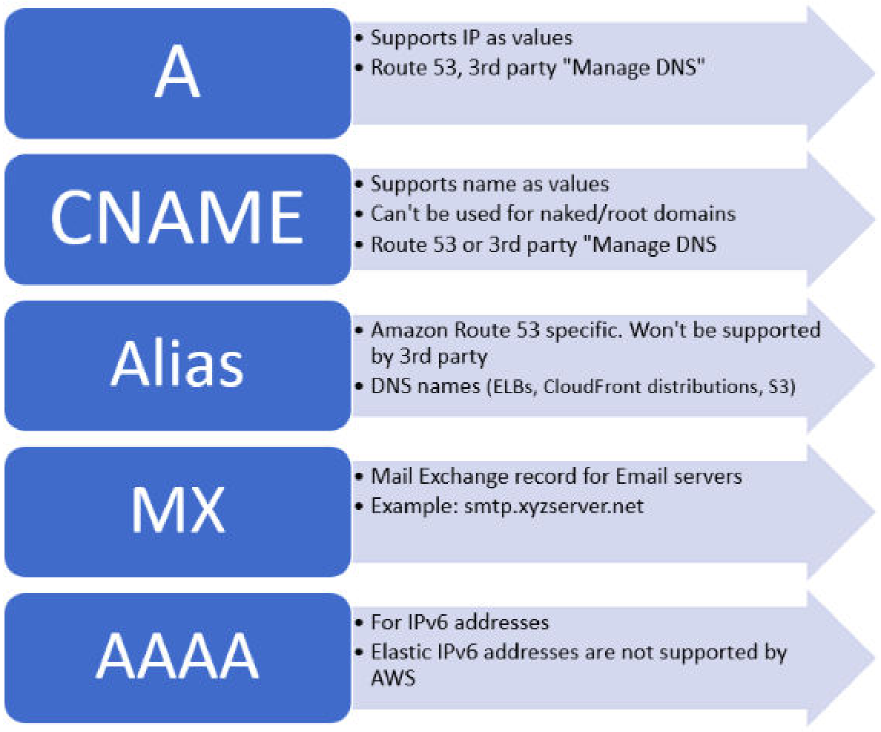


Alias to canonical name mapping

- **CNAME** record
- Owner is non-canonical domain name (alias)
- Data is canonical domain name

```
; ftp.xyz.com is an alias
; ftp.ghana.com is the canonical name
ftp.ghana.com.    CNAME  austin.ghana.com
```

ISOC NTW 2000 - T2 The Domain Name System 32



A
- Supports IP as values
- Route 53, 3rd party "Manage DNS"

CNAME
- Supports name as values
- Can't be used for naked/root domains
- Route 53 or 3rd party "Manage DNS

Alias
- Amazon Route 53 specific. Won't be supported by 3rd party
- DNS names (ELBs, CloudFront distributions, S3)

MX
- Mail Exchange record for Email servers
- Example: smtp.xyzserver.net

AAAA
- For IPv6 addresses
- Elastic IPv6 addresses are not supported by AWS

DNS records

DNS: distributed db storing resource records (RR)

RR format: `(name, value, type, ttl)`

Type=A
- **name** is hostname
- **value** is IP address

Type=NS
- **name** is domain (e.g. foo.com)
- **value** is hostname of authoritative name server for this domain

Type=CNAME
- **name** is alias name for some "canonical" (the real) name
- `www.ibm.com` is really `servereast.backup2.ibm.com`
- **value** is canonical name

Type=MX
- **value** is name of mailserver associated with **name**

Application 12

Unit 1: Class 8

Content Distribution Networks (CDNs)

- Many Internet video companies are **distributing on-demand** multi-Mbps streams to millions of users
- **YouTube**, for example, with a library of hundreds of millions of videos, distributes hundreds of millions of video streams to users around the world every day

But there are few problems with this approach.

- If the client is **far from the data center**, server-to-client packets will cross many communication links and likely pass through many ISPs, with some of the ISPs possibly located on different continents.
- If one of these links provides a **throughput** that is less than the video consumption rate, the end-to-end throughput will also be below the consumption rate, resulting in annoying freezing **delays** for the user.
- A **popular video** will likely be sent many times over the **same communication links**. Not only does this **waste network bandwidth**, but the **Internet video company itself will be paying its provider ISP** (connected to the data center) for sending the same bytes into the Internet over and over again
- This solution is that a **single data center** represents a **single point of failure**—if the data center or its links to the Internet goes down, it would not be able to distribute any video streams.

In order to meet the challenge of distributing massive amounts of video data to users distributed around the world, almost all major video-streaming companies make use of **Content Distribution Networks (CDNs).**
A **CDN manages servers in multiple geographically distributed locations**, stores **copies of the videos** (and other types of Web content, including documents, images, and audio) in its servers, and attempts to direct each user request to a CDN location that will provide the best user experience
The CDN **may be a private CDN**, that is, owned by the content provider itself; for example, **Google's** CDN distributes YouTube videos and other types of content.
The CDN may alternatively be a **third-party CDN** that distributes content on behalf of multiple content providers

When a **browser** in a user's host is instructed to retrieve a **specific video**
(1) **determine a suitable CDN server** cluster for that client at that time, and
(2) **redirect the client's request** to a server in that cluster.

Suppose a content provider, **NetCinema**, employs the **third-party CDN company, KingCDN**, to distribute its videos to its customers.

1. The user **visits** the Web page at NetCinema.

2. When the user **clicks on the link http://video.netcinema.com/6Y7B23V**, the user's host sends a **DNS query** for video.netcinema.com

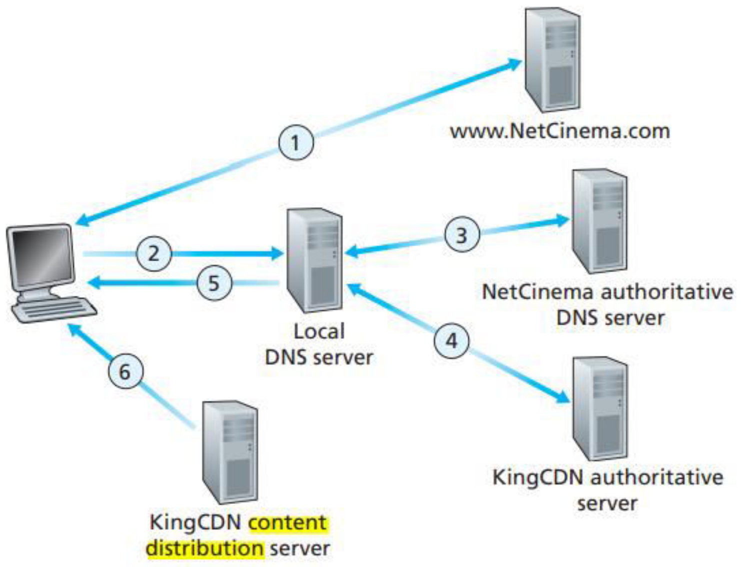

www.NetCinema.com
1
2
5
3
4
6
Local DNS server
NetCinema authoritative DNS server
KingCDN authoritative server
KingCDN content distribution server

Figure 7.4 ♦ DNS redirects a user's request to a CDN server

3. The user's **Local DNS Server (LDNS)** relays the DNS query to an authoritative DNS server for NetCinema, which observes the string "video" in the hostname video.netcinema.com.

- To "hand over" the DNS query to KingCDN, instead of returning an IP address, the NetCinema **authoritative DNS server returns to the LDNS** a hostname in the KingCDN's domain, for example, a1105.kingcdn.com.

4. From this point on, the DNS query enters into KingCDN's private DNS infrastructure. The user's LDNS then sends a second query, now for a1105.kingcdn.com, and KingCDN's DNS system eventually returns the IP addresses of a KingCDN content server to the LDNS. It is thus here, within the KingCDN's DNS system, that the CDN server from which the client will receive its content is specified.

5. The **LDNS forwards the IP address** of the content-serving CDN node to the user's host.

6. Once the client receives the IP address for a KingCDN content server, it establishes a **direct TCP connection with the server** at that IP address and issues an HTTP GET request for the video. If DASH is used, the server will first send to the client a manifest file with a list of URLs, one for each version of the video, and the client will dynamically select chunks from the different versions.

Cluster Selection Strategies

At the core of any CDN deployment is a cluster selection strategy, that is, a mechanism for **dynamically directing clients** to a server cluster or a data center within the CDN

One simple strategy is to assign the client to the cluster that is **geographically closest**

When a DNS request is received from a particular LDNS, the CDN chooses the **geographically closest cluster**

In order to **determine the best cluster** for a client based on the current **traffic** conditions, CDNs can instead perform **periodic real-time measurements of delay** and loss performance between their clusters and clients.

Case Studies: Netflix, YouTube, and Kankan

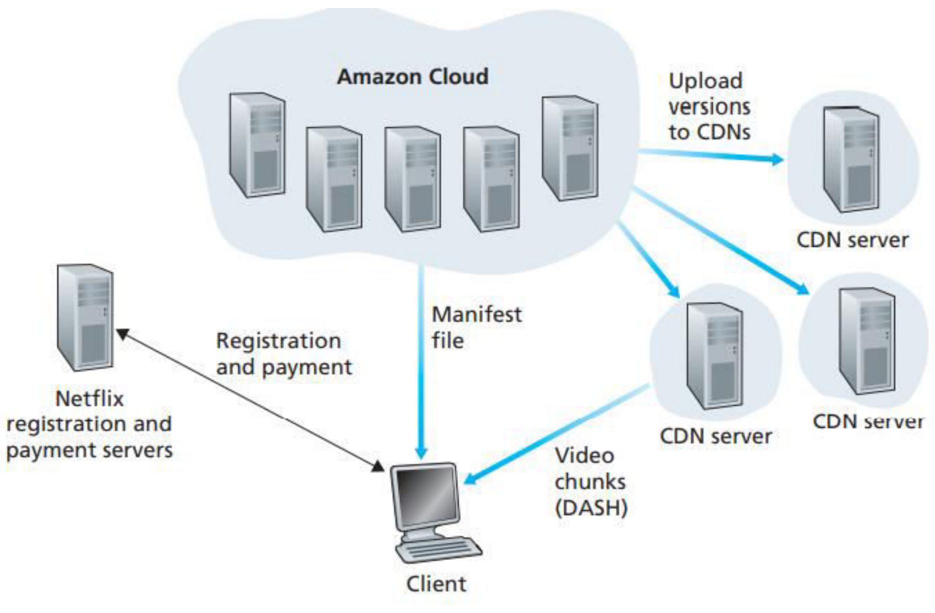


Figure 7.6 ♦ Netflix video streaming platform

Some of the functions taking place in the Amazon cloud include:

- ***Content ingestion***: Before Netflix can distribute a movie to its customers, it must first ingest and process the movie. Netflix receives studio master versions of movies and **uploads them to hosts in the Amazon cloud.**
- **Content processing:** The machines in the Amazon cloud create many different formats for each movie, suitable for a diverse array of client video players **running on desktop computers, smartphones, and game consoles connected to televisions.** A different version is created for each of these formats and at **multiple bit rates**, allowing for adaptive streaming over HTTP using DASH.
- **Uploading versions to the CDNs:** Once all of the versions of a movie have been created, the hosts in the Amazon cloud upload the versions to the CDNs

GOOGLE'S NETWORK INFRASTRUCTURE

Google has deployed an **extensive private network** and CDN infrastructure. Google's CDN infrastructure has **three tiers** of server clusters:

Fourteen "**mega data centers**," with eight in **North America**, four in **Europe**, and two in **Asia** [Google Locations 2016], with each data center having on the order of 100,000 servers. These mega data centers are responsible for **serving dynamic** (and often personalized) **content**, including search results and gmail messages.

An estimated **50 clusters** in IXPs scattered throughout the world, with each cluster consisting on the order of **100–500 servers** [Adhikari 2011a]. The cluster locations are distributed around the world, with each **location typically near multiple tier-1 ISP PoPs**.

These clusters are responsible for serving **static content, including YouTube videos** [Adhikari 2011a].

Many hundreds of "**enter-deep" clusters** (see discussion in 7.2.4), with each cluster located within an access ISP. Here a cluster typically consists of tens of servers within a single rack. These enter-deep servers perform **TCP splitting** (see Section 3.7) and serve static content [Chen 2011], including the static portions of Web pages that embody search results.

Bit torrent:

- BitTorrent is a popular **P2P protocol** for file distribution
- **P2P video streaming** is very similar to BitTorrent file downloading
- When a peer wants to see a video, it **contacts a tracker** to discover other peers in the system that have a **copy of that video**.
- This p**eer then requests chunks of the video** file in parallel from these **other peers that have the file**
- In BitTorrent lingo, the collection of all peers participating in the distribution of a particular file is called a **torrent**. Peers in a torrent **download equal-size chunks** of the file from one another, with a typical chunk size of 256 Kbytes
- When a peer first joins a torrent, it **has no chunks**.
- Over time it **accumulates more and more chunks**. While it **downloads chunks it also uploads chunks** to other peers.
- Once a peer has acquired the entire file, it may (selfishly) **leave** the torrent, or (altruistically) **remain** in the torrent and continue to upload chunks to other peers.
- Also, any peer may leave the torrent at any time with only a subset of chunks, and later **rejoin** the torrent

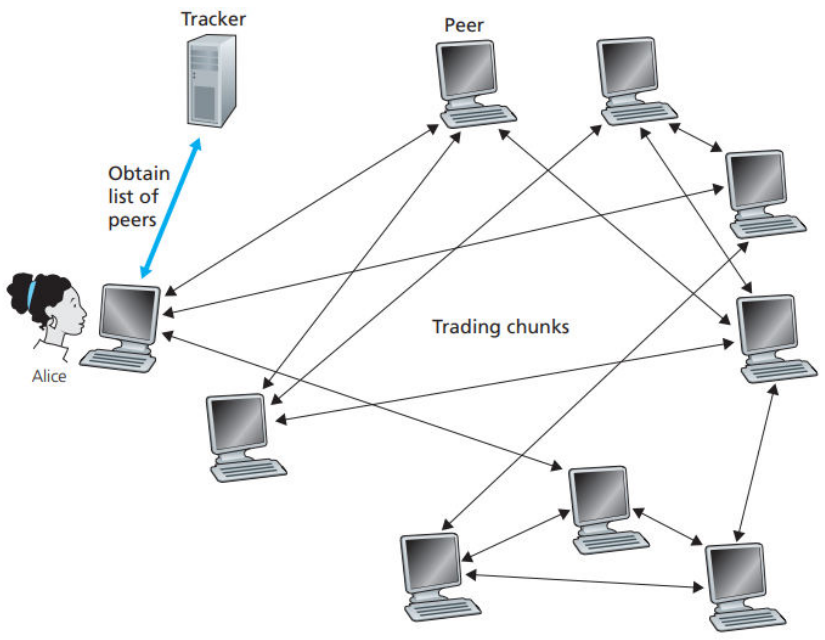


Figure 2.26 ♦ File distribution with BitTorrent

Periodically, Alice will ask each of her **neighboring peers** (over the TCP connections) for the list of the chunks they have.

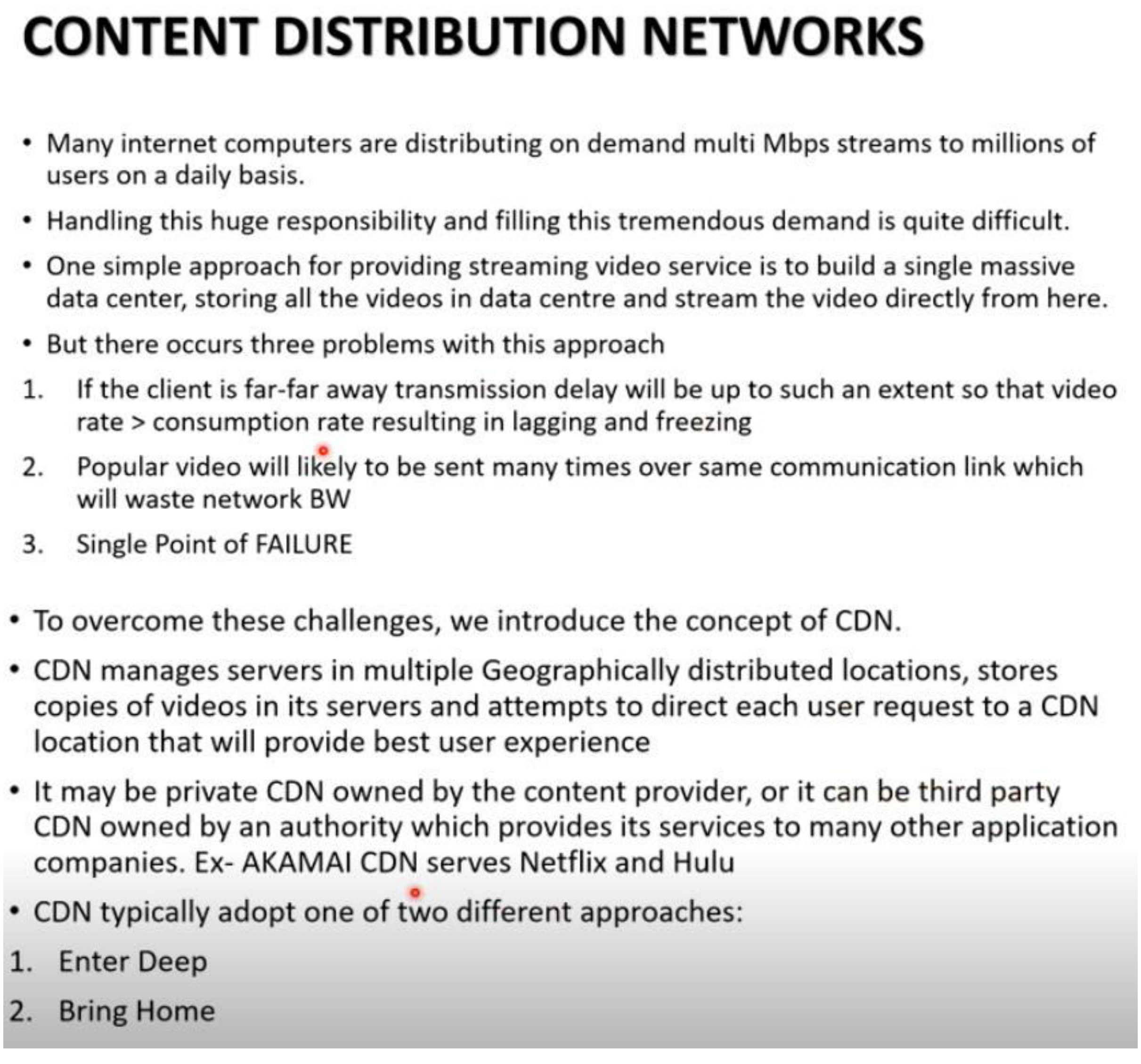
If Alice has **L different neighbors**, she will obtain L lists of chunks. With this knowledge, Alice will issue **requests** (again over the TCP connections) for chunks she currently does not have.

In deciding which chunks to request, Alice uses a technique called **rarest first**

The idea is to determine, from among the chunks she does not have, the chunks that are the **rarest among her neighbors** (that is, the chunks that have the **fewest repeated copies** among her neighbors) and then request those rarest chunks first. In this manner, the rarest chunks get more quickly redistributed, aiming to (roughly) equalize the numbers of copies of each chunk in the torrent

The basic idea is that Alice gives priority to the neighbors that are currently supplying her data at the highest rate

BitTorrent has a **number of interesting mechanisms** that are not discussed here, including pieces (**mini-chunks**), **pipelining**, **random first selection**, **endgame mode, and anti-snubbing**



CONTENT DISTRIBUTION NETWORKS

- Many internet computers are distributing on demand multi Mbps streams to millions of users on a daily basis.
- Handling this huge responsibility and filling this tremendous demand is quite difficult.
- One simple approach for providing streaming video service is to build a single massive data center, storing all the videos in data centre and stream the video directly from here.
- But there occurs three problems with this approach

1. If the client is far-far away transmission delay will be up to such an extent so that video rate > consumption rate resulting in lagging and freezing
2. Popular video will likely to be sent many times over same communication link which will waste network BW
3. Single Point of FAILURE

- To overcome these challenges, we introduce the concept of CDN.
- CDN manages servers in multiple Geographically distributed locations, stores copies of videos in its servers and attempts to direct each user request to a CDN location that will provide best user experience
- It may be private CDN owned by the content provider, or it can be third party CDN owned by an authority which provides its services to many other application companies. Ex- AKAMAI CDN serves Netflix and Hulu
- CDN typically adopt one of two different approaches:

1. Enter Deep
2. Bring Home

ENTER DEEP:

- This approach aims to enter deep in network and deploy several clusters in access ISPs all over the world.
- Goal is to get close to end user.(Making the delay less)
- As it is distributed, maintenance and managing of clusters is difficult.

BRING HOME:

- Here it brings the ISP home by building large clusters at a smaller number of key location using a private high speed network.
- Instead of Access ISPs these CDNs are typically placed near the POPs of many tier 1 ISPs.

ADVANTAGES

- Less maintenance and lower throughput to end users.
- Once clusters are installed, CDN replicates content to clusters.
- Not necessary to place video in each clusters

Some CDN use simply pull strategy, if a client requests the video and stores a video locally while streaming the video to the client at the same time .

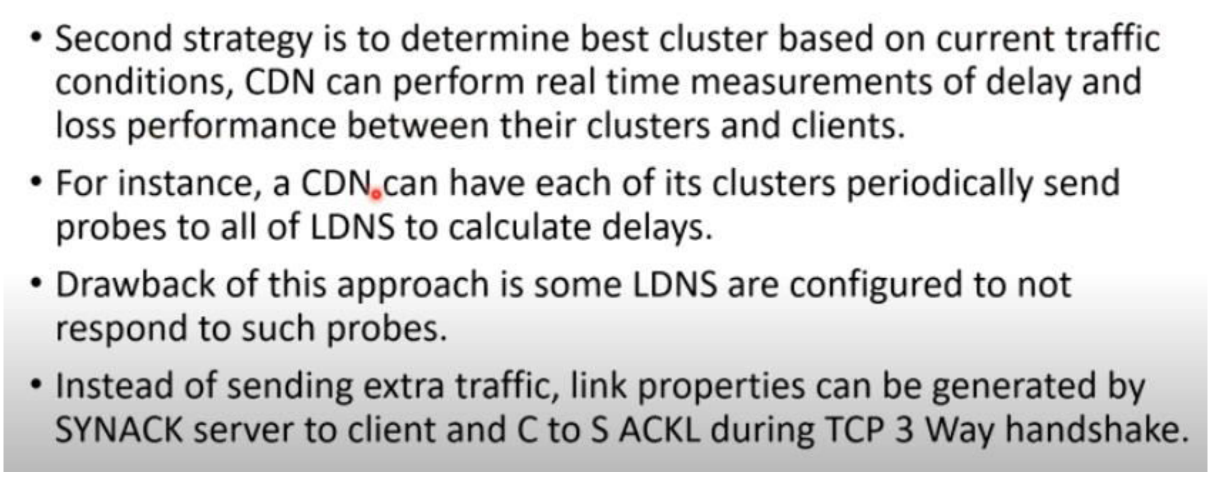
CDN operations

- When browser requests the video, CDN must intercept the request so that it can

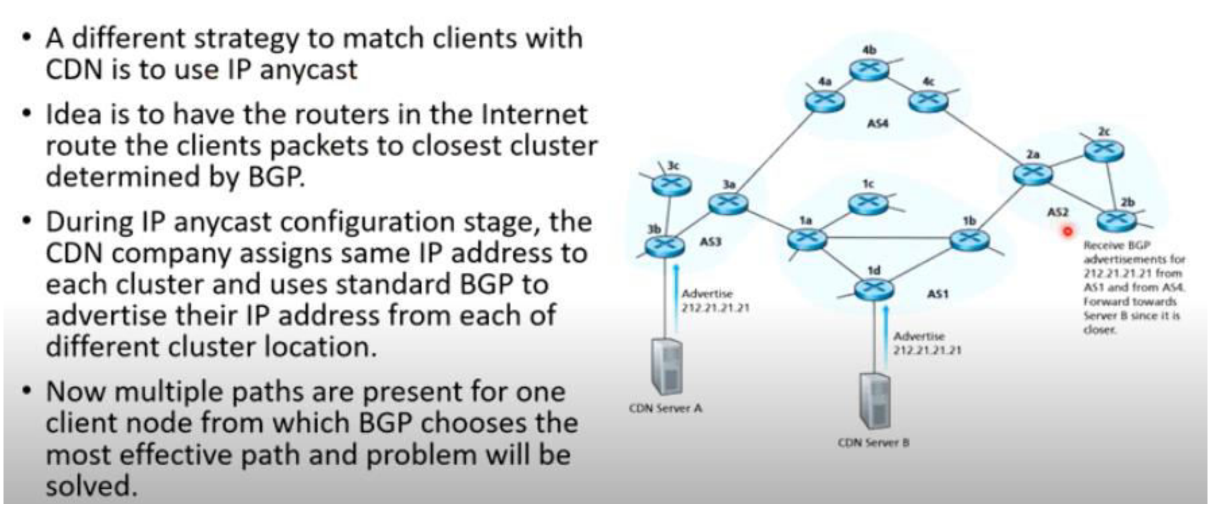
1. Determine a suitable CDN server cluster for that client
2. Redirect the client's request to a server in that cluster

CLUSTER SELECTION STRATEGIES

- CDN generally employ cluster selection strategies
- It uses different Approaches for selecting clusters
- One simple strategy is to assign cluster which is geographically closest.
- This approach is good for some clients but there may be a possibility that geographically closest cluster may not be closest in terms of network path.
- Also some end users are far from LDNS and hence CDN will track the location of DNS and provide cluster according to location of LDNS which is not so efficient.

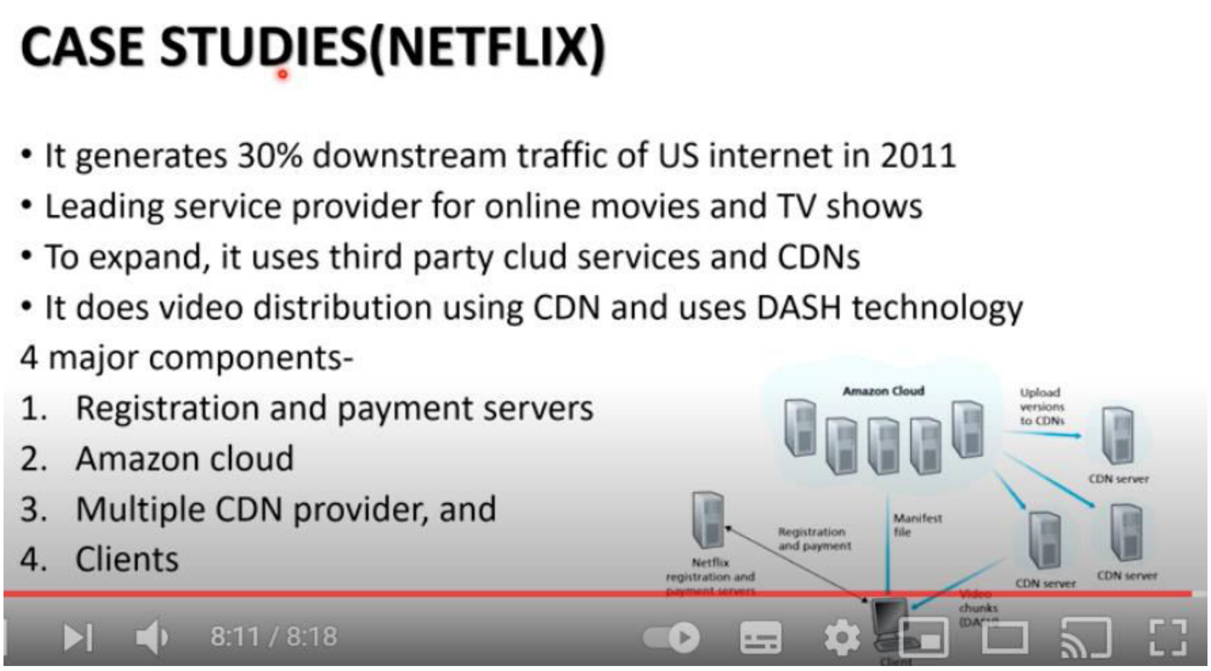


- Second strategy is to determine best cluster based on current traffic conditions, CDN can perform real time measurements of delay and loss performance between their clusters and clients.
- For instance, a CDN can have each of its clusters periodically send probes to all of LDNS to calculate delays.
- Drawback of this approach is some LDNS are configured to not respond to such probes.
- Instead of sending extra traffic, link properties can be generated by SYNACK server to client and C to S ACKL during TCP 3 Way handshake.



- A different strategy to match clients with CDN is to use IP anycast
- Idea is to have the routers in the Internet route the clients packets to closest cluster determined by BGP.
- During IP anycast configuration stage, the CDN company assigns same IP address to each cluster and uses standard BGP to advertise their IP address from each of different cluster location.
- Now multiple paths are present for one client node from which BGP chooses the most effective path and problem will be solved.

4b
4a
4c
AS4
3c
3a
3b
AS3
1c
1a
1b
1d
AS1
2a
2c
2b
AS2
Advertise 212.21.21.21
Advertise 212.21.21.21
CDN Server A
CDN Server B
Receive BGP advertisements for 212.21.21.21 from AS1 and from AS4. Forward towards Server B since it is closer.



CASE STUDIES(NETFLIX)

- It generates 30% downstream traffic of US internet in 2011
- Leading service provider for online movies and TV shows
- To expand, it uses third party clud services and CDNs
- It does video distribution using CDN and uses DASH technology

4 major components-

1. Registration and payment servers
2. Amazon cloud
3. Multiple CDN provider, and
4. Clients

Peer-to-Peer Applications

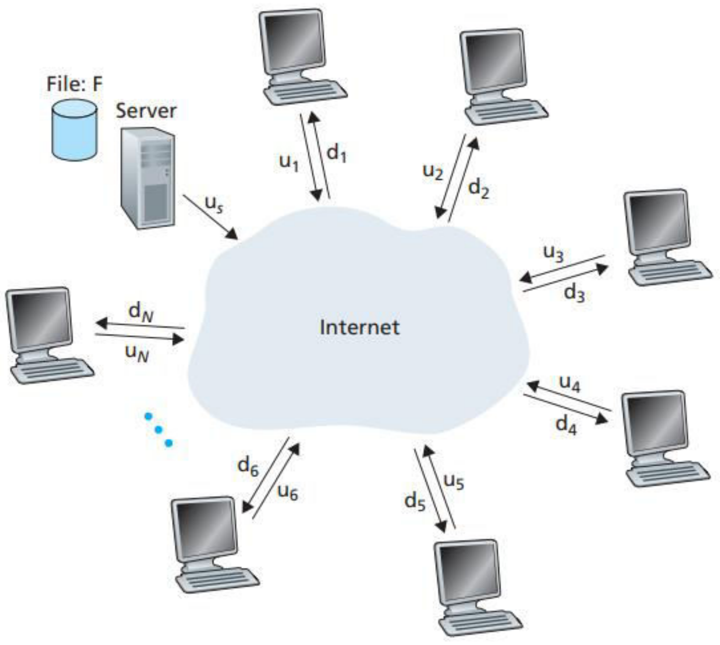


Figure 2.24 ♦ An illustrative file distribution problem

- ➢ The server and the peers are connected to the Internet with access links.
- ➢ Denote the upload rate of the **server's access link by u_s** , the upload rate of the ith peer's access link by **u_i** , and
- ➢ The download rate of the ith peer's access link by **d_i** .
- ➢ Also denote the size of the file to be distributed (in bits) by **F** and the number of peers that want to obtain a copy of the file by **N**.
- ➢ The **distribution time** is the time it takes to get a copy of the file to all N peers.

  Let's first determine the **distribution time for the client-server architecture**, which we denote by **D_{cs}**.

  - The server must transmit one copy of the file to each of the N peers. Thus the server must transmit NF bits. Since the server's upload rate is u_s, the time to distribute the file must be at least NF/u_s.
  - Let $d_{\min}$ denote the download rate of the peer with the lowest download rate, that is, $d_{\min} = \min\{d_1, d_p, \ldots, d_N\}$. The peer with the lowest download rate cannot obtain all F bits of the file in less than $F/d_{\min}$ seconds. Thus the minimum distribution time is at least $F/d_{\min}$.

  Putting these two observations together, we obtain

$$D_{cs} \geq \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\}.$$

For N large enough, the client-server distribution time is given by **NF/u_s** . Thus, the distribution time increases linearly with the number of peers **N**.

Let's now go through a **similar analysis** for the **P2P architecture**, where each peer can assist the server in distributing the file. In particular, when a peer receives some file data, it can use its own upload capacity to **redistribute the data to other peers**.

Calculating the distribution time for the P2P architecture is somewhat more complicated than for the client-server architecture, since the distrib**ution time depends on how each peer distributes portions of the file to the other peers**. Nevertheless, a simple expression for the minimal distribution time can be obtained

- At the beginning of the distribution, only the server has the file. To get this file into the community of peers, the server must send each bit of the file at least once into its access link. Thus, the minimum distribution time is at least F/u_s. (Unlike the client-server scheme, a bit sent once by the server may not have to be sent by the server again, as the peers may redistribute the bit among themselves.)
- As with the client-server architecture, the peer with the lowest download rate cannot obtain all F bits of the file in less than F/d_{min} seconds. Thus the minimum distribution time is at least F/d_{min}.
- Finally, observe that the total upload capacity of the system as a whole is equal to the upload rate of the server plus the upload rates of each of the individual peers, that is, $u_{\text{total}} = u_s + u_1 + \ldots + u_N$. The system must deliver (upload) F bits to each of the N peers, thus delivering a total of NF bits. This cannot be done at a rate faster than u_{total}. Thus, the minimum distribution time is also at least $NF/(u_s + u_1 + \ldots + u_N)$.

Putting these three observations together, we obtain the minimum distribution time for P2P, denoted by D_{P2P}.

$$D_{\text{P2P}} \geq \max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\} \tag{2.2}$$

Equation 2.2 provides a lower bound for the minimum distribution time for the P2P architecture. It turns out that if we imagine that each peer can redistribute a bit as soon as it receives the bit, then there is a redistribution scheme that actually achieves this lower bound [Kumar 2006]. (We will prove a special case of this result in the homework.) In reality, where chunks of the file are redistributed rather than individual bits, Equation 2.2 serves as a good approximation of the actual minimum distribution time. Thus, let's take the lower bound provided by Equation 2.2 as the actual minimum distribution time, that is,

$$D_{\text{P2P}} = \max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\} \tag{2.3}$$

Figure 2.25 compares the minimum distribution time for the client-server and P2P architectures assuming that all peers have the same upload rate u. In Figure 2.25, we have set $F/u = 1$ hour, $u_s = 10u$, and $d_{\min} \geq u_s$. Thus, a peer can transmit the entire file in one hour, the server transmission rate is 10 times the peer upload rate, and (for simplicity) the peer download rates are set large enough so as not to have an effect. We see from Figure 2.25 that for the client-server architecture, the distribution time increases linearly and without bound as the number of peers increases. However, for the P2P architecture, the minimal distribution time is not only always less than the distribution time of the client-server architecture; it is also less than one hour for *any* number of peers N. Thus, applications with the P2P architecture can be self-scaling. This scalability is a direct consequence of peers being redistributors as well as consumers of bits.

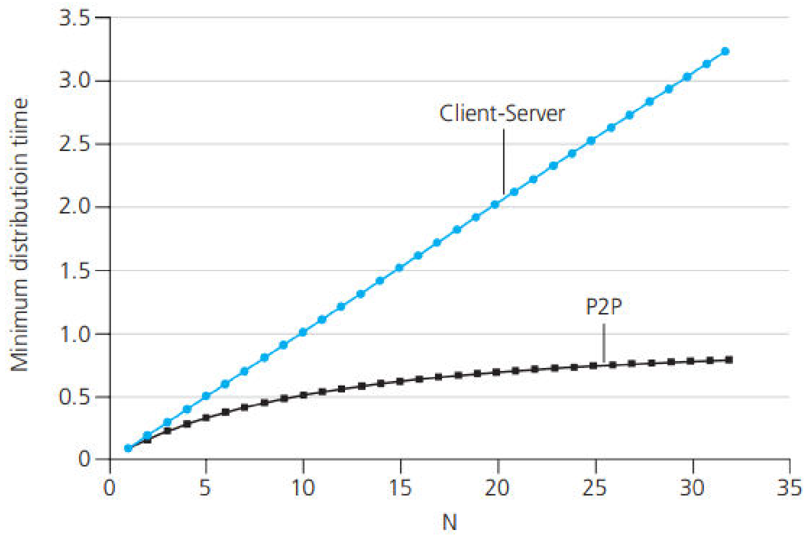


Figure 2.25 ♦ Distribution time for P2P and client-server architectures

Unit 1: Class 9

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.
Figure 2.11, a user's browser can be configured so that **all of the user's HTTP requests are first directed to the Web cache**

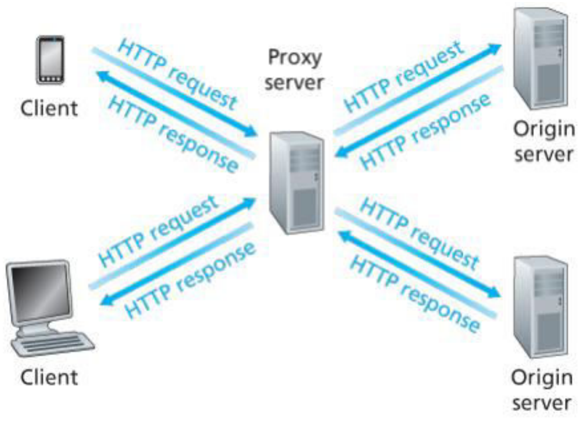


Figure 2.11 Clients requesting objects through a Web cache

As an example, suppose a browser is **requesting** the object http://www.someschool.edu/campus.gif . Here is what happens:

1. The **browser establishes a TCP connection** to the Web cache and sends an **HTTP request** for the object to the Web cache.
2. The Web cache checks to see if it has a **copy of the object stored locally**. If it does, the Web cache returns the object within an HTTP response message to the client browser.
3. If the Web cache does not have the object, the Web cache **opens a TCP connection to the origin server**, that is, to www.someschool.edu . The Web cache then sends an **HTTP request** for the object into the cache-to-server TCP connection. After receiving this request, the origin server sends the object within an **HTTP response** to the Web cache.
4. When the Web cache receives the object, it stores a copy in its **local storage** and sends a copy, within an **HTTP response message**, to the client browser (over the existing TCP connection between the client browser and the Web cache).

- Note that a **cache is both a server and a client** at the same time.
- When it **receives requests from and sends responses** to a browser, it is a **server**.
- When it **sends requests to and receives responses from an origin** server, it is a **client**

- Typically a Web cache is **purchased and installed by an ISP**.
- For example, a university might install a **cache on its campus network** and configure all of the campus browsers to point to the cache

- Web caches can substantially **reduce traffic** on an institution's access link to the Internet.
- By reducing traffic, the institution (for example, a company or a university) does not have to upgrade bandwidth as quickly, thereby **reducing costs.**

Numerical 1: Consider a 3 Mbps link being shared by 10 users.

- **Suppose we want to achieve maximum throughput using circuit switching. What should be the maximum data packet size for a user assuming 10% guard interval? Assuming a user wants to transmit 64000 bits of data how much time will he take to complete full data transmission?**

  Solution:

Throughput is maximized so link rate is divided equally among the users.

3 Mbps / 10

So per user gets (R) 0.3Mbps.

Slot time = 1/10 = 0.1s

Transmission delay (d_t) = 0.09 sec (i.e., excluding the 10% of the slot time).

So the maximum packet size L = dt * R =0.3M * 0.09 = 27 kb

To transfer 64000 bits of data, the user spends

64kb / 27kb = 2.37 sec

Numerical 1 (contd.): Consider a 3 Mbps link being shared by 10 users.

- **Suppose packet switching is used. What is the maximum achievable throughput? Assuming every user is active only 10% of the time and transmits at a rate of 1Mbps. What is the probability of no queuing?**
- Solution:

Maximum achievable throughput is 3Mbps. However, user only transmits at the rate of 1Mbps.

Probability of one user being active (p) is 0.1.

As long as there are less than or equal to 3 active users, the rate of transmission will not exceed the link rate so no queuing occurs.

$$P(no\ queuing) = P(\leq 3\ users\ are\ active)$$
$$= P(0\ users\ active) + P(1\ user\ active) + P(2\ users\ active) + P(3\ users\ active)$$

$$P(no\ queuing) = (1-p)^{10} + 10C_1 p(1-p)^9 + 10C_2 p^2(1-p)^8 + 10C_3 p^3(1-p)^7$$
$$= 0.9872$$

- **Numerical 2: Suppose there is a 10 Mbps microwave link between a geostationary satellite and its base station on Earth. Every minute the satellite takes a digital photo and sends it to the base station. Assume a propagation speed of 2.4×10^8 m/s.**

  What is the propagation delay of the link?

  What is the bandwidth-delay product, R · d_{prop}?

Let x denote the size of the photo. What is the minimum value of x for the microwave link to be continuously transmitting?

Solution:

Note: A **geostationary satellite** is in an orbit that can only be achieved at an **altitude** very close to 35,786 km (22,236 miles)

Propagation delay is (36000 km)/(2.4×108 m/s) = 150 ms

Bandwidth-delay product 1500 kb

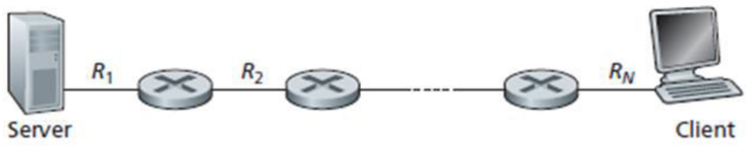
Time between photo transmission is 60s therefore, transmit **60sx10MBps = 600 Mb**

- **<u>Numerical 3:</u> Consider the figure below where transmission delay is the only significant delay. Each link is 2Mbps. Suppose the number of links N is 3. Calculate the end to end delay for the two cases given below. Note that each switch is a store and forward switch.**

  1. **If message of size 8 Mb is transferred without segmentation.**
  2. **If the message is segmented into 800 packets of 10 kb length.**

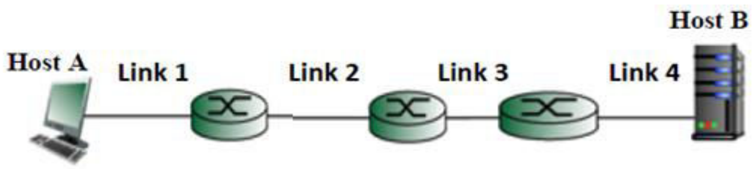
  <u>Solution:</u>

  1. L = 8 Mb, End to end delay = **3 × L/R = 3x8Mb/2Mbps = 12 sec**
  2. L = 10 kb, End to end delay = **800 × 3 × L/R = 800x3x10kb/2Mbps = 12 sec**



- **<u>Numerical 4:</u>**

  Calculate the time taken in transmission of 20,000 bits from host A to B. The data is divided into 5 packets of 4000 bits each. All four links have an identical rate of 2 Mbps and 10 Km long. Assume optical link & no processing or queuing delays. (Note-Speed of light in optical fiber is approximately 70% the speed of light).



<u>Solution:</u>

- Speed of light in optical fiber is approximately 70% the speed of light $c_o = 2 \times 10^8 m/s$
- Propagation delay per link is given by $d_p = \frac{d}{c_o} = 10 \times 10^3 / 2 \times 10^8$ = 50μs
- Transmission time for all the bits at host A is $d_t = \frac{L}{R} = 4000 / 2 \times 10^6$ = 2ms
- At $4 \times \frac{L}{R}$ the 5th packet starts transmission at Host A, at $5 \times \frac{L}{R} + d_p$ the 5th packet is at R1, at $6 \times \frac{L}{R} + 2d_p$ the 5th packet is at R2, at $7 \times \frac{L}{R} + 3d_p$, the 5th packet is at R3 and at $8 \times \frac{L}{R} + 4d_p$, the 5th packet is at Host B.
- Therefore, the total time taken is $8 \times \frac{L}{R} + 4d_p = 8 \times 4000 / 2 \times 10^6 + (4 \times 50μs) =$ 16.2ms

Numerical:

Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that n DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of $RTT_1, . . ., RTT_n$. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the object. Assuming zero transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?

Solution:

Total elapsed time = Delay accessing n DNS servers recursively + RTT due to TCP connection establishment + RTT due to HTTP request-response

$= \Sigma_{i=1}^{n} RTT_i + 2\ RTT_0$

Numerical 1 (Cont): Suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with

1. **Non-persistent HTTP with no parallel TCP connections?**
2. **Non-persistent HTTP with the browser configured for 5 parallel connections?**
3. **Persistent HTTP (no parallel connections)?**

Solution:

1. Non-persistent with no parallel connections : $\Sigma_{i=1}^{n} RTT_i + 2\ RTT_0 + 16\ RTT_0$

The first two terms as before. The 3rd term is equal to No. of objects * (RTT due to TCP handshake + RTT due to HTTP request-response)

2. Non-persistent with 5 parallel connections : $\Sigma_{i=1}^{n} RTT_i + 2\ RTT_0 + 4\ RTT_0$

The first two terms as before. First 5 out of the 8 objects can be downloaded in parallel and then the remaining 3 objects can be downloaded in parallel. Each object is downloaded after establishing a separate TCP connection. Hence, we get the 3rd term 4 RTT_0

3. Persistent with no parallel connections : $\Sigma_{i=1}^{n} RTT_i + 2\ RTT_0 + 8\ RTT_0$

The first two terms as before. The TCP connection has been established for the HTML file already. As we are dealing with persistent connection, we request each object on the existing TCP connection. Hence the 3rd term is simply No. of objects multiplied with the RTT due to HTTP request-response.

Numerical:

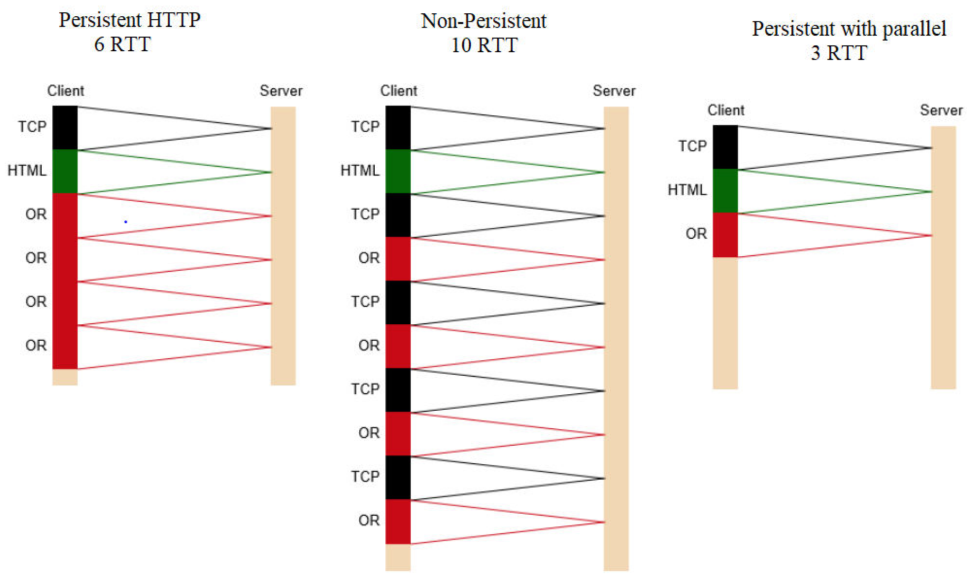
Draw a simple timing diagram indicating the delays involved in retrieving a web page containing the base HTML object and 4 additional images using

(i) Persistent HTTP

(ii) Non-persistent HTTP and

(iii) Persistent HTTP with three parallel connections. Assume size of all objects to be negligible. Express total delay in terms of RTT.

Solution:


Persistent HTTP
6 RTT
Client
Server
TCP
HTML
OR
OR
OR
OR
Non-Persistent
10 RTT
Client
Server
TCP
HTML
TCP
OR
TCP
OR
TCP
OR
TCP
OR
Persistent with parallel
3 RTT
Client
Server
TCP
HTML
OR