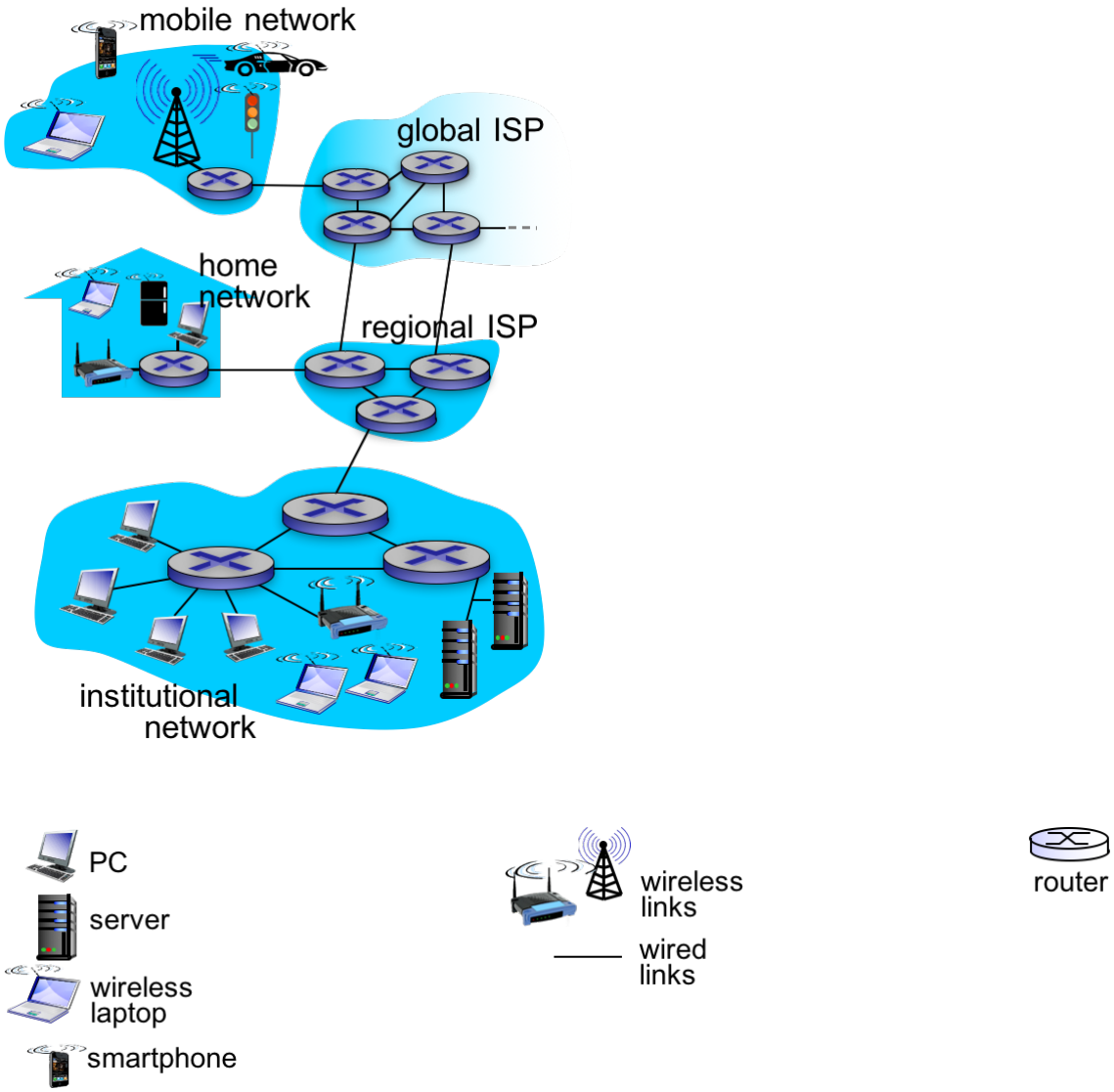
1. Introduction to the Internet

1.1. Computer Networks

- computer networks are everywhere
- they form an essential part of our infrastructure
- used
    - at home
    - at work
    - by governments
    - on the move
    - ...
- there are many different types of networks and standards
- we will concentrate on the Internet

1.2. The Internet



- (the above figure is taken from the book by Kurose and Ross)
- *an internet* is a federation of computer networks, connected by routers
- *the Internet* is the world-wide federation of packet-switched networks running TCP/IP
- important applications include *email* and the *World Wide Web (WWW)*

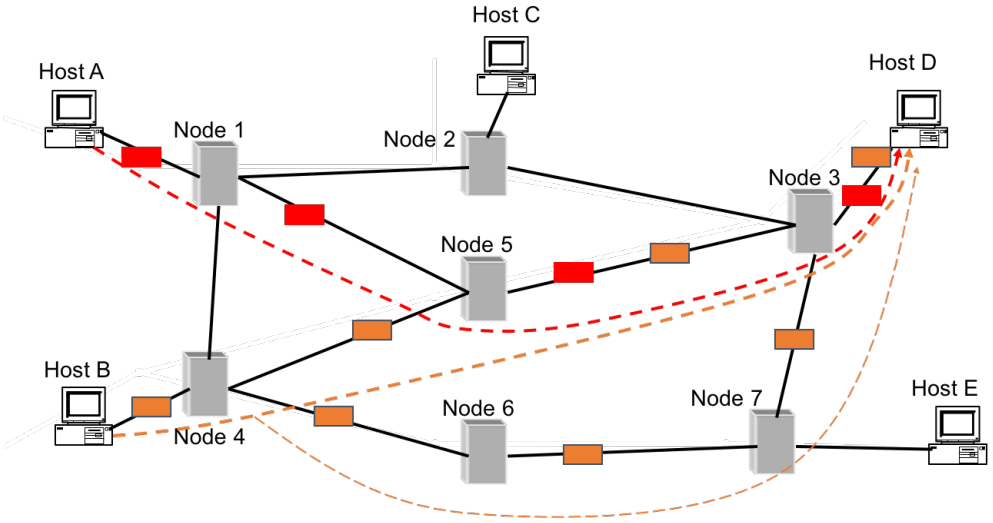
1.3. Circuit Switching

- early communication networks evolved from telephone systems
- used physical pair of wires between two parties to form a dedicated *circuit*
- *circuit switching* was the task of deciding which circuit to use when two parties wanted to communicate
- the circuit is reserved for the two parties during communication
- so it is not available to other parties

1.4. Packet Switching

- the Internet uses *packet switching* which is considered more efficient
- packet switching
  - divides data into small blocks, called packets
  - allows multiple users to share a network
  - includes identification of the intended recipient in each packet
  - routers each have information about where to send each packet next

1.5. Packet Switching Example



1.6. Brief History of the Internet

- (1957) Advanced Research Projects Agency (ARPA) established by US Department of Defense
- (1968-9) first packet-switching networks
- (1972) Telnet
- (1973) File Transfer Protocol (FTP); ARPANET goes international:
  - University College, London (UK)
  - Royal Radar Establishment (Norway)
- (1974) design of TCP (Transmission Control Protocol)
- (1977) email
- (1982) TCP and IP (Internet Protocol) used for ARPANET
- (1984) DNS (Domain Name Service) introduced
- (1991) WWW released

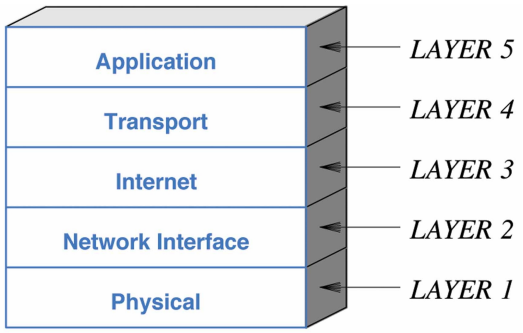
1.7. Communication Protocols

- communication always involves at least two entities
  - one that *sends* information and another that *receives* it
- all entities in a network must *agree* on how information will be represented and communicated
  - the way that electrical signals are used to represent data
  - procedures used to initiate and conduct communication
  - the format of messages
- all communicating parties follow the same *set of rules*, a set of *specifications*
- a specification for network communication is called a communication *protocol*

1.8. Protocols and Layering

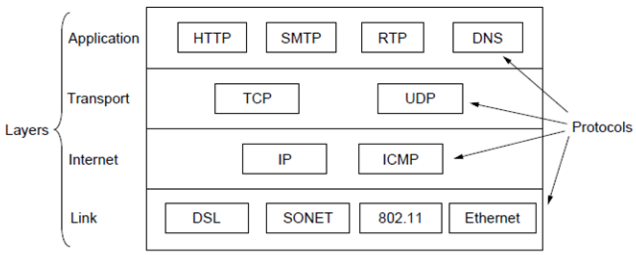
- computer networks are complex systems including both hardware and software
- rather than a single, huge specification for all possible forms of communication, designers divide the communication problem into subparts, called *layers*
- the interfaces between the layers are defined by *protocols*
- layers provide for *modularity*, making implementation and changes easier
- the combination of layers is sometimes called a *protocol stack*

1.9. TCP/IP 5-layer Reference Model



- *physical* layer corresponds to the basic network hardware
- *network interface*, or *link*, layer specifies how machines on the same medium/network communicate
- *Internet* layer specifies how packets are routed from one network to another over the Internet
- *transport* layer specifies how to communicate with particular *processes* (running programs) on machines
- *application* layer specifies how applications (e.g., Web, email) use the Internet

1.10. TCP/IP layers with some protocols

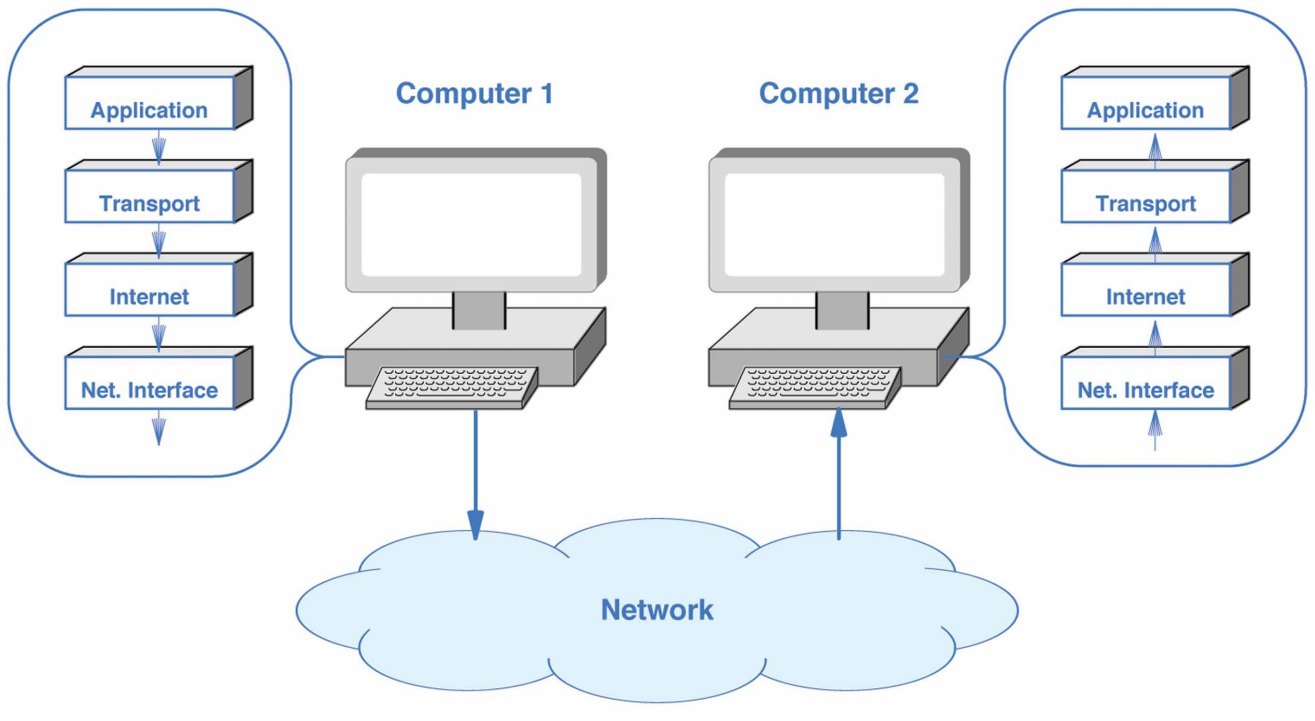


- HTTP = HyperText Transfer Protocol; SMTP = Simple Mail Transfer Protocol;

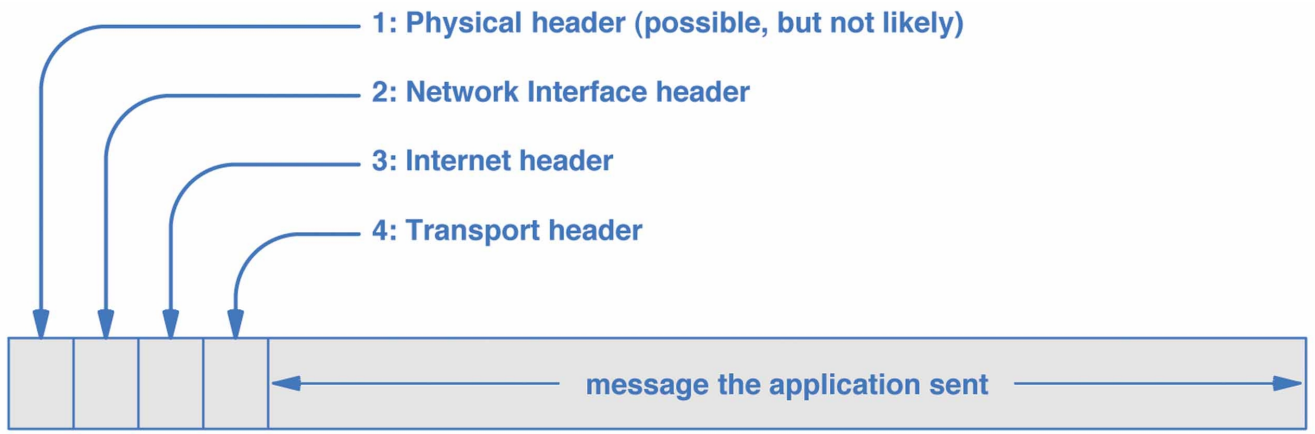
RTP = Real-time Transport Protocol; DNS = Domain Name System

- TCP = Transmission Control Protocol; UDP = User Datagram Protocol
- IP = Internet Protocol; ICMP = Internet Control Message Protocol
- DSL = family of Digital Subscriber Line technologies; SONET = Synchronous Optical Networking protocol; 802.11 = a set of wireless protocols (WiFi)

1.11. Data Passing Through Layers



1.12. Headers and Layers



1.13. Internet Communication Paradigms

- Internet supports two basic communication paradigms:
  - *stream* paradigm
  - *message* paradigm

stream paradigm	message paradigm
connection-oriented	connectionless
one-to-one communication	many-to-many communication
sequence of individual bytes	sequence of individual messages
arbitrary length transfer	each message limited to 64 Kbytes
used by most applications	often used for multimedia applications
built on TCP protocol	built on UDP protocol

- we will focus mostly on the stream paradigm

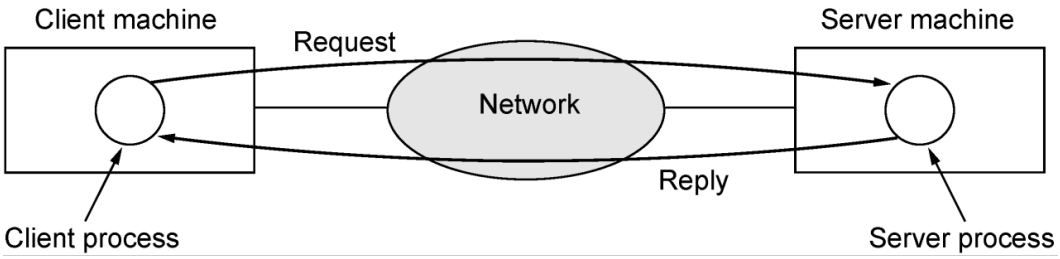
1.14. Connection-Oriented Communication

- the Internet stream service is connection-oriented
- two applications must request that a connection be created
- by contrast, connectionless communication allows messages to be sent at any time
- once it has been established, the connection allows the applications to send data in either direction
- each pair of applications has its *own* (logical) connection
- finally, when they finish communicating, the applications request that the connection be terminated

1.15. Designing Applications

- networked applications follow a small number of design patterns
- two most common are *client-server* and *peer-to-peer*
- in client-server, there is usually a single server and many clients
- in peer-to-peer (e.g. Skype, BitTorrent) there is no single server
- we will focus on client-server

1.16. Client-Server Model



The client-server model involves requests and replies.

1.17. Client Software

- is an arbitrary application program that becomes a client temporarily when *remote* access is needed, but also performs other computation
- is *invoked* directly by a user, and executes only for one session
- runs *locally* on a user's personal computer
- actively *initiates* contact with a server
- can access multiple services as needed, but usually contacts one remote server at a time

1.18. Server Software

- is a special-purpose, *privileged* program
- is dedicated to providing one service that can handle multiple remote clients at the same time
- is invoked automatically when a system *boots*, and continues to execute through many *sessions*
- runs on a large, powerful computer
- waits *passively* for contact from arbitrary remote clients
- accepts contact from arbitrary clients, but offers a single service

1.19. Server Identification

- Internet protocols divide identification into two pieces:
  - an *identifier for the computer* on which a server runs
  - an *identifier for a service* on the computer
- identifying a computer
  - each computer on the Internet is assigned a unique identifier known as an Internet Protocol address (*IP address*)
    - for IPv4, this is a *32-bit* quantity
    - for IPv6, this is a *128-bit* quantity
  - 4 bytes of an IPv4 address are written as $n_1.n_2.n_3.n_4$ where each n_i is a decimal number, e.g., `18.23.0.22`
  - a client process must specify the IP address of the machine on which the server process is running
  - to make server identification easy for humans, each computer is also assigned a *domain name*
  - the *Domain Name System* (DNS) is used to translate a name into an IP address (see later)
  - so a user specifies a name such as `www.dcs.bbk.ac.uk` rather than an IP address

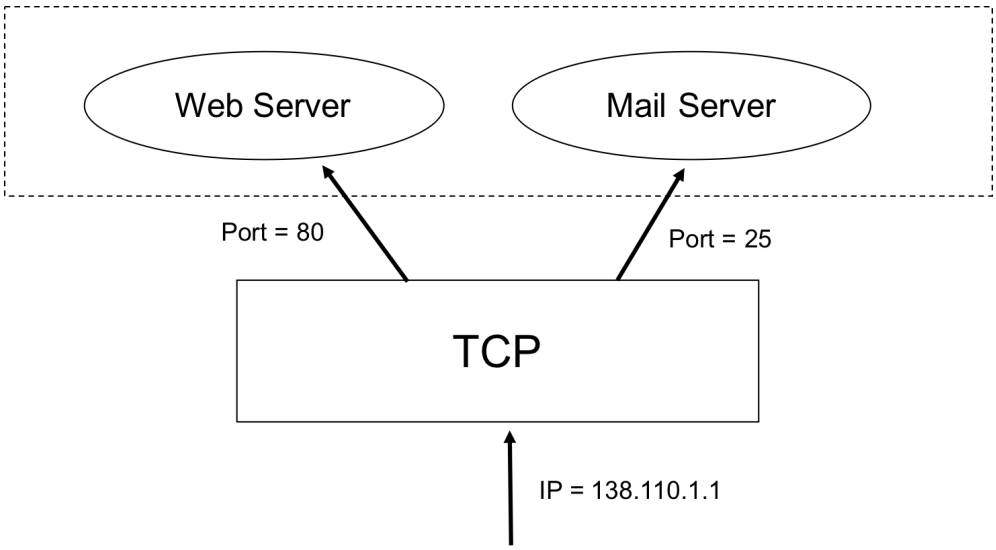
1.20. IPv6 Addresses

- IPv6 addresses are 128-bits
- conventional notation is a series of (up to 8) *blocks* or *fields* of 4 hexadecimal numbers each
- example is: `5f05:2000:80ad:5800:0058:0800:2023:1d71`
- easy to convert hexadecimal numbers to binary
- there are a number of standardised simplifications:
  - leading zeroes of a block can be omitted, so `0058` can be written as `58`
  - blocks of all zeroes can be replaced by `::`, so `0:0:0:0:0:0:0:1` can be written as `::1`
  - an IPv4-compatible IPv6 address can be written as `::` followed by an IPv4 address in dotted decimal notation

1.21. Service Identification

- each service available on the Internet is assigned a unique *16-bit* identifier
- this identifier is known as a protocol port number (or *port number*), e.g.,
  - (sending) email uses port number 25 by default
  - the web uses port number 80 by default
- when a server process begins execution
  - it registers with its local OS by specifying the port number for its service
- when a client process contacts a computer to request a service
  - the request contains the port number for the service
  - as well as the IP address for the computer
- when a request arrives at a server computer
  - TCP/IP software uses the port number in the request to determine which application (process) should handle the request

1.22. TCP Demultiplexing



1.23. A specific example

- suppose I want to retrieve a web page from `www.w3.org`
- my browser will use DNS to find the IP address `128.30.52.37`
- my browser will compose a message based on HTTP asking to get the page
- HTTP will ask TCP to connect to port 80 on `128.30.52.37`
- TCP will ask IP to send the message to `128.30.52.37`
- IP will send the message to a router on the local network
- this router will send the message to another router
- ...
- the router on the local network for `128.30.52.37` will receive the message
- it will send the message to `128.30.52.37`
- IP will receive it and pass it up to TCP
- TCP will see that it is for port 80 and will pass it to the web server process
- the web server will interpret the HTTP and send the page to my browser

1.24. Links to more information

- Brief History of the Internet
- The web resources for Tanenbaum's Computer Networks book, Chapter 1
- Wikipedia articles on the Internet, packet switching and IP addresses.

See Chapters 1, 2 and 3 of [Comer] and Chapter 1 of [Tanenbaum] and of [Kurose and Ross].