ICA

Introduction to Computer Networking Chapter 1

Silvia Giordano

ICA, EPFL

CH-1015 Ecublens
silvia.giordano@epfl.ch
http://icawww.epfl.ch

Introduction

Chapter goal:

- understand TCP/IP and networking terminology
- more depth, detail *later* in course
- approach:
  - Top-down
  - descriptive
  - use Internet as example

Text Books:

"Computer Networking", J. Kurose – K. Ross, Addison Wesley

"TCP/IP illustrated volume I", The protocols, W. Richard Stevens, Addison Wesley (Very detailed, experimental hands-on description of TCP/IP. Also volume III for HTTP)

"Java Network Programming"E. Harold, O'Reilly (for Java sockets)

In this lecture we study computer networks.

We use a top-down approach, starting with socket programming. We will study in this chapter the global picture, which will enable you to get started with writing your first programs. Then in the following chapters, we will study the various components (called "layers"), one by one.

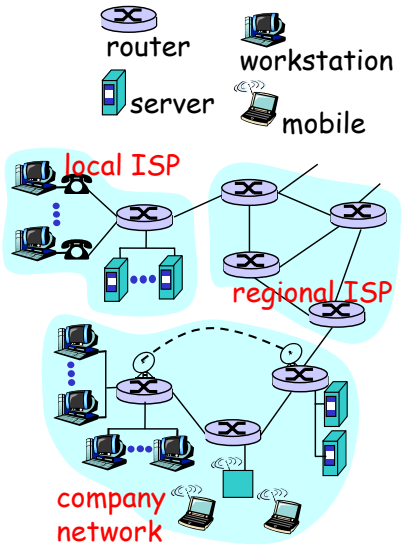
Overview

- what's the Internet
- what's a computer network
- what's a protocol?
- protocol layers, service models
- network edge
- network core
- access net, physical media
- network delay and throughput
- history

What's the Internet: "nuts and bolts" view

- millions of connected computing devices: *hosts, end-systems*
  - pc's workstations, servers
  - PDA's phones, toasters

  running *network apps*
- *communication links*
  - fiber, copper, radio, satellite
- *routers:* forward packets (chunks) of data thru network



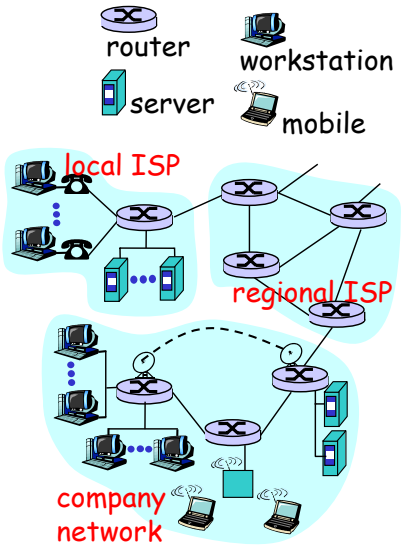
1: Introduction 4

In this course we use the public Internet, a specific computer network (and one which probably most readers have used), as our principle vehicle for discussing computer networking protocols.

The public Internet is a world-wide computer network, that is, a network that interconnects millions of computing devices throughout the world. Most of these computing devices are traditional desktop PCs, Unix-based workstations, and so called servers that store and transmit information such as Web (WWW) pages and e-mail messages. Increasingly, nontraditional computing devices such as Web TVs, mobile computers, pagers, and toasters are being connected to the Internet. (Toasters are not the only rather unusual devices to have been hooked up to the Internet). In the Internet jargon, all of these devices are called hosts or end systems. The Internet applications with which many of us are familiar, such as the Web and e-mail, are network application programs that run on such end systems.

What's the Internet: "nuts and bolts" view

- *protocols*: control sending, receiving of msgs
  - e.g., TCP, IP, HTTP, FTP, PPP
- *Internet*: "network of networks"
  - loosely hierarchical
  - public Internet versus private intranet



End systems, as well as most other "pieces" of the Internet, run protocols that control the sending and receiving of information within the Internet. TCP (the Transmission Control Protocol) and IP (the Internet Protocol) are two of the most important protocols in the Internet. The Internet's principal protocols are collectively known as TCP/IP.

End systems are connected together by communication links. Links are made up of different types of physical media, including coaxial cable, copper wire, fiber optics, and radio spectrum. Different links can transmit data at different rates. The link transmission rate is often called the link bandwidth and is typically measured in bits/second.

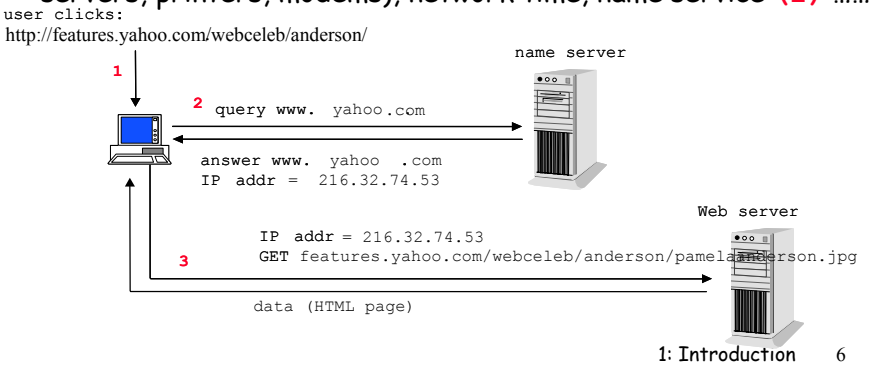
Usually, end systems are not directly attached to each other via a single communication link. Instead, they are indirectly connected to each other through intermediate switching devices known as routers.

The topology of the Internet, that is, the structure of the interconnection among the various pieces of the Internet, is loosely hierarchical. Roughly speaking, from bottom-to-top, the hierarchy consists of end systems connected to local Internet service providers (ISPs) though access networks. An access network may be a so-called local area network within a company or university, a dial telephone line with a modem, or a high-speed cable-based or phone-based access network. Local ISPs are in turn connected to regional ISPs, which are in turn connected to national and international ISPs. The national and international ISPs are connected together at the highest tier in the hierarchy. New tiers and branches (that is, new networks, and new networks of networks) can be added just as a new piece of Lego can be attached to an existing Lego construction.

Computer Network

A computer network provides several network services.

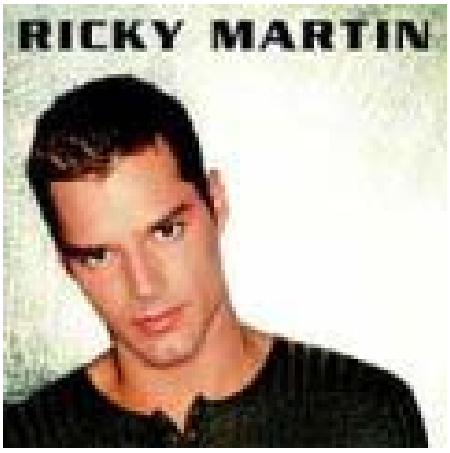
- network services examples:
  - distributed database, Web (3) , file transfer, remote login, email,news, talk, remote processing, resource sharing (file servers, printers, modems), network time, name service (2) .....



What are computer networks used for ?

Computer networks allow people and machines to communicate, using a number of services. The slide shows a small subset of services.

What are computer networks used for?



Computer Network

A computer network is made of

- **network infrastructure:**
  - supports transport of data between computers where distributed applications reside
  - in computers (Ethernet card, modem + software) + in special network devices (bridges, routers, concentrators, switches)

A computer network enables

- **distributed applications**
  - provides service to users and applications on other machines, or to other machines
  - is in computers

A computer network is made of two disctinct subsets of components

- distributed applications are programs running on interconnected computers; a web server, a remote login server, an email exchanger are examples. This is the visible part of what people call “the Internet”. In this lecture we will study the simplest aspects of distributed applications. More sophisticated aspects are the object of lectures called “Distributed Systems” and “Information Systems”.

- the network infrastructure is the collection of systems which are required for the interconnection of computers running the distributed applications. It is the main focus of this lecture.

The network infrastructure problem has itself two aspects:

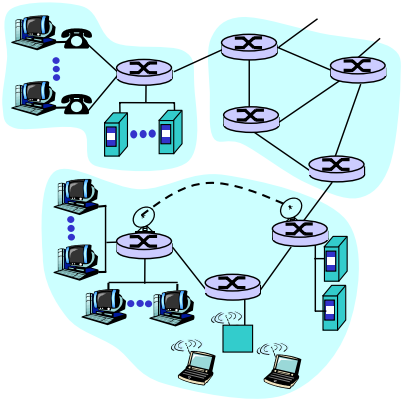
- distance: interconnect systems that are too far apart for a direct cable to be possible

- meshing: interconnect systems together; even in the case of systems located close to each other, it is not possible in non-trivial cases to draw cables from all systems to all systems (combinatorial explosion, cable *salad* management problems).

The distance problem is solved by using a network, such as the telephone network with modems (see later). The meshing problem was originally solved easily because the terminals were not able to communicate with each other, but always has to go through a main computer. The mesh in such cases is reduced to a star network. Today this is solved by a complex set of bridges and routers.

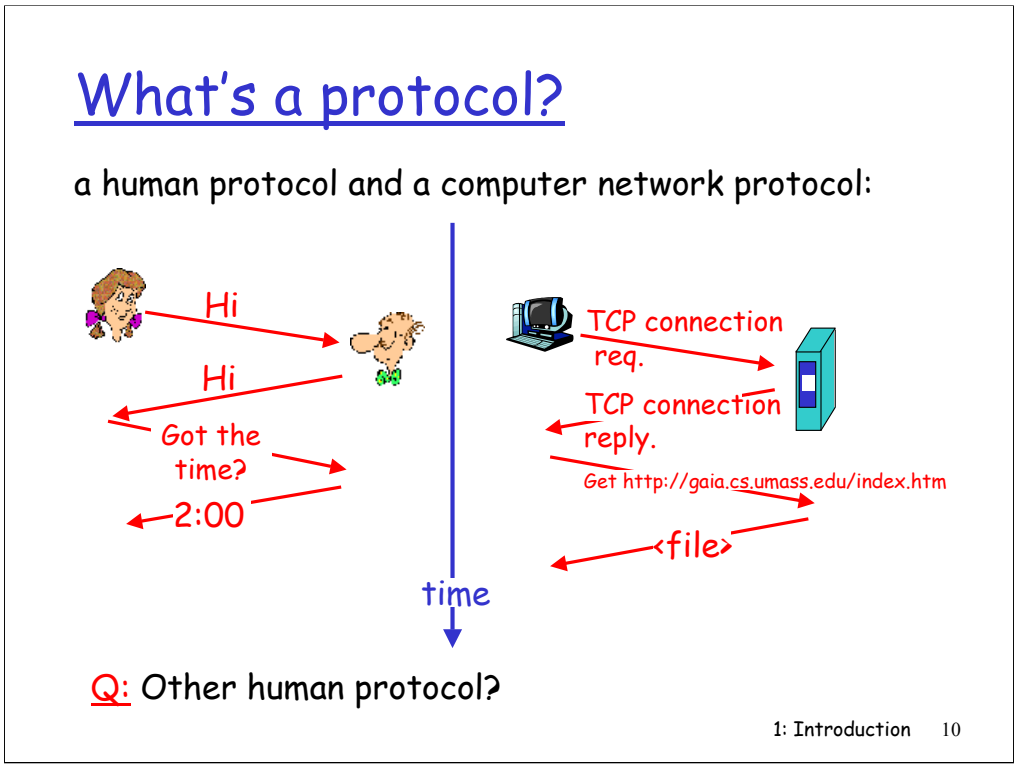
What's the Internet: a service view

- Internet enables distributed applications:
  - WWW, email, games, e-commerce, database., voting,
- *Two services*
  - *connection-oriented*
  - *connectionless*



The Internet allows distributed applications running on its end systems to exchange data with each other. These applications include remote login, file transfer, electronic mail, audio and video streaming, real-time audio and video conferencing, distributed games, the World Wide Web, and much, much more.

The Internet provides two services to its distributed applications: a connection-oriented service and a connectionless service. Loosely speaking, connection-oriented service guarantees that data transmitted from a sender to a receiver will eventually be delivered to the receiver in order and in its entirety. Connectionless service does not make any guarantees about eventual delivery.



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 the figure. Human protocol (or good manners, at least) dictates that one first offers a greeting (the first "Hi" in the figure) to initiate communication with someone else. The typical response to a "Hi" message (at least outside of New York City) is a returned "Hi" message. Implicitly, one then takes a cordial "Hi" response as an indication that one can proceed ahead 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 an unprintable reply that one might receive in New York City) might indicate an unwillingness or inability to communicate. In this case, the human protocol would be to not 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.

As an example of a computer network protocol with which you are probably familiar, consider what happens when you make a request to a Web server, that is, when you type in the URL of a Web page into your Web browser. The scenario is illustrated in the right half of the figure. 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 page it wants to fetch from that Web server in a "get" message. Finally, the Web server returns the contents of the Web document to your computer.

What's a protocol?

protocols define format, order of msgs sent and received among network entities, and actions taken on msg transmission, receipt

human protocols:

- humans
- specific msgs sent
- specific actions taken when msgs received, or other events

network protocols:

- machines rather than humans
- all communication activity in Internet governed by protocols

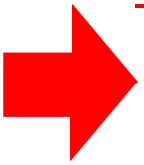
Note that in our human protocol, there are specific messages we send, and specific actions we take in response to the received reply messages or other events (such as no reply within some given amount of time). Clearly transmitted and received messages, and actions taken when these messages are sent or received or other events occur, play a central role in a human protocol. If people run different protocols (for example, if one person has manners but the other does not, or if one understands the concept of time and the other does not) the protocols do not interoperate and no useful work can be accomplished. The same is true in networking--it takes two (or more) communicating entities running the same protocol in order to accomplish a task.

A network protocol is similar to a human protocol, except that the entities exchanging messages and taking actions are hardware or software components of a computer network, components that we will study shortly in the following sections. All activity in the Internet that involves two or more communicating remote entities is governed by a protocol. Protocols in routers determine a packet's path from source to destination; hardware-implemented protocols in the network interface cards of two physically connected computers control the flow of bits on the "wire“ between the two computers; a congestion-control protocol controls the rate at which packets are transmitted between sender and receiver. Protocols are running everywhere in the Internet, and consequently much of this course and also the project is about computer network protocols.

Protocol "Layers"

Networks are complex!

- many "pieces":
  - hosts
  - routers
  - links of various media
  - applications
  - protocols
  - hardware, software

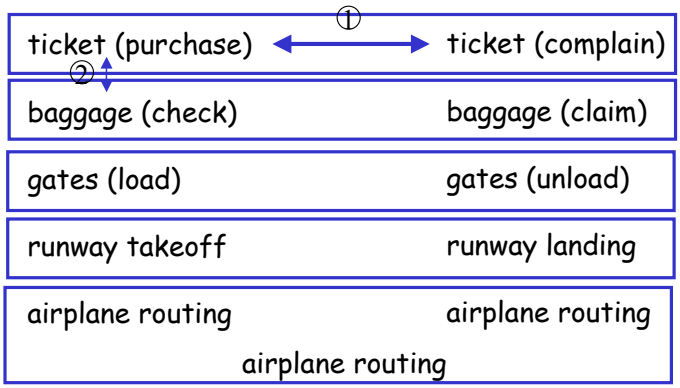


ORGANIZATION BY LAYERS

The airline system: a human example of layering.

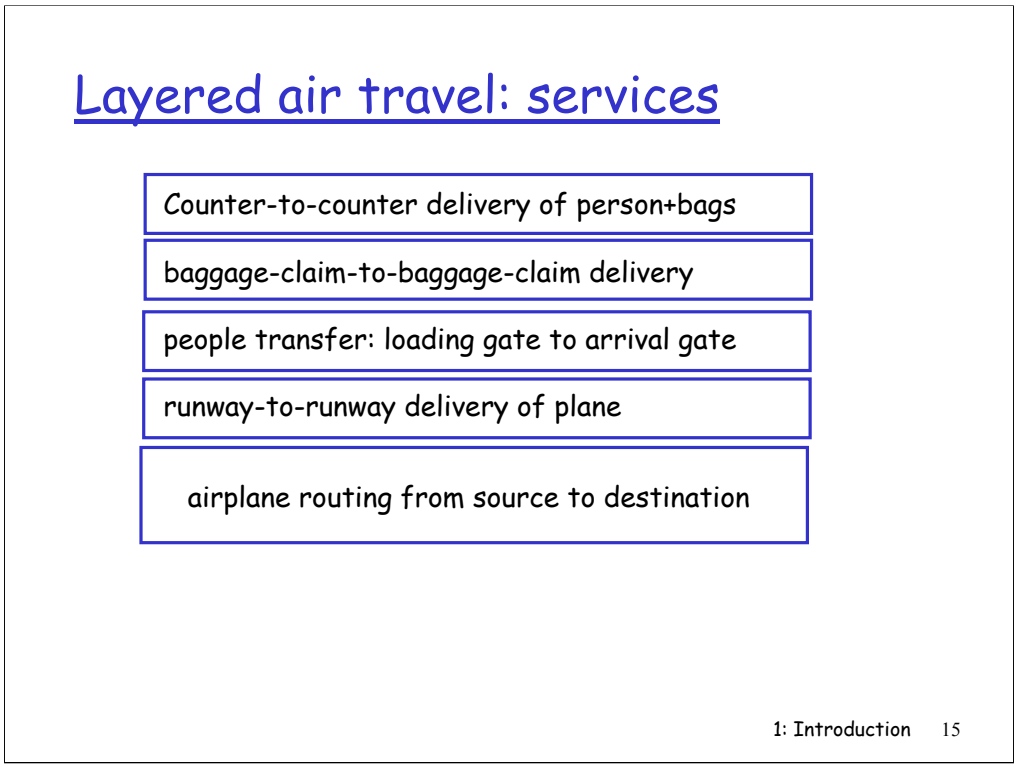
How would you find the structure to describe this complex system that has ticketing agents, baggage checkers, gate personnel, pilots, airplanes, air traffic control, and a worldwide system for routing airplanes? One way to describe this system might be to describe the series of actions you take (or others take for you) when you fly on an airline. You purchase your ticket, check your bags, go to the gate, and eventually get loaded onto the plane. The plane takes off and is routed to its destination. After your plane lands, you de-plane at the gate and claim your bags. If the trip was bad, you complain about the flight to the ticket agent (getting nothing for your effort).

Organization of air travel: a different view

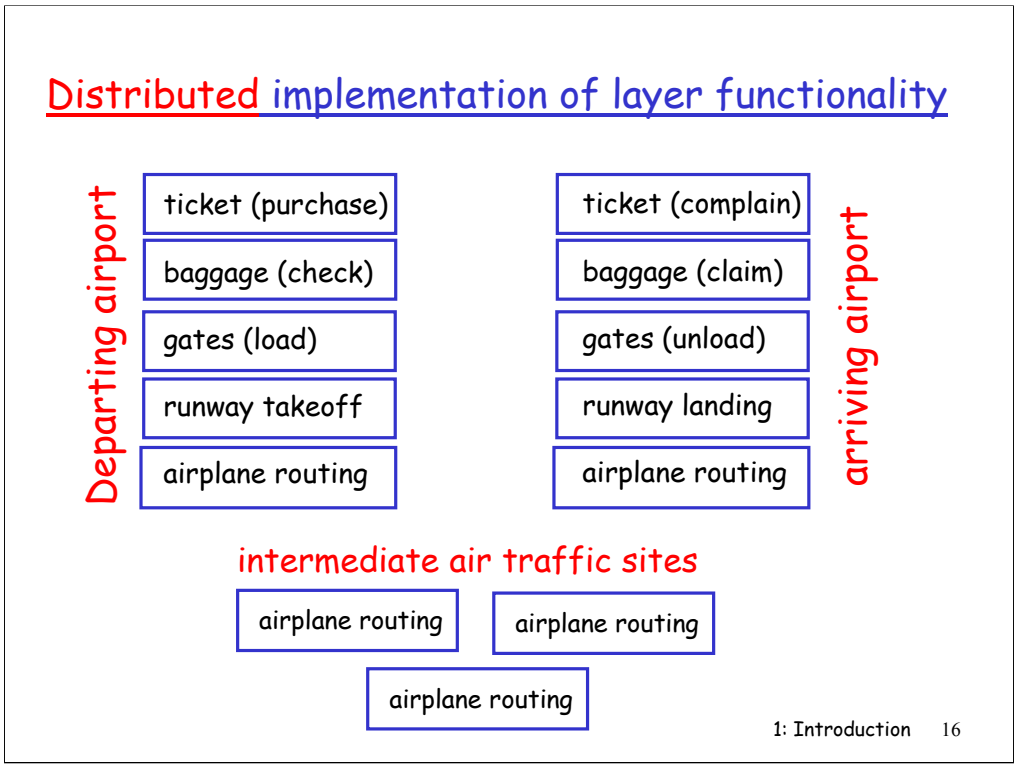


Layers: each layer implements a service
- via its own internal-layer actions (1)
- relying on services provided by layer below (2)

However, this is not very efficient. A better structure results if we can look at the functionality in a *horizontal* manner, as shown in the figure. In the figure, the airline functionality are divided into layers, providing a framework in which we can discuss airline travel. Now, when we want to describe a part of airline travel, we can talk about a specific, well-defined component of airline travel. For example, when we discuss gate functionality, we know we are discussing functionality that sits "below" baggage handling, and "above" takeoff and landing. We note that each layer, combined with the layers below it, implement some functionality, some *service.*



Each layer provides its service by (1) performing certain actions within that layer (for example, at the gate layer, loading and unloading people from an airplane) and by (2) 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).



The layered airline architecture is distributed between the departing and arriving airports.

Why layering?

Dealing with complex systems:

- explicit structure allows identification, relationship of complex system's pieces
  - layered reference model for discussion
- modularization eases maintenance, updating of system
  - change of implementation of layer's service transparent to rest of system
  - e.g., change in gate procedure doesn't affect rest of system

To reduce design complexity, network designers organize protocols--and the network hardware and software that implements the protocols--in **layers.** With a layered protocol architecture, each protocol belongs to one of the layers. It's important to realize that a protocol in layer *n* is *distributed* among the network entities (including end systems and packet switches) that implement that protocol.

As long as the layer provides the same service to the layer above it, and uses the same services from the layer below it, the remainder of the system remains unchanged when a layer's implementation is changed.

Protocol, service and other fancy definitions

- Peer entities
  - two (or more) instances of the same layer
- Protocol and PDU:
  - the "rules of the game" observed by peer entities
  - the data exchanged is called PDU (protocol data unit)
  - there is one protocol (or more) at every layer
- Service and SDU
  - the interface between a layer and the layer above
  - the interface data is called SDU (service data unit)
- Connection
  - a protocol is connection oriented if the peer entity must be synchronized before exchanging useful data; otherwise it is connectionless.

A **protocol** is the formal definition of external behaviour for communicating entities. It defines:

- message formats
- expected actions (message sent, data delivered, abort)

Examples of protocols are:

TCP

UDP

IP

Ethernet

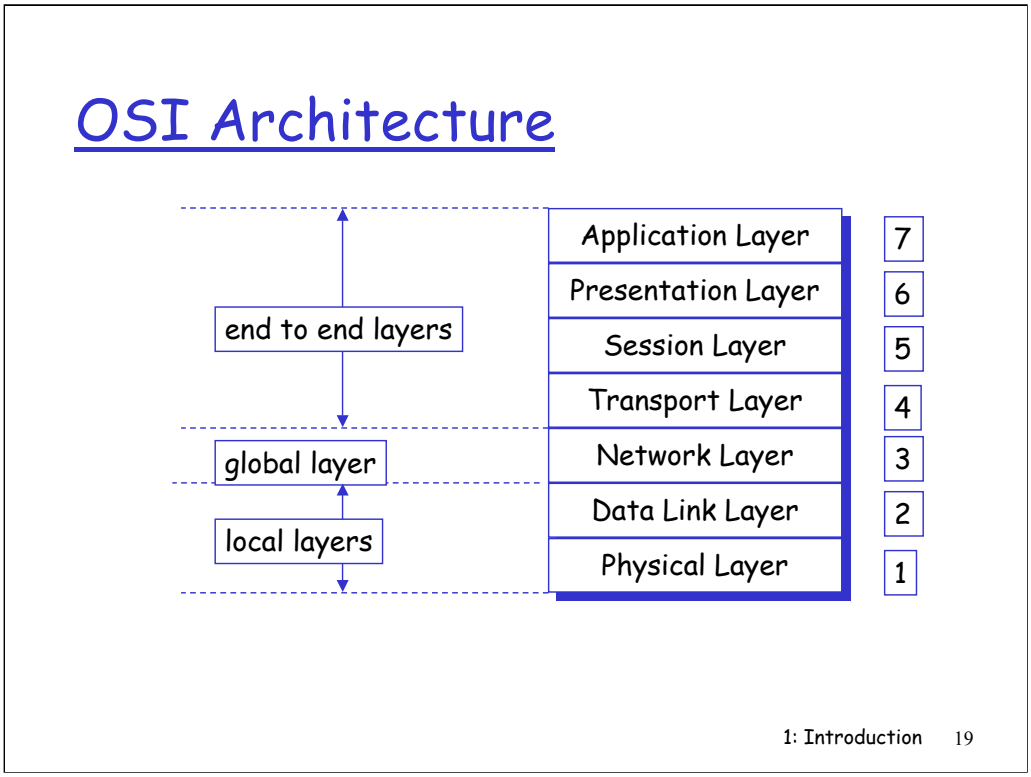
Protocols are connection oriented or connectionless. A connection exists if the communication requires some synchronization of all involved parties before communication can take place. The telephone system is connection oriented: before A can send some information to B, A has to call B (or vice versa) and say "hello". The postal (mail) system is connectionless. If A wants to send some information to B, A can write a letter and mail it, even if B is not ready to read it.

Networking functions are ordered in a layered model:

- layer *n* communicates with other layer *n* entities using the layer *n* protocol, the deat units exchanged are called layer *n* **PDUs** (protocol data units)

- layer *n* uses the **service** of layer *n-1* and offers a service to layer *n+1*.

- entities at the same layer are said **peer entities.**



he OSI architecture defines protocols and service specifications.

It is the official standard, similar to the TCP/IP architecture, but is not much implemented. However, th OSI *model* is used most frequently to describe all systems, including TCP/IP

rchitectures do not interoperate by themselves at the protocol level. For example, the OSI transport protocols ar ot compatible with TCP or UDP. Worse, there is no compatibility at the service level, so it is not possible to us ayer *n* of one architecture and put it on top of layer *n-1* of some other architectures.

here are fortunately exceptions to this statement. Layer interfaces where service compatibility is ofte mplemented are:

the data link layer

the transport layer.

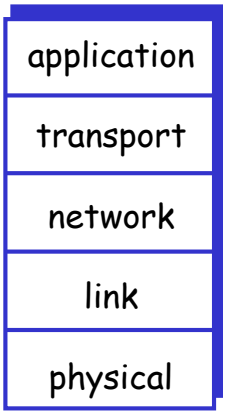
or example, it is possible to use various protocol families over the same local area network (LAN).

he OSI presentation layer is in charge of hiding specific data representation formats. It defines ASN.1, a bstract, universal means for coding all types of data structures. ASN.1 has also become part of the TCP/I rchitecture, in the application layer

he OSI session layer synchronizes events between end-systems, in order for example to support failure recovery is implemented in TCP/IP over a number of application layer protocols and TCP.

Internet protocol stack

- application: supporting network applications
  - ftp, smtp, http
- transport: host-host data transfer
  - tcp, udp
- network: routing of datagrams from source to destination
  - ip, routing protocols
- link: data transfer between neighboring network elements
  - ppp, ethernet
- physical: bits "on the wire"



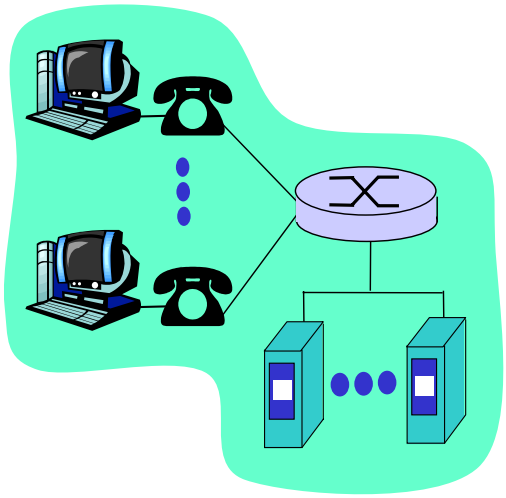
The objective of this and the following slides is to introduce the concept of layers. Like any complex computer system, a network is decomposed into functions. This decomposition is, to a large extend, stable: computer networking people have agreed on a reasonable way to divide the set of functions into what is called "layers".

The decomposition always assumes that the different components can be ordered such that one component interfaces only with two adjacent components. We call "layers" the components.

Layering: logical communication

Each layer:

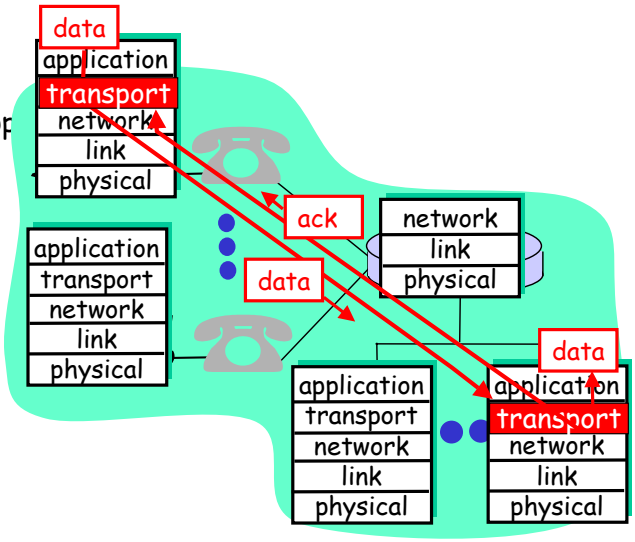
- distributed
- "entities" implement layer gfunctions at each node
- entities perform actions, exchange messages with peers



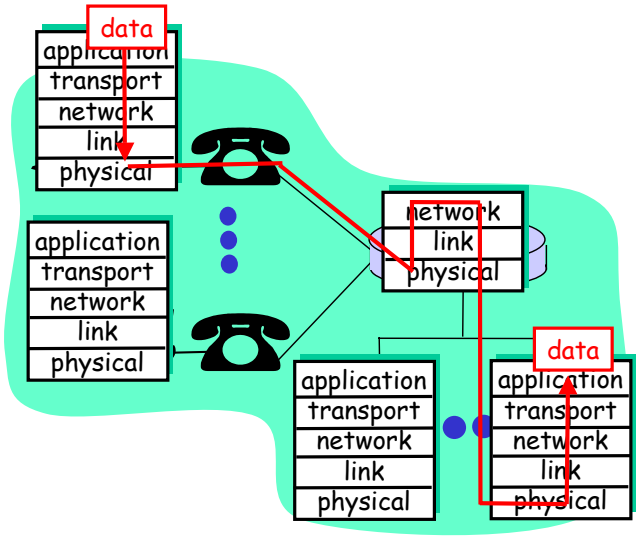
Layering: *logical* communication

E.g.: transport

- take data from app
- add addressing, reliability check info to form "datagram"
- send datagram to peer
- wait for peer to ack receipt
- analogy: post office



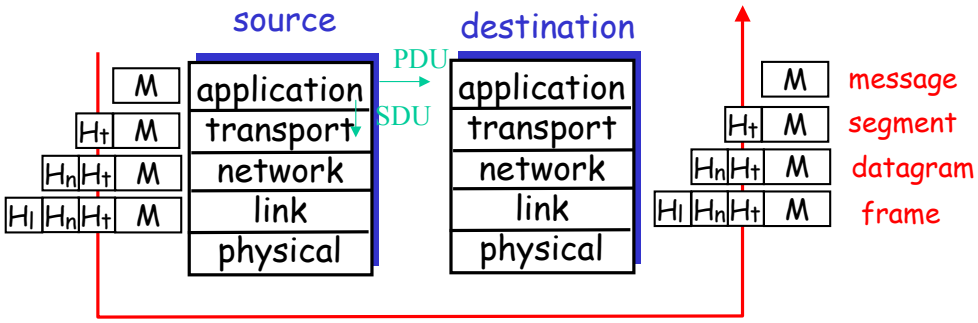
Layering: physical communication



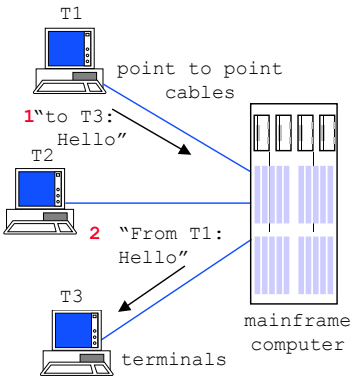
Protocol layering and data

Each layer takes data from above

- adds header information to create new data unit
- passes new data unit to layer below



Physical Layer
Data Link Layer



- physical transmission = **Physical** function
  - bits <-> electrical / optical signals
  - transmit individual bits over the cable: modulation, encoding
- packet transmission = **Data Link** function
  - bits <-> frames
  - bit error detection
  - packet boundaries
  - in some cases: error correction by retransmission
- Modems, Ethernets

We start with the simplest, and oldest, network example: it is a mainframe connected to terminals. In that case, there are mainly two functions

• physical layer: translates bits into electromagnetic waves;

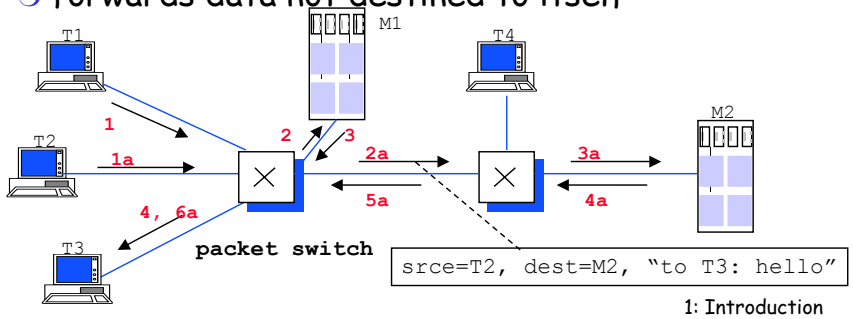
• data link layer: translates packets into bits.

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). For example, Ethernet has many physical layer protocols: one for twisted-pair copper wire, another for coaxial cable, another for fiber, and so on. In each case, a bit is moved across the link in a different way.

Link Layer: The services provided at the link layer depend on the specific link-layer protocol that is employed over the link. For example, some protocols provide reliable delivery on a link basis, that is, from transmitting node, over one link, to receiving node. The process is analogous to the postal worker at a mailing center who puts a letter into a plane that will deliver the letter to the next postal center along the route. Examples of link layers include Ethernet and PPP; in some contexts, ATM and frame relay can be considered link layers. As datagrams typically need to traverse several links to travel from source to destination, a datagram may be handled by different link-layer protocols at different links along its route. For example, a datagram may be handled by Ethernet on one link and then PPP on the next link. The network will receive a different service from each of the different link-layer protocols.

Network Layer

- **Network layer**
  - set of functions require to transport packets end-to- end
  - examples: IP, Appletalk, IPX
- **intermediate system**
  - forwards data not destined to itself



Modern networks have more than physical and data link. The **network layer** is the set of mechanisms that can be used to send packets from one computer to another in the world. There are two types of networks:

With **Packet switching,** data packets can be carried together on the same link. They are differentiated by addressing information. Packet switching is the basis for all data networks today, including the Internet, public data networks such as Frame Relay, X.25, or ATM.

Circuit Switching is the way telephone networks operate. A circuit emulates the physical signals of a direct end-to-end cable. When computers are connected by a circuit switched network, they establish a direct data link over the circuit. This is used today for modem access to a data network.

Modern circuit switches are based on byte multiplexing and are thus similar to packet switches, with the main difference that they perform non-statistical multiplexing (see later in this chapter).

A network has **Intermediate systems** (ISs): those are systems that send data to next ISs or to the destination. Using interconnected ISs saves cable and bandwidth. ISs are known under various terms depending on the context: routers (TCP/IP, AppleTalk,…), switches (X.25, Frame Relay, ATM, telephone), communication controllers (SNA), network nodes (APPN).

The Internet's network layer has two principle components. It has a protocol that defines the fields in the IP datagram as well as how the end systems and routers act on these fields. This protocol is the celebrated **IP protocol**. There is only one IP protocol, and all Internet components that have a network layer must run the IP protocol. The Internet's network layer also contains routing protocols that determine the routes that datagrams take between sources and destinations. The Internet has many routing protocols.

Transport Layer

- Why a transport layer ?
  - **transport layer** = makes network service available to programs
  - is end-to-end only, not in routers
- in TCP/IP there are two transport protocols
  - UDP (user datagram protocol)
    - unreliable
    - offers a datagram service to the application (unit of information is a message)
  - TCP (transmisssion control protocol)
    - reliable
    - offers a stream service (unit of information is a byte)
- an application uses UDP or TCP, it is a designer's choice
  - use for example the socket API: a library of C functions
  - socket also means (IP address, port number)

Physical, data link and network layers are sufficient to build a packet transport system between computers. However, this is not enough for the programmer. When you write a low-level program which uses the network (as we will do in this lecture), you do not handle packets, but data. The primary goal of the **transport layer** is to provide the programmer with an interface to the network.

Second, the transport layer uses the concept of **port**. A port is a number which is used locally (on one machine) and identifies the source and destination of the packet **inside the machine**. We will come back to the concept of ports later in this chapter.

The transport layer exists in two varieties: unreliable and reliable. The unreliable variety simply sends packets, and does not attempt to guarantee any delivery. The reliable variety, in contrast, makes sure that data does reach the destination, even if some packets may be lost from time to time. 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. This service includes guaranteed delivery of application-layer messages to the destination and flow control (that is, sender/receiver speed matching). TCP also segments long messages into shorter segments and provides a congestion control mechanism, so that a source throttles its transmission rate when the network is congested. The UDP protocol provides its applications a connectionless service, which is very much a no-frills service.

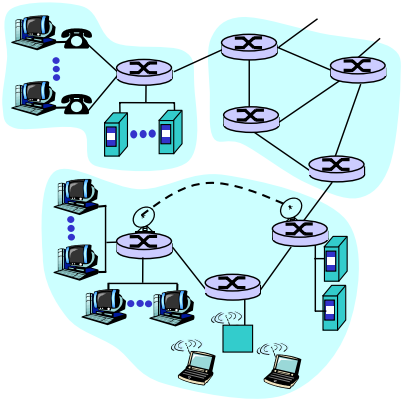
Application Layer

- application layer supports network application
  - applications that are distributed into the network
  - applications that communicates through the network
- Many known protocols
  - ftp: file transfer
  - smtp: email protocol
  - http:web protocol
- an application uses UDP or TCP, it is a designer's choice
  - use for example the socket API: a library of C functions
  - socket also means (IP address, port number)

The Application Layer is responsible for supporting network applications. The application layer includes many protocols, including HTTP to support the Web, SMTP to support electronic mail, and FTP to support file transfer. We shall see in Chapter 2 that it is very easy to create our own new application-layer protocols.

A closer look at network structure:

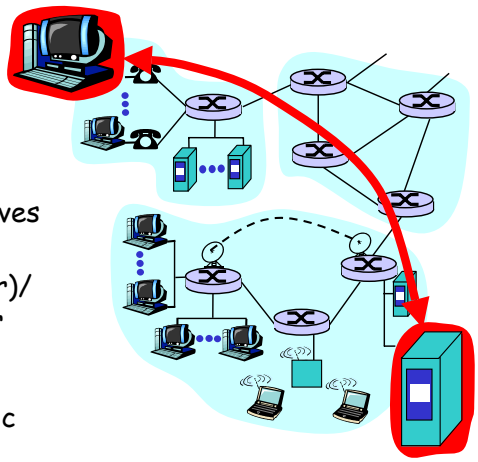
- network edge: applications and hosts
- network core:
  - routers
  - network of networks
- access networks, physical media: communication links



We are now going to delve a bit more deeply into the components of a computer network. We begin at the edge of network and look at the components with which we are most familiar--the computers (for example, PCs and workstations) that we use on a daily basis. Then, moving from the network edge to the network core we have switchs and routers. Finally, we have the access network – the physical link(s) that connect an end system to its edge router – that is, to the first router on a path from the end system to any other end system.

The network edge:

- end systems (hosts):
  - run application programs
  - e.g., WWW, email
  - at "edge of network"
- client/server model
  - client host requests, receives service from server
  - e.g., WWW client (browser)/ server; email client/server
- peer-peer model:
  - host interaction symmetric
  - e.g.: teleconferencing



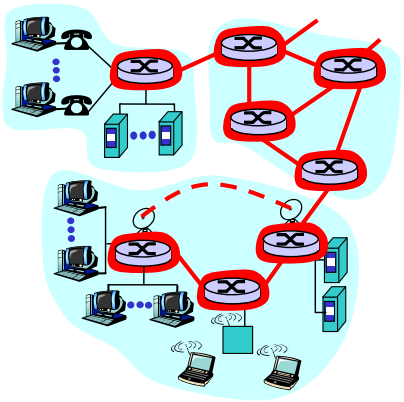
In computer networking jargon, the computers that we use on a daily basis are often referred to as hosts or end systems. They are referred to as hosts because they host (run) application-level programs such as a Web browser or server program, or an e-mail program. They are also referred to as end systems because they sit at the edge of the network.

Hosts are sometimes further divided into two categories: clients and servers. Informally, clients often tend to be desktop PCs or workstations, whereas servers are more powerful machines. But there is a more precise meaning of a client and a server in computer networking. In the so-called client/server model, a client program running on one end system requests and receives information from a server running on another end system. This client/server model is undoubtedly the most prevalent structure for Internet applications. The Web, e-mail, file transfer, remote login (for example, Telnet), newsgroups, and many other popular applications adopt the client/server model.

The other model used in computer networks is referred to as peer-to-peer model. In this model the two hosts takes the same role and run the same programs. A typical example of peer-to-peer application is the teleconferencing.

The Network Core

- mesh of interconnected routers
- *the* fundamental question: how is data transferred through net?
  - circuit switching: dedicated circuit per call: telephone net
  - packet-switching: data sent thru net in discrete "chunks"



The network core is the mesh of routers that interconnect the end systems. In the figure, we highlights the network core in the thick, shaded lines.

There are two fundamental approaches towards building a network core: circuit switching and packet switching. In circuit-switched networks, the resources needed along a path (buffers, link bandwidth) to provide for communication between the end systems are reserved for the duration of the session. In packet-switched networks, these resources are not reserved; a session's messages use the resource on demand, and as a consequence, may have to wait (that is, queue) for access to a communication link.

The ubiquitous telephone networks are examples of circuit-switched networks. Consider what happens when one person wants to send information (voice or facsimile) to another over a telephone network. Before the sender can send the information, the network must first establish a connection between the sender and the receiver.

In modern packet-switched networks, the source breaks long messages into smaller packets. Between source and destination, each of these packets can take different communication links and packet switches (also known as routers). Packets are transmitted over each communication link at a rate equal to the full transmission rate of the link. Most packet switches use store-and-forward transmission at the inputs to the links. Store-and-forward transmission means that the switch must receive the entire packet before it can begin to transmit the first bit of the packet onto the outbound link. Thus store-and-forward packet switches introduce a store-and-forward delay at the input to each link along the packet's route. This delay is proportional to the packet's length in bits. In particular, if a packet consists of L bits, and the packet is to be forwarded onto an outbound link of R bps, then the store-and-forward delay at the switch is L/R seconds.

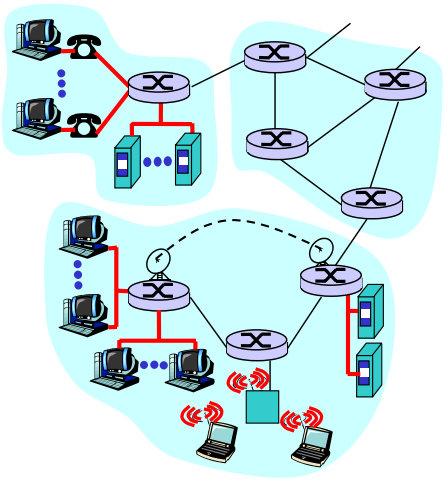
Access networks and physical media

How to connect end systems to edge router?

- residential access nets
- institutional access networks (school, company)
- mobile access networks

Characteristics:

- bandwidth (bits per second) of access network
- shared or dedicated



The access networks are the physical link(s) that connect an end system to its edge router.The figure shows the access networks' links highlighted in thick, shaded lines.

Access networks can be loosely divided into three categories:

o **Residential access networks**, connecting a home end system into the network

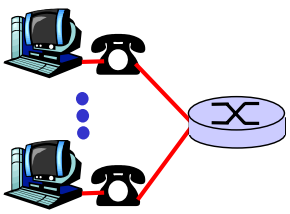
o **Institutional access networks**, connecting an end system in business or educational institution into the network

o **Mobile access networks**, connecting a mobile end system into the network

These categories are not hard and fast; some corporate end systems may well use the access network technology that we ascribe to residential access networks, and vice versa.

Residential access: point to point access

- Dialup via modem
  - up to 56Kbps direct access to router (conceptually)
- ISDN: intergrated services digital network: 128Kbps all-digital connect to router
- ADSL: asymmetric digital subscriber line
  - up to 1 Mbps home-to-router
  - up to 8 Mbps router-to-home
  - ADSL deployment ongoing



A residential access network connects a home end system (typically a PC, but perhaps a Web TV or other residential system) to an edge router. Probably the most common form of home access is by use of a modem over a POTS (plain old telephone system) dialup line to an Internet service provider (ISP). The home modem converts the digital output of the PC into analog format for transmission over the analog phone line. A modem in the ISP converts the analog signal back into digital form for input to the ISP router. In this case, the access network is simply a point-to-point dialup link into an edge router. The point-to-point link is your ordinary twisted-pair phone line. Today's modem speeds allow dialup access at rates up to 56 Kbps (nominal).

Narrowband ISDN technology (Integrated Services Digital Network) allows for all-digital transmission of data from a home end system over ISDN "telephone" lines to a phone company central office. Although ISDN was originally conceived as a way to carry digital data from one end of the phone system to another, it is also an important network access technology that provides higher speed access (for example, 128 Kbps) from the home into a data network such as the Internet.

Two new technologies, asymmetric digital subscriber line (ADSL) and hybrid fiber coaxial cable (HFC) are currently being deployed. ADSL is conceptually similar to dialup modems: It is a new modem technology again running over existing twisted-pair telephone lines, but it can transmit at rates of up to about 8 Mbps from the ISP router to a home end system. The data rate in the reverse direction, from the home end system to the central office router, is less than 1 Mbps. The asymmetry in the access speeds gives rise to the term asymmetric in ADSL. The asymmetry in the data rates reflects the belief that a home user is more likely to be a consumer of information (bringing data into the home) than a producer of information. ADSL uses frequency division multiplexing. In particular, ADSL divides the communication link between the home and the ISP into three non-overlapping frequency bands:

•A high-speed downstream channel, in the 50 kHz to 1 MHz band

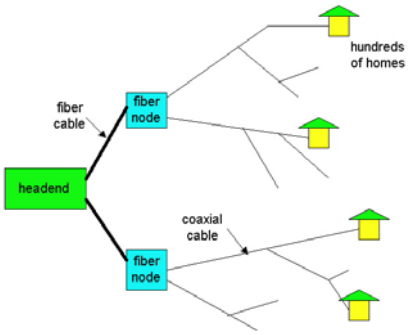
•A medium-speed upstream channel, in the 4 kHz to 50 kHz band

•An ordinary POTS two-way telephone channel, in the 0 to 4 KHz band

One of the features of ADSL is that the service allows the user to make an ordinary telephone call, using the POTS channel, while simultaneously surfing the Web. This feature is not available with standard dialup modems.

Residential access: cable modems

- HFC: hybrid fiber coax
  - asymmetric: up to 10Mbps upstream, 1 Mbps downstream
- network of cable and fil attaches homes to ISP router
  - shared access to router among home
  - issues: congestion, dimensioning
- deployment: available via cable companies, e.g., MediaOne

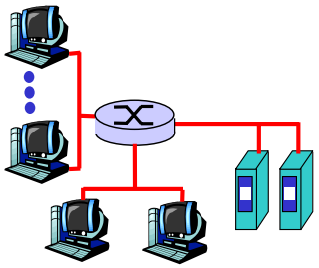


ADSL, ISDN, and dialup modems all use ordinary phone lines, but HFC access networks are extensions of the current cable network used for broadcasting cable television. In a traditional cable system, a cable head end station broadcasts through a distribution of coaxial cable and amplifiers to residences. As with ADSL, HFC requires special modems, called cable modems.

One important characteristic of HFC 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. For this reason, if several users are receiving different Internet videos on the downstream channel, the actual rate at which each user receives its video will be significantly less than the downstream rate. On the other hand, if all the active users are Web surfing, then each of the users may actually receive Web pages at the full downstream rate, as a small collection of users will rarely request a Web page at exactly the same time. Because the upstream channel is also shared, packets sent by two different homes at the same time will collide, which further decreases the effective upstream bandwidth.

Institutional access: local area networks

- company/univ local area network (LAN) connects end system to edge router
- Ethernet:
  - shared or dedicated cable connects end system and router
  - 10 Mbs, 100Mbps, Gigabit Ethernet
- deployment: institutions, home LANs soon
- LANs: chapter 5

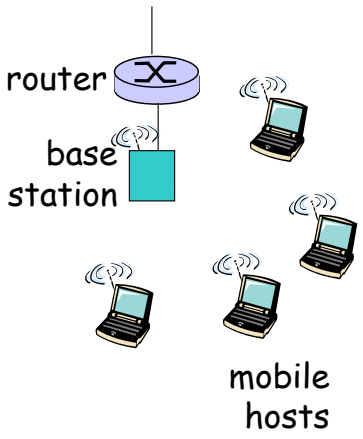


In company access networks, a local area network (LAN) is used to connect an end system to an edge router. There are many different types of LAN technology. However, Ethernet technology is currently by far the most prevalent access technology in company networks. Ethernet operates at 10 Mbps or 100 Mbps (and now even at 1 Gbps). It uses either twisted-pair copper wire or coaxial cable to connect a number of end systems with each other and with an edge router.

The edge router is responsible for routing packets that have destinations outside of that LAN. Like HFC, Ethernet uses a shared medium, so that end users share the transmission rate of the LAN. More recently, shared Ethernet technology has been migrating towards switched Ethernet technology.

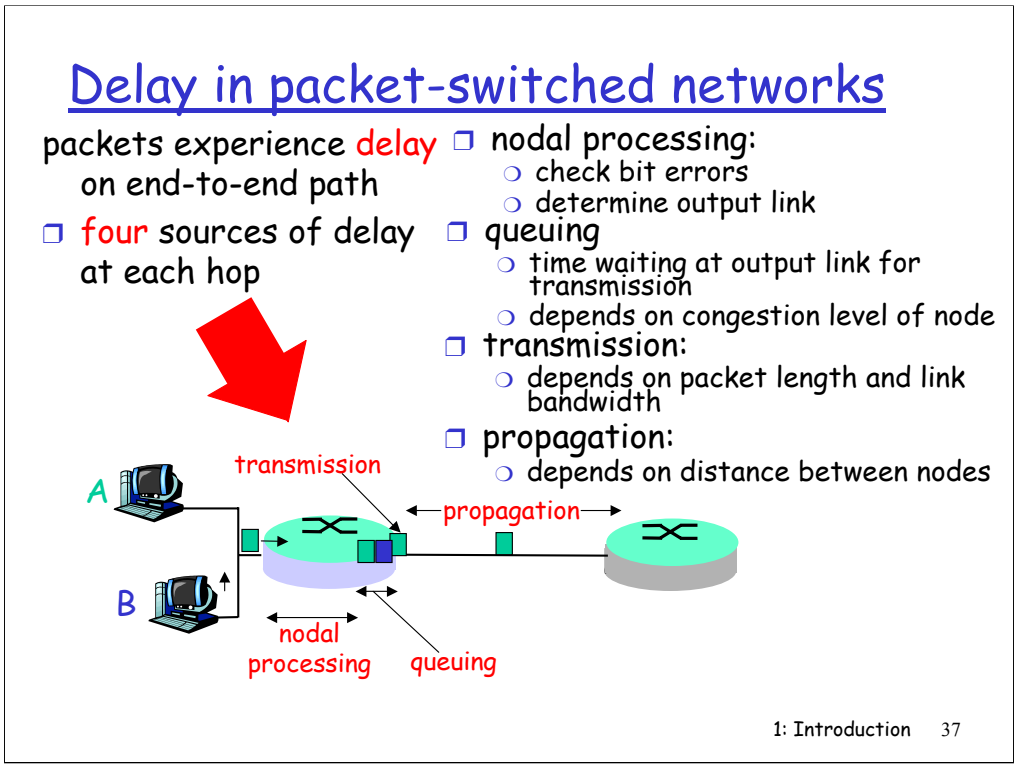
Wireless access networks

- shared *wireless* access network connects end system to router
- wireless LANs:
  - radio spectrum replaces wire
  - e.g., Lucent Wavelan 10 Mbps
- wider-area wireless access
  - CDPD: wireless access to ISP router via cellular network



Mobile access networks use the radio spectrum to connect a mobile end system (for example, a laptop PC or a PDA with a wireless modem) to a base station. This base station, in turn, is connected to an edge router of a data network. An emerging standard for wireless data networking is cellular digital packet data (CDPD).

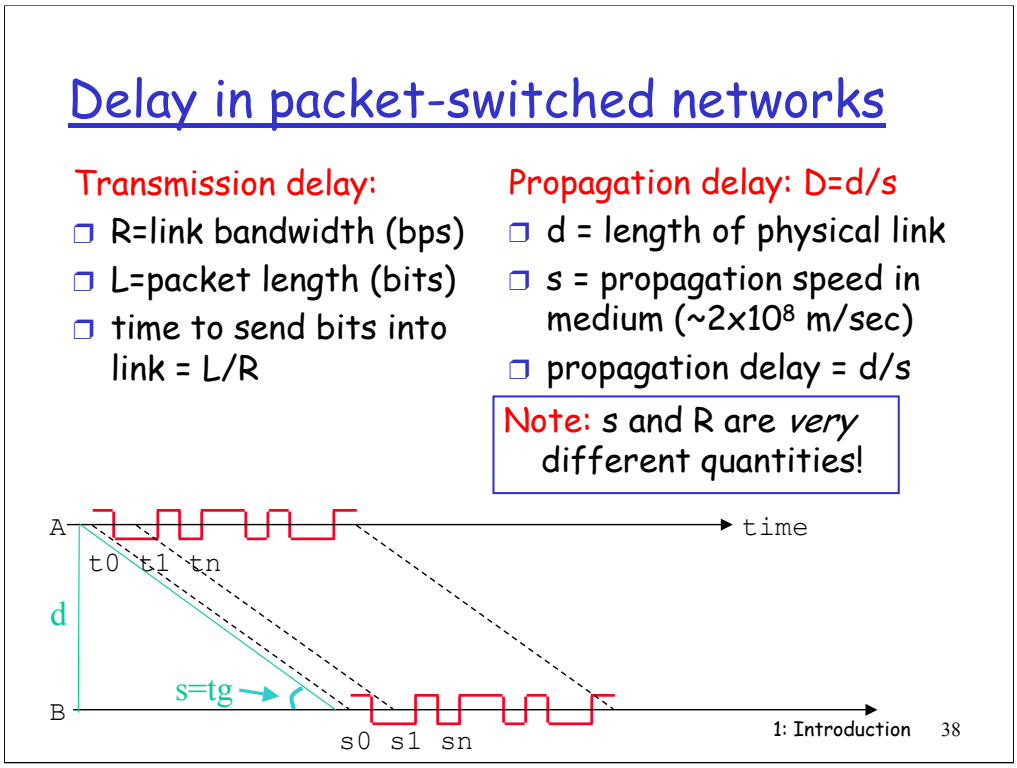
CDPD network operates as an overlay network (that is, as a separate, smaller virtual network, as a piece of the larger network) within the cellular telephone network. A CDPD network thus uses the same radio spectrum as the cellular phone system, and operates at speeds in the tens of Kbits per second.



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 different 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.

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 processing delay can also include other factors, such as the time needed to check for bit-level errors in the packet that occurred in transmitting the packet's bits from the upstream router to router A. Processing delays in high-speed routers are typically on the order of microseconds or less. After this nodal processing, the router directs the packet to the queue that precedes the link to router B. (In Section 4.6 we will study the details of how a router operates.)



Transmission Delay

Assuming that packets are transmitted in first-come-first-serve manner, as is common in the Internet, our packet can be transmitted once all the packets that have arrived before it have been transmitted. 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. The rate R is determined by transmission rate of the link to router B. 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 (also called the store-and-forward delay, as discussed in Section 1.4) is L/R. This is the amount of time required to transmit all of the packet's bits into the link. Transmission delays are typically on the order of microseconds or less in practice.

Propagation Delay

Once a bit is pushed onto 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 bit propagates at the propagation speed of the link. The propagation speed depends on the physical medium of the link (that is, multimode fiber, twisted-pair copper wire, and so on) and is in the range of

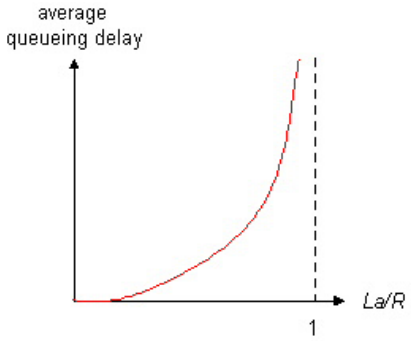
2 • 108 meters/sec to 3 • 108 meters/sec

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. Once the last bit of the packet propagates to node B, it and all the preceding bits of the packet are stored in router B. The whole process then continues with router B now performing the forwarding. In wide-area networks, propagation delays are on the order of milliseconds.

Queueing delay

- R=link bandwidth (bps)
- L=packet length (bits)
- a=average packet arrival rate

traffic intensity = La/R



- La/R ~ 0: average queuing delay small
- La/R -> 1: delays become large
- La/R > 1: more "work" arriving than can be serviced, average delay infinite!

Queuing Delay

At the queue, the packet experiences a queuing delay as it waits to be transmitted onto the link. The queuing delay of a specific packet will depend on the number of other, earlier-arriving packets that are queued and waiting for transmission across the link. The delay of a given packet can vary significantly from packet to packet. If the queue is empty and no other packet is currently being transmitted, then our packet's queuing delay is zero. On the other hand, if the traffic is heavy and many other packets are also waiting to be transmitted, the queuing delay will be long. We will see shortly that the number of packets that an arriving packet might expect to find on arrival is a function of the intensity and nature of the traffic arriving to the queue. Queuing delays can be on the order of milliseconds to microseconds in practice.

Example

- highway with toll booth every 100 Km
- cars (3 m) travel at 100 Km/h
- toll booths services one car per 12 seconds
- a caravan is 10 cars

- the booth's barrier opens after 3 seconds
- N caravans arrive at first booth at the same time NxL/R (so each block of N caravans finds the booth empty)
- cars do 0-100 in 0 seconds (WOW!)

Example

For each packet:

- Processing delay: 12X10 = 120 seconds
- Average queuing delay: $((L/R) \sum_{1}^{(N-1)} i)/N$ = (N-1)L/2R seconds
- Transmission delay: 10x3 = 30 seconds
- Propagation delay: 100/100 = 1 hour

Examples

- At time 0, computer A sends a packet of size 1000 bytes to B; at what time is the packet received by B (s = 2e+08 m/s) ?

distance	20 km	20000 km	2 km	20 m
bit rate	10kb/s	1 Mb/s	10 Mb/s	1 Gb/s
1-way propagation	0.1 ms	100 ms	0.01 ms	0.1 μs
transmission	800 ms	8 ms	0.8 ms	8 μs
reception time	800.1 ms	108 ms	0.81 ms	8.1 μs
	modem	*satellite*	*LAN*	*Hippi*

Throughput

- **Throughput** (*am* thruput, *f* débit utile, *g* Durchsatz) for a transmission system or a communication flow =

  number of useful data bits / time unit units:
  - b/s, kb/s, Mb/s
- Example 1:

  PCM voice (8 kHz, 8 bits per sample -> 64 b/s)

  throughput = 64 kb/s

The throughput defines how much data can be moved by time unit. It is equal to the bit rate if there is no protocol (example 1). However, in most practical cases, the throughput is less than the bit rate for two reasons:

- protocol overhead: protocols like UDP use some bytes to transmit protocol information. This reduces the throughput. If you send one-byte messages with UDP, then for every byte you create an Ethernet packet of size 1 + 8 + 20 + 26 = 53 bytes; thus the maximum throughput you could ever get at the UDP service interface if you use a 64 kb/s channel would be 1.2 kb/s.

- protocol waiting times: some protocols may force you to wait for some event.

Example

Each 9 cars there is a police car

- 1 caravan in 30 seconds -> 120c/h
- 1200 cars/h

- Throughput (i.e. without police)
  - 1080 cars/h

Internet History

1961-1972: Early packet-switching principles

- 1961: Kleinrock - queuing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in military nets
- 1967: ARPAnet conceived by Advanced Reearch Projects Agency
- 1969: first ARPAnet node operational
- 1972:
  - ARPAnet demonstrated publicly
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
  - ARPAnet has 15 nodes

Internet History

1972-1980: Internetworking, new and proprietary nets

- 1970: ALOHAnet satellite network in Hawaii
- 1973: Metcalfe's PhD thesis proposes Ethernet
- 1974: Cerf and Kahn - architecture for interconnecting networks
- late70's: proprietary architectures: DECnet, SNA, XNA
- late 70's: switching fixed length packets (ATM precursor)
- 1979: ARPAnet has 200 nodes

Cerf and Kahn's internetworking principles:

- minimalism, autonomy - no internal changes required to interconnect networks
- best effort service model
- stateless routers
- decentralized control

define today's Internet architecture

Internet History

1980-1990: new protocols, a proliferation of networks

- 1983: deployment of TCP/IP
- 1982: smtp e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: ftp protocol defined
- 1988: TCP congestion control
- new national networks: Csnet, BITnet, NSFnet, Minitel
- 100,000 hosts connected to confederation of networks

Internet History

1990's: commercialization, the WWW

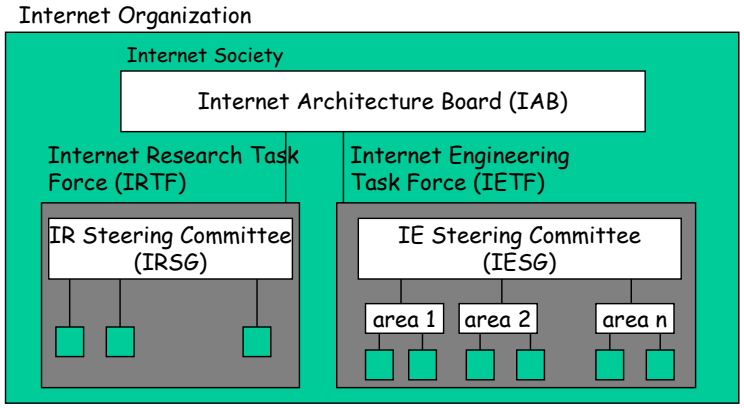
- Early 1990's: ARPAnet decomissioned
- 1991: NSF lifts restrictions on commercial use of NSFnet (decommissioned, 1995)
- early 1990s: WWW
  - hypertext [Bush 1945, Nelson 1960's]
  - HTML, http: Berners-Lee
  - 1994: Mosaic, later Netscape
  - late 1990's: commercialization of the WWW

Late 1990's:

- est. 50 million computers on Internet
- est. 100 million+ users
- backbone links runnning at 1 Gbps

The Internet Organization

- Coordinates development of Internet standards = (TCP/IP standards)



RFCs = official Internet documentation:

standards, other documents maintained by INTERNIC, shadows at ftp.switch.ch

INTERNIC manages IP addresses and domain names

IANA manages constant names (eg: port 53 for DNS)

At the technical and developmental level, the Internet is made possible through creation, testing, and implementation of **Internet standards.** These standards are developed by the Internet Engineering Task Force (IETF). The IETF standards documents are called **RFCs** (request for comments). RFCs started out as general request for comments (hence the name) to resolve architecture problems that faced the precursor to the Internet. RFCs, though not formally standards, have evolved to the point where they are cited as such. RFCs tend to be quite technical and detailed. They define protocols such as TCP, IP, HTTP (for the Web), and SMTP (for open-standards e-mail). There are more than 2,000 different RFCs.

Chapter 1: Summary

Covered a "ton" of material!

- Internet overview
- what's a protocol?
- layering and service models
- network edge, core, access network
- delay & throughput
- history

You now hopefully have:

- context, overview, "feel" of networking
- more depth, detail *later* in course